Glossary

ABM Agent Based Modelling is a technique whereby complex systems are analysed by simulating the actions of the autonomous agents they are comprised of. 3

attribute Data that defines an object. 9

class A template for objects which defines attributes and methods. 9

class diagram A static representation of an object oriented system which displays the system's classes and the relationships between them. 10

floor A plane which agents can travel across in MassMotion simulations. 9

inheritance A sub-class that 'inherits' from a super-class takes on all the properties of that super-class. 11

L-SATS Low Speed Autonomous Transport System: in this project an L-SATS is exemplified by a fleet of autonomous pods which operate in pedestrianised areas. 3

MassMotion A leading crowd simulation software package used as a platform for the simulations created in this project. 6

method Behaviour that defines an object. 9

object Instance of a class. 10

Object Oriented Programming A type of computer programming in which data structures called objects are created which contain attributes and methods. 9

portal Locations where agents can enter or exit MassMotion simulations. 9

random seed A number used to initialise a pseudorandom number generator. 21

social forces algorithm The algorithm that determines agent behaviour between waypoints in MassMotion. 11

UML Unified Modelling Language is a general-purpose language in software engineering used to provide a standard way of visualising the design of a system. 9

Contents

1 Introduction

1.1 Context

The automotive sector is currently experiencing a period of drastic change as the market has been disrupted by four technologically-driven trends: diverse mobility, autonomous driving, electrification and connectivity [1]. These changes have spurred innovation in new transport systems, to replace those which are no longer suited to modern environments. 'Low Speed Autonomous Transport Systems L-SATSs have been proposed as a 'last mile' transport solution. Modern cities typically suffer from high levels of vehicle congestion and increasingly, city policies are focusing on reducing the number of automobiles in city centres. L-SATSs can offer a novel service whereby slow-moving vehicles travel in pedestrianised areas, freeing up road space for public use. This project will use Agent Based Modelling (ABM), to scrutinise the effectiveness of this L-SATS as a future transport solution for city centres.

Diverse mobility is principally characterised by the shift away from individual vehicle ownership towards shared mobility models, pioneered by companies such as Uber. However, increases in diverse mobility have also been caused by city policies which have discouraged the use of private vehicles and internal combustion engine vehicles in city centres. This 'pedestrianisation' of city centres is being promoted in many cities throughout the world; there are plans to ban vehicle traffic from congested areas in London such as Oxford Street and in Barcelona authorities are seeking to re-purpose roads into public spaces. The changing design of cities has contributed to the development of L-SATSs, since a fleet of small autonomous vehicles operating in pedestrianised areas could become the most viable last mile transport solution.

The development of L-SATSs has been influenced by the other disruptive trends facing the automotive industry. It is desirable for these vehicles to be autonomous since drivers are costly, error-prone and wasteful of space. Electric powertrains are environmentally-friendly, and connectivity facilitates the efficient use of a fleet of vehicles. Although there are currently technical and regulatory challenges facing new transport systems with these characteristics, as these systems become established the trends are expected to accelerate and influence each other. Systems such as the L-SATS have hurdles to overcome including safety, profitability and hesitant public opinion, but these are likely to reduce as the transport sector adapts to the disruptive trends and the public becomes acclimatised to the new systems.

Figure 1: Photograph of an L-SATS comprised of autonomous pods

1.2 Current Progress

The UK government is sponsoring initiatives which are trialling new forms of autonomous vehicle technology on the public roads and in communal spaces, in a bid to improve the country's expertise in this sector. One of these initiatives is UK Autodrive, a three-year long project testing connected and self-driving vehicles in Coventry and Milton Keynes. One of the key components of UK Autodrive is a trial of a L-SATS, consisting of self-driving pods which operate in pedestrianised zones in Milton Keynes city centre. This project focuses on analysing the viability of using a fleet of these autonomous pods to provide a taxi-service in central Milton Keynes.

It is insufficient to ensure that a transport system is reliable and safe without justification that it is cost effective, otherwise it can never be realised in commercial situations. This project will seek to evaluate the commercial viability of the L-SATS consisting of autonomous pods operating in Milton Keynes, since the system is technologically feasible but not necessarily profitable. As part of the project, sensors were installed throughout Milton Keynes city centre which collect data on the number and directions of pedestrians which pass them. This data has been used to estimate the how the busyness of the city centre varies depending on the day of the week and time of day. This information is a critical component of the simulation, for the busyness of the pathways the pods operate

on is a key factor which influences their average speed.

The speed at which an autonomous pod can travel, at different locations across the city, is of paramount importance to the commercial viability of the L-SATS because it can be used to estimate the journey time of a pod travelling between two points in the city centre. The journey time for the L-SATS is one of the quantities that determines whether customers will choose to use the autonomous pods over other forms of transport. The second quantity is the cost of a journey; it is assumed that the pricing mechanism for this transport system is a small fee based on the distance travelled by the customer. This fee is calculated to be competitive with the other forms of transport that would compete with the L-SATS, in the case of Milton Keynes the competition is primarily buses and taxis. The journey time also defines the necessary fleet size since it restricts the number of journeys a pod can complete per day, a large fleet size reduces the profitability of the system.

Figure 2: Photograph of autonomous pods operating in Milton Keynes

L-SATSs are a novel idea and there is little academic analysis relating to their potential impact on urban environments. There is significant literature which utilises ABM to analyse the impact of autonomous vehicles on the public roads. Research has evaluated the implications of shared autonomous vehicle systems and has estimated that these transport systems could reduce the number of vehicles on the road by a factor of 10, but with an 11% increase in journey time [2]. However, this analysis utilised high-level agent-based modelling, treating cities as a grid and journeys as transfers of people from

one square of the grid to another. This analysis does not model the micro interactions between autonomous vehicles and pedestrians which is a technical challenge this project attempts to solve.

There has been significant ABM relating to pedestrian behaviour and there are established ways of moving pedestrians agents spatially in a realistic fashion. Typically, the medium-range and long-range movement for these agents are separated. A series of waypoints break down the overall route into smaller segments, and rule-based algorithms govern agent behaviour between waypoints [3]. This project makes use of a leading crowd simulation software package, MassMotion, as a platform on which to build simulations; this software has sophisticated models of pedestrians that behave realistically in a variety of scenarios. Simulations which model realistic individual interactions between pedestrians and autonomous vehicles are not well-established and so novel work has been undertaken in this project to create these agent models.

1.3 Project Overview

L-SATSs have the potential to be more environmentally friendly than existing transport solutions. The electrification of vehicles reduces greenhouse gas emissions and the pollution in city centres. In addition, connected systems can utilise vehicle resources more efficiently via journey sharing, which has the potential to reduce the distances vehicles travel. Given the sustainable impact of this project, its remit falls within the the Civil Engineering Group at Cambridge University Engineering Department (CUED). However, this project approaches solving the problem through simulation and ABM and so it has a large software engineering component. Since much of the engineering used in this project is not typical to Civil Engineering project reports, much of this work is restricted to the 'Technical approach' section of this report, where it is explained as if the reader has no prior expertise of the software engineering involved.

ABM has been selected as the tool to analyse L-SATSs because it offers a way of realistically simulating the complex dynamics of a geographical system [4]. ABM involves breaking down a system into its individual components (agents) which have their own characteristics and sets of rules; in this project the simulated agents are autonomous vehicles and pedestrians. It is challenging to derive simple and concise sets of rules for these agents since they have complex behaviour. In recent years, it has been advances in artificial intelligence and computer vision which have helped realise autonomous transport systems. The algorithms which underpin these technologies are too complex to fully model in the simulation and so reasonable approximations of these algorithms must be

made for the agents.

The primary aim of this project is to create an accurate simulation of the L-SATS operating in Milton Keynes. This involves designing realistic agents to represent vehicles and pedestrians in the simulation by creating accurate and concise models of their attributes and behaviour. Data of pedestrian movement throughout the city has been integrated into the simulation so the impact of pedestrian traffic on pod journey time can be analysed. This allows the parameters required to make the fleet of pods a profitable taxi service to be evaluated. The simulation also provides a means of analysing some of the protocols autonomous pods should follow when interacting with pedestrians.

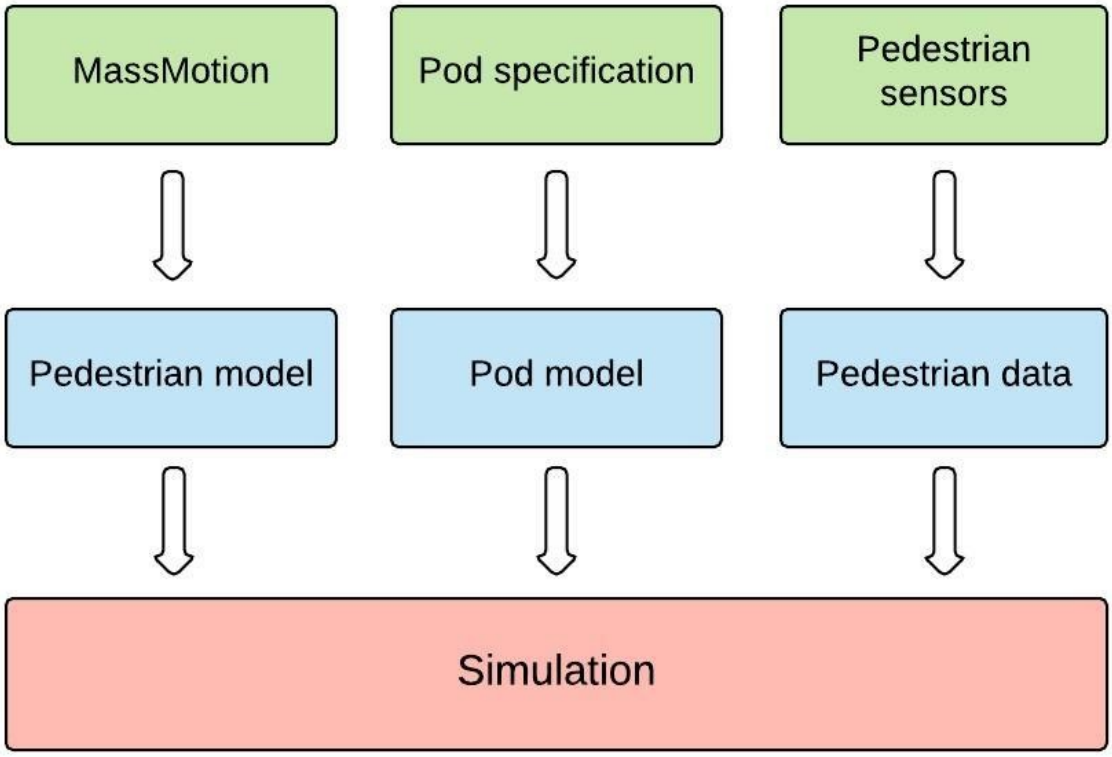

Figure 3: Illustration of the components that make up the simulation

There has been some previous high-level ABM in this scope at CUED which estimated how the average speed of the autonomous pod is influenced by the density of people in a pedestrianized area. This project will build on this work by creating a lower-level agent-based simulation, which models the specific interactions that agents have with each other rather than approximating them. This yields a more accurate and precise simulation at the expense of simplicity and computational efficiency.

2 Technical approach

2.1 Overview

As the UK Autodrive initiative reaches its conclusion in October 2018, a decision will have to be made regarding the future use of the autonomous pods in Milton Keynes city centre. The work undertaken in this project will contribute to this decision, by using agent-based simulation to analyse the performance of the autonomous pods operating in a variety of different scenarios. Preliminary simulations have analysed variables such as the different algorithms that pods should use to navigate pedestrianised paths, and how the width and design of paths influences their average pod speed. These are followed by a large-scale simulation which estimates the journey times for pods travelling between different locations in the city centre.

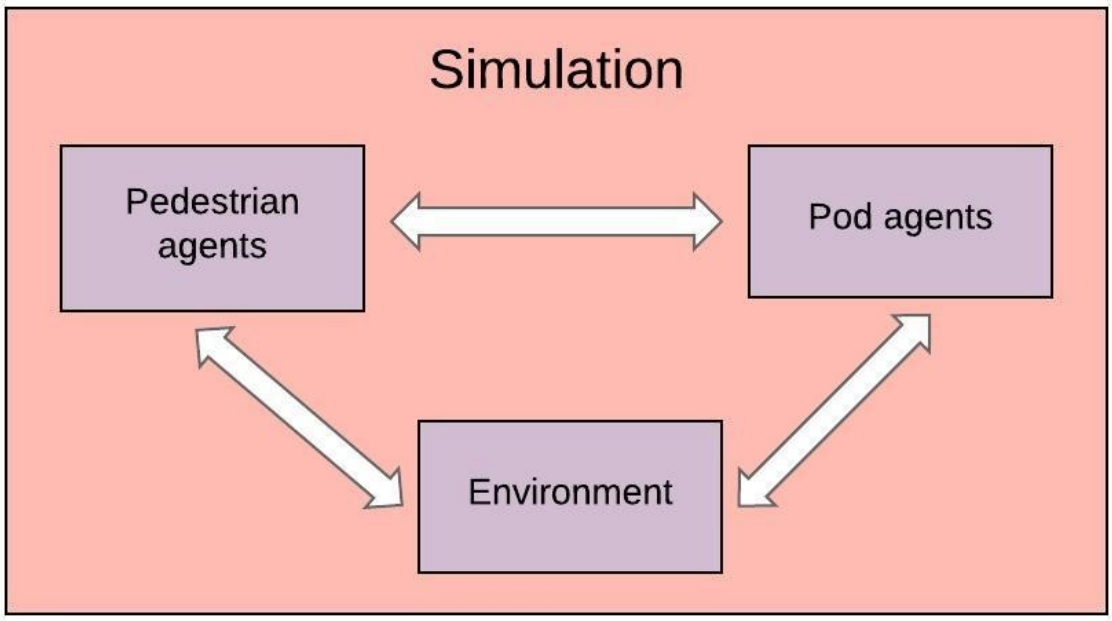

Figure 4: Overview of the agent-based simulation which consists of interactions between pedestrian agents, pod agents and their environment

The fundamental assumption of this project is that a complex system, consisting of pedestrians and autonomous vehicles interacting with each other in an urban environment, can be broken down into a collection of objects. These objects include the different components of the simulation environment and the agents which travel within it. Individual objects are considerably simpler to model than the system as a whole as they are each governed by a simpler set of rules. Given this assumption ABM is an appropriate tech-

nique to simulate the system with because the dynamic objects (the pedestrians and autonomous vehicles) are modelled as individual agents in the simulation.

Object Oriented Programming is a natural framework for ABM because of the way programs can be organised into cooperative collections of objects, with clearly defined rules of how the objects interact with each other. In this instance, examples of objects in the simulation are pedestrians, pods, portals, the floor etc. Portals are the entrance and exit points for agents in the simulation. The floor is a plane that the agents are always required to remain on; for the simulation of Milton Keynes the floor consists of the pedestrianised walkways in the city centre.

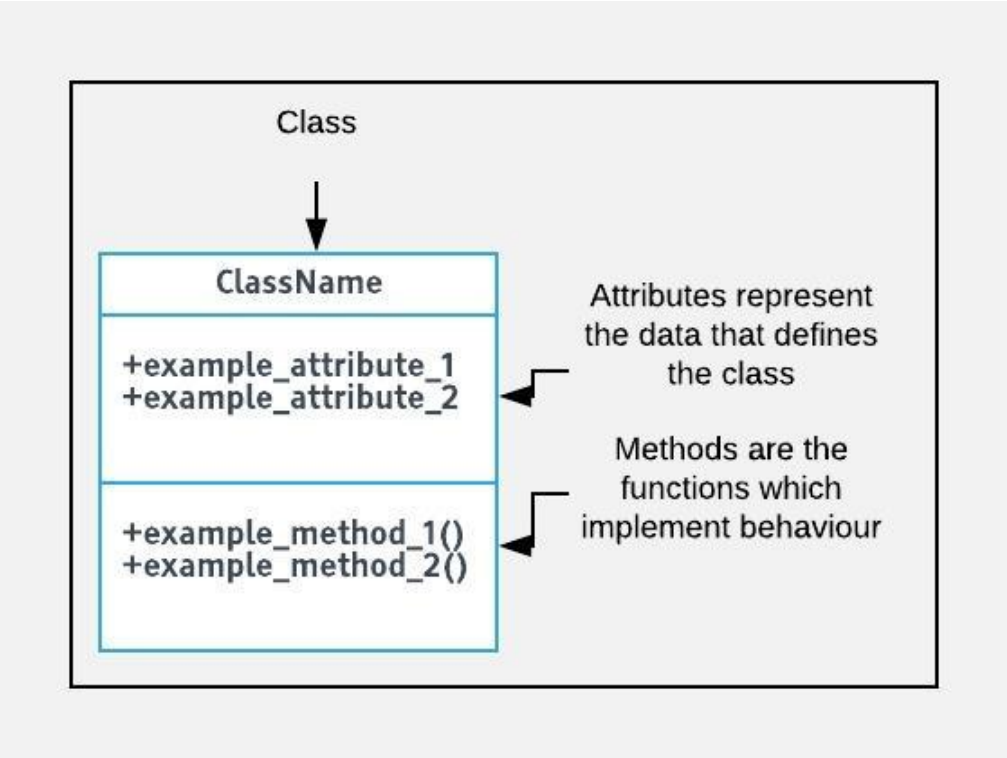

Figure 5: Illustration of class in UML

Classes in Object Oriented Programming are templates for objects; they define the attributes and methods that belong to them. Figure 5 shows an illustration of a class and its associated attributes and methods. Attributes are the data that is required to define an object, for example an autonomous pod object has attributes such as a 'top speed', 'turning radius' and 'acceleration' rate. Methods are functions that implement the behaviour of an object, for example an autonomous pod object has methods such as 'slow down', 'move to waypoint' and 'avoid collision'. Another example of a class is the simple model of a pedestrian. A pedestrian can be approximated as a cylinder and so it has attributes 'height' and 'radius'. It could be able to behave in two different ways, so it

could have two methods ‘move’ and ‘stand still’.

2.2 System design

The design of the system responsible for running simulations can best be expressed using a class diagram, one of the fundamental Unified Modelling Language (UML) diagrams in software engineering. A class diagram is a static representation of a system. The diagram provides abstraction of the overall system, because it expresses it solely in terms of the objects which it is comprised of and the relationships between them. The diagram also encapsulates information about the objects, by expressing all the information about them as attributes and methods, hiding details of their implementation. Figure 6 provides a simplified high-level view of the design of the system; not all classes, methods and attributes are shown in this diagram as it is a simplification of how the system works.

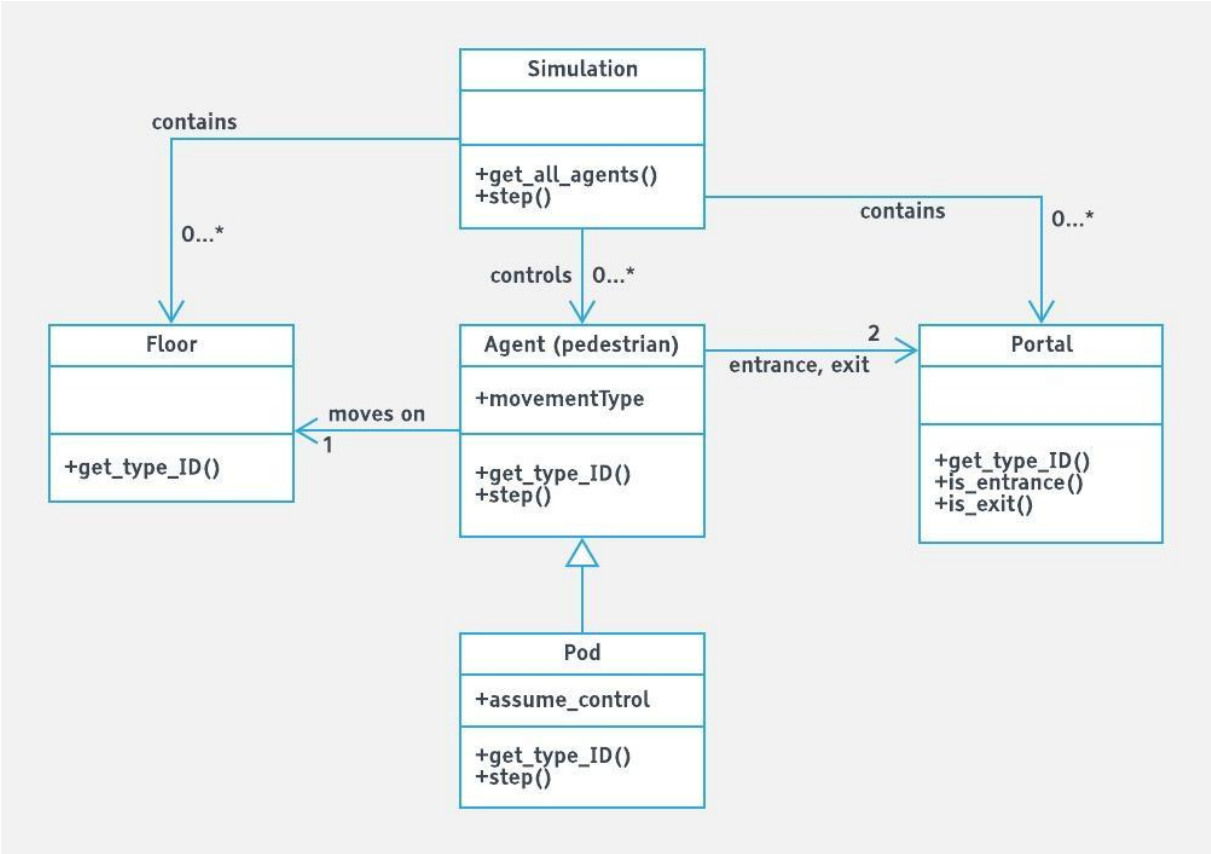

Figure 6: Class diagram for the system that runs the agent-based simulations

The open arrowheads represent relationships between objects. These arrowheads have a description and number associated with them. For example, an agent has a relationship with the floor; the diagram illustrates how the agent must be always linked to one floor

which it can 'move on'. Each agent is also linked to two portals, one to 'enter' the simulation and one to 'exit' it. There is a simulation object which 'contains' a variable number (0...*) of portals and floors and 'controls' a variable number of agents since different simulations can be run in different ways.

The closed arrowhead in Figure 6 denotes inheritance between objects. Inheritance is the process whereby a sub-class (in this instance the pod class) takes on the properties of the super-class (the pedestrian agent class). The pod class inherits from the default agent class which represents pedestrians in MassMotion, meaning it has all the attributes and methods of the pedestrian. This is useful because the pod class inherits methods that allow it to be moved to new positions in the simulation and to check for agents and obstacles around it. It also allows the pod to have an avatar set for it and for the agent to have its colour changed, which are useful properties for visualising the agent in the simulation.

However, the pod class also inherits properties from the pedestrian agent class that are not useful to it. For example, agents in MassMotion have a radius attribute that is used to represent their presence in the simulation as a circle on the floor. MassMotion's social forces algorithm calculates the route between waypoints for agents to follow, which prevents them from colliding with obstacles or other agents [5]. The social forces algorithm is not calibrated for agents which are significantly larger than pedestrians, so the pod agent cannot simply have a radius corresponding to the size of the autonomous vehicle [6]. Instead the radius attribute for the pod is set to zero so it does not interfere with the social forces algorithm and instead new behaviour is added to the pedestrian agent class, so it can interact with the pods. This new behaviour is illustrated in the pedestrian agent class in Figure 7. cite

Figure 6 also illustrates some of the attributes and methods of the different objects in the system. The simulation object has a method step() which advances the simulation by a single frame, of length 0.2s, each time it is called. Every agent in the simulation has its own step() method, and the simulation's method is responsible for calling the corresponding method for each of the agents. Each default pedestrian agent has a movement type attribute that can be set to either standing or walking depending on the action the pedestrian is performing. The agent must also always be assigned to a floor, else it is removed from the simulation. The autonomous pod agent is not part of the default simulation. The pod agent extends the default pedestrian agent through inheritance, meaning it has all the attributes and methods of a pedestrian. The pod agent also has a step() method so it can be controlled by the simulation object and it has the same relationships to the floor and portal agents as the default pedestrian agent.

2.3 Extensions to the default system

Since this project uses MassMotion, as a platform on which to build the simulations between pedestrians and autonomous vehicles, it already contains much of the functionality to model pedestrians and evaluate their journey times. MassMotion allows pedestrians to be created with a set of tunable characteristics, which then travel from their origin to their goal avoiding obstacles in their path and other pedestrians in a realistic fashion. However, these pedestrians are unaware of and unable to interact with autonomous vehicles, so MassMotion's software development kit was used to extend the simulation. It was necessary to add methods to the default agent (pedestrian) class, so they could recognise and interact with autonomous pods and a new class was created to model the autonomous pods themselves. The extensions to the simulation are illustrated in Figure 7 with red font.

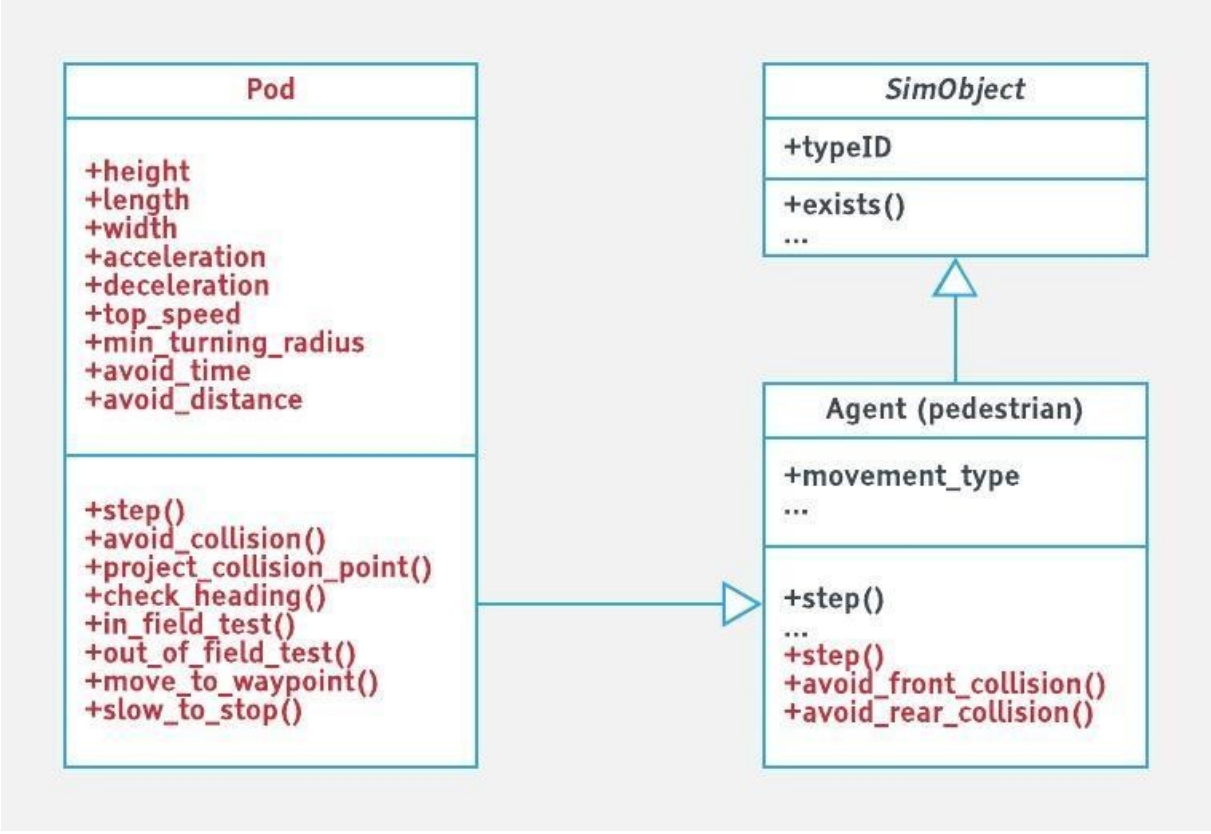

Figure 7: Class diagram illustrating how the default MassMotion system has been extended (additions shown in red font)

The pedestrian agent has an amended step method which implements new behaviour for when the agent is located close to a pod. In this instance, pedestrians must have a way of checking if they will collide with a pod, since the social forces algorithm cannot be used

(as mentioned earlier). Figure 8 illustrates the two regions in which pedestrians check for pods. The angles of these regions are both approximately 60 degrees; the front region represents the agent's line of sight and the rear region correspond to the area where a pod approaching would emit a noise to notify the pedestrian of its presence. The pedestrian's collision avoidance algorithms mimic the in-field test collision avoidance algorithm used by the pod, detailed fully in Figure 10.

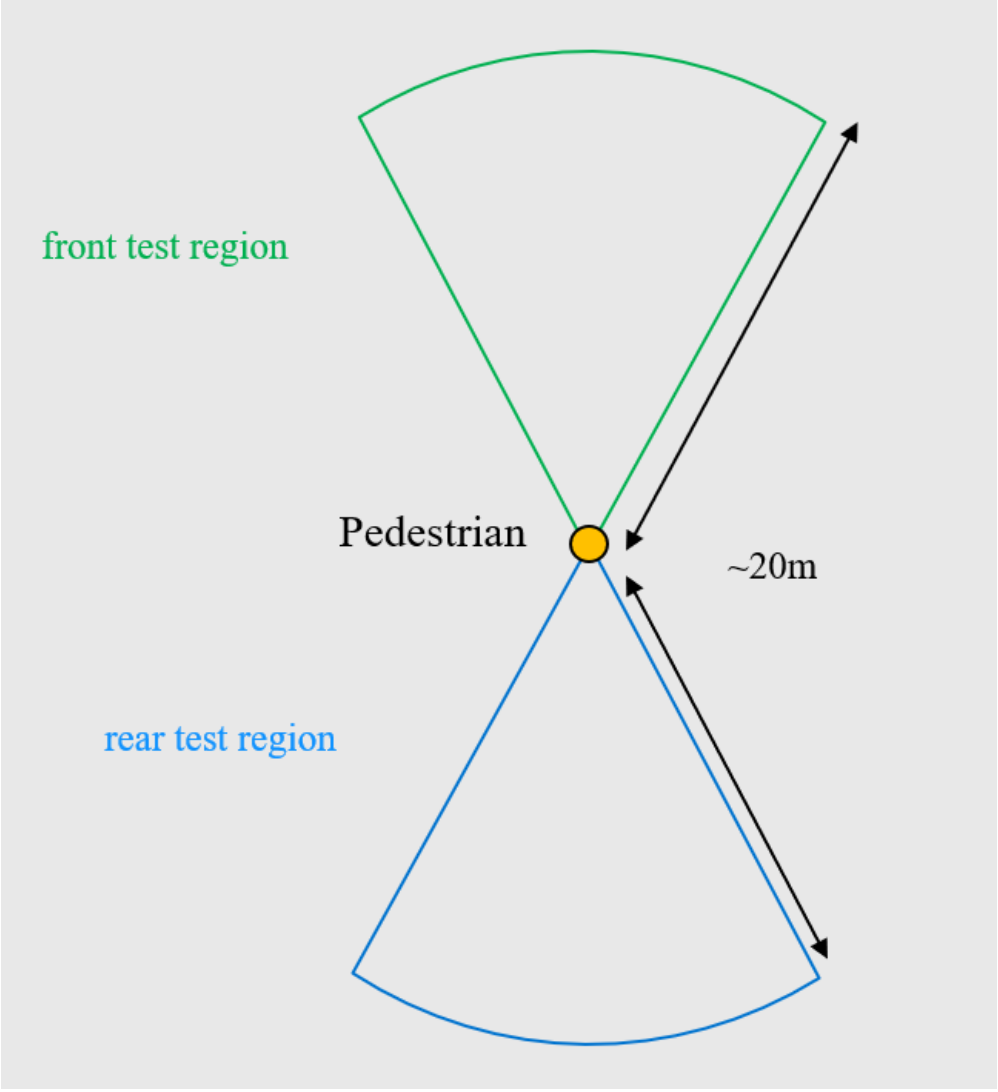

Figure 8: Collision detection regions for the pedestrian agent

Since a pod agent cannot be spatially defined with a radius like a pedestrian agent, because of the calibration of the social forces algorithm, it has been given new attributes for height, width and length. The pod also receives new attributes for its movement characteristics such as a top speed and acceleration. After a discussion with the manufacturer, it was decided that a minimum turning radius was required to be implemented; the turning radius is constrained to increase linearly with speed in a realistic fashion. The avoid

distance attribute is illustrated in Figures 10 and 11; it represents the distance that a waypoint is placed at from an agent, in a direction perpendicular to the agent's heading, to ensure the pod will not collide with it. Similarly, the avoid time attribute refers to the time threshold before a collision when a pod will act to avoid it. This attribute is necessary since when collisions are calculated as occurring far in the future it is not sensible to take evasive action given pedestrians often change their course.

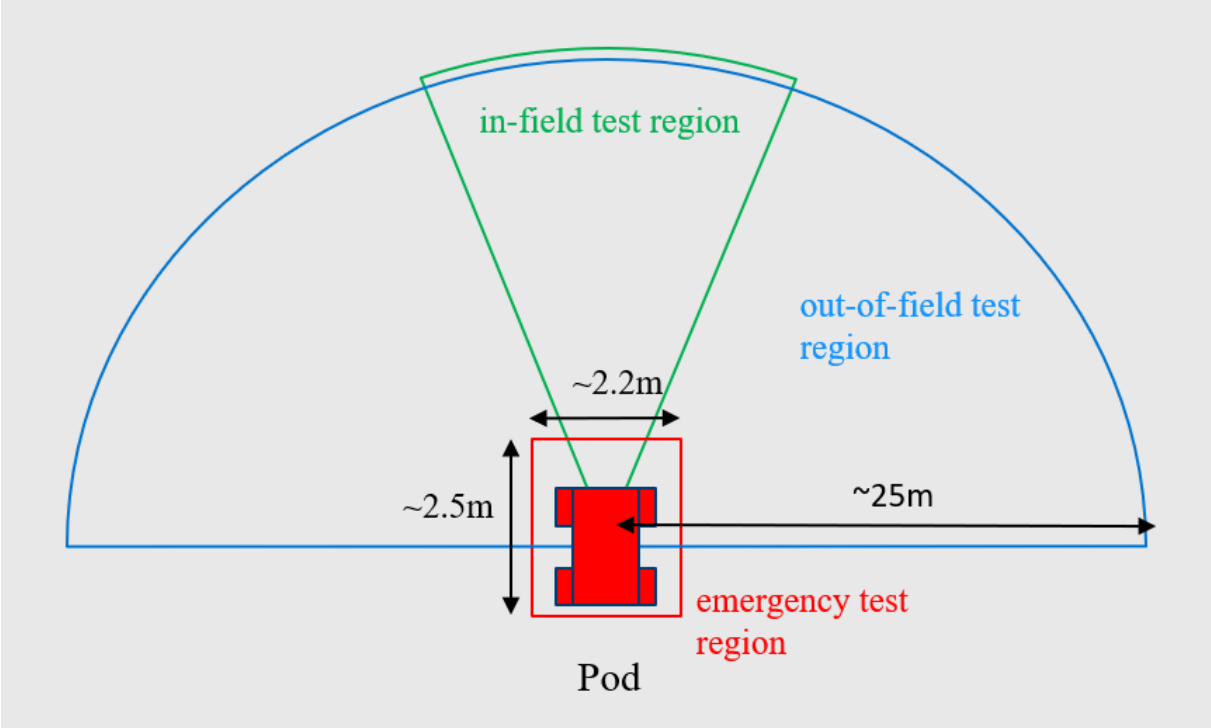

Figure 9: Collision detection regions for the pod agent

The pod agent contains a step method, so its position and velocity can be updated for each frame of the simulation. The pod's other behaviour is implemented within this method. For example, check_heading() implements the turning radius constraint since a pod cannot change direction instantaneously like a pedestrian. The 'move to waypoint' method is used when there are no obstacles or other agents that a pod must avoid; the pod moves in the direction of its next waypoint accelerating to its maximum speed. On the other hand, the 'slow to stop' method is used when a collision cannot be avoided and so a pod's speed is decreased without changing its heading.

Figures 10 and 11 illustrate the different collision detection algorithms used by the pod. The two tests for collisions are an in-field test and an out-of-field test. The different tests are applied to nearby pedestrians which are in the correspondingly labeled regions in Figures 8 and 9. It was necessary to create two distinct collision detection algorithms given pedestrians often do not travel in perfectly straight lines, instead they wander in a

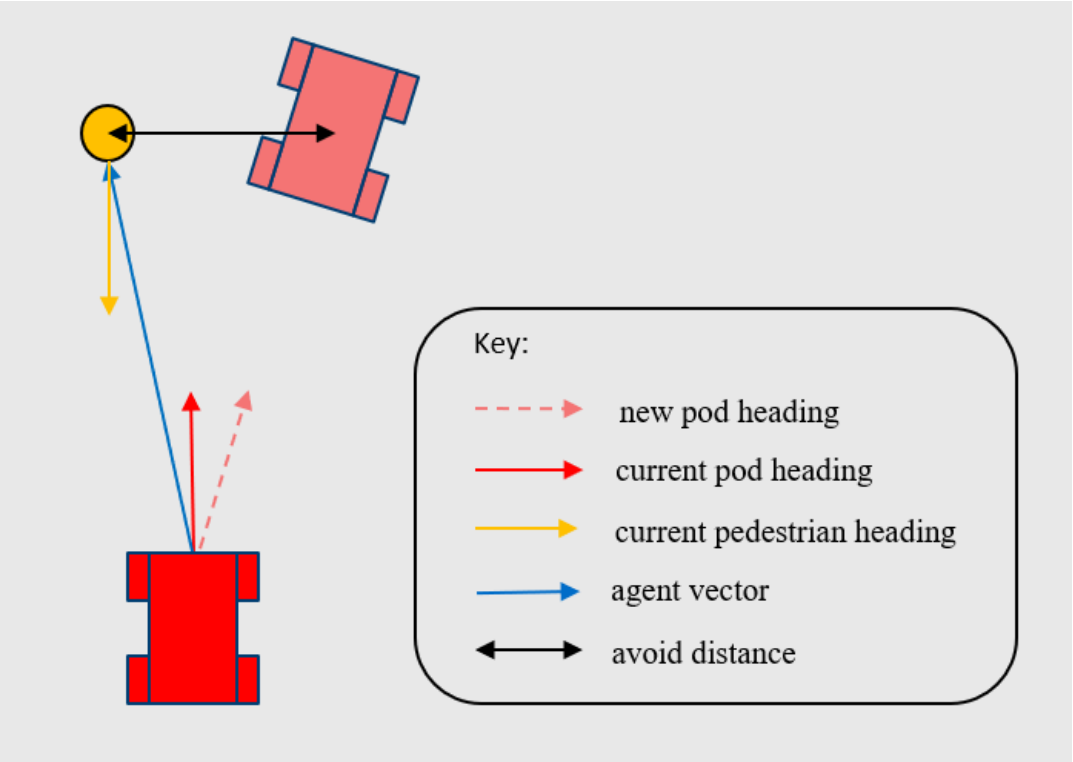

Figure 10: Collision avoidance with an in-field test

random fashion. The in_field_test() method is used to avoid head-on collisions, where the two agents' headings approximately oppose each other. The out_of_field_test() method is more complex; it is used to evaluate collisions that are not head-on. To check for this type of collision the method project_collision_point() is used to estimate the position of a pedestrian once a pod reaches the intersection of the two agent's headings. The project collision method is used to yield the projected pedestrian position shown in Figure 11.

If a collision will occur but there is still time to avoid it, then the pod will make use of its avoid collision method. There are two types of collision to avoid depending on whether the pedestrian is in-field or out-of-field. For the in-field case, the pod is assigned a new intermediate waypoint to travel to, which will prevent a collision with the other agent. This new waypoint is placed a distance specified by the avoid distance attribute away from the agent, in a direction perpendicular to its heading. For the out-of-field collision the logic is the same, but the projected pedestrian position is used rather than the pedestrian's current position. The in-field test is required in addition to the out-of-field test because the out-of-field test is poor at dealing with head on collisions. Pedestrians do not travel in perfectly straight lines and so their projected position, calculated by the out-of-field test, can vary significantly between consecutive frames of the simulation.

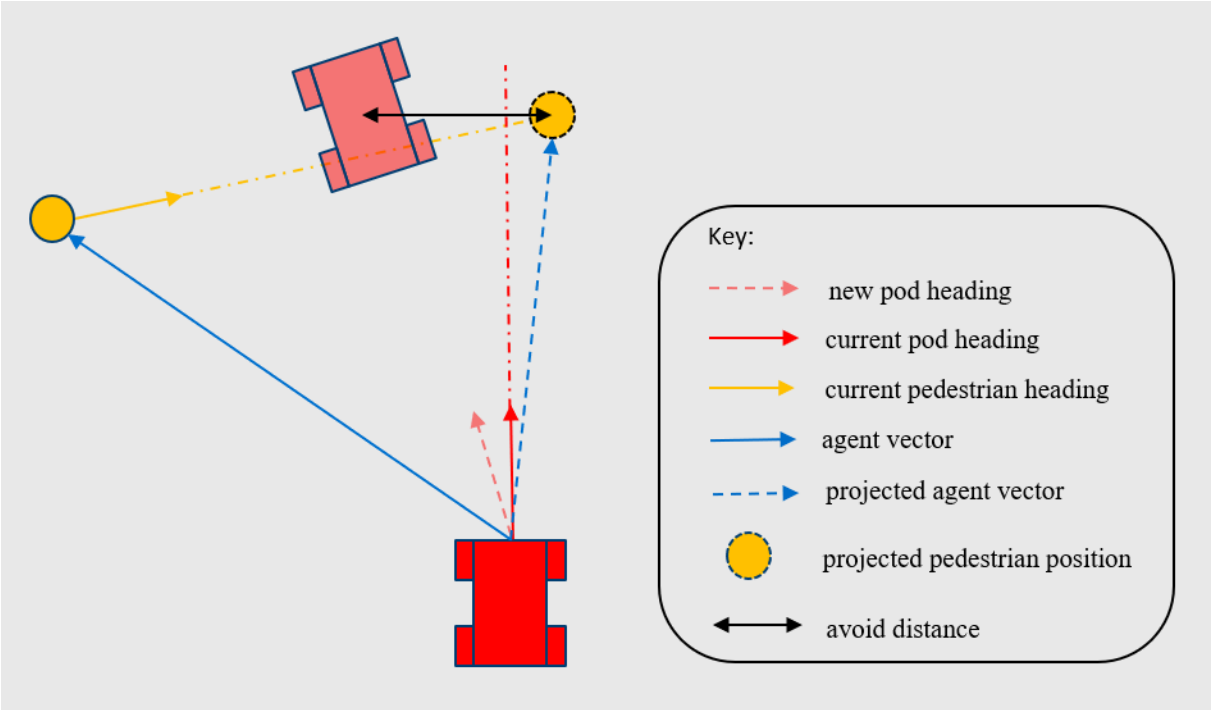

Figure 11: Collision avoidance with an out-of-field test

2.4 Execution of simulation

The sequence diagram, another key UML diagram, expresses how different objects in a system interact with each other sequentially in time. The objects are represented by the coloured boxes at the top of the diagram. The labelled arrows represent the methods called on objects which are ordered vertically to reflect the fact that they are called sequentially in time. The transparent box labelled 'loop' represents how a block of methods are repeatedly executed subject to a condition not being fulfilled. The transparent box labelled 'alternative' illustrates how different methods are executed subject to a condition. The light-grey blocks represent the time period over which methods are executed; blocks which overlap each other represent nested methods.

The sequence diagram depicted in Figure 12 encapsulates the logic implemented in the step method for a pod. The simulation object updates the simulation to the next time frame, which occurs 0.2s after the current frame, by calling the step method of every agent in the simulation. The pod executes its step method and loops through all the pedestrian agents that are nearby to it, analysing their positions and velocities. If a collision is imminent for any of these agents a pod will decide to slow down to prevent the collision from occuring. A pod checks if it needs to slow down using the in-field and out-of-field test discussed earlier and an emergency test. The emergency test checks whether pedestrians are close to touching a pod, by checking if they are present in the emergency test region

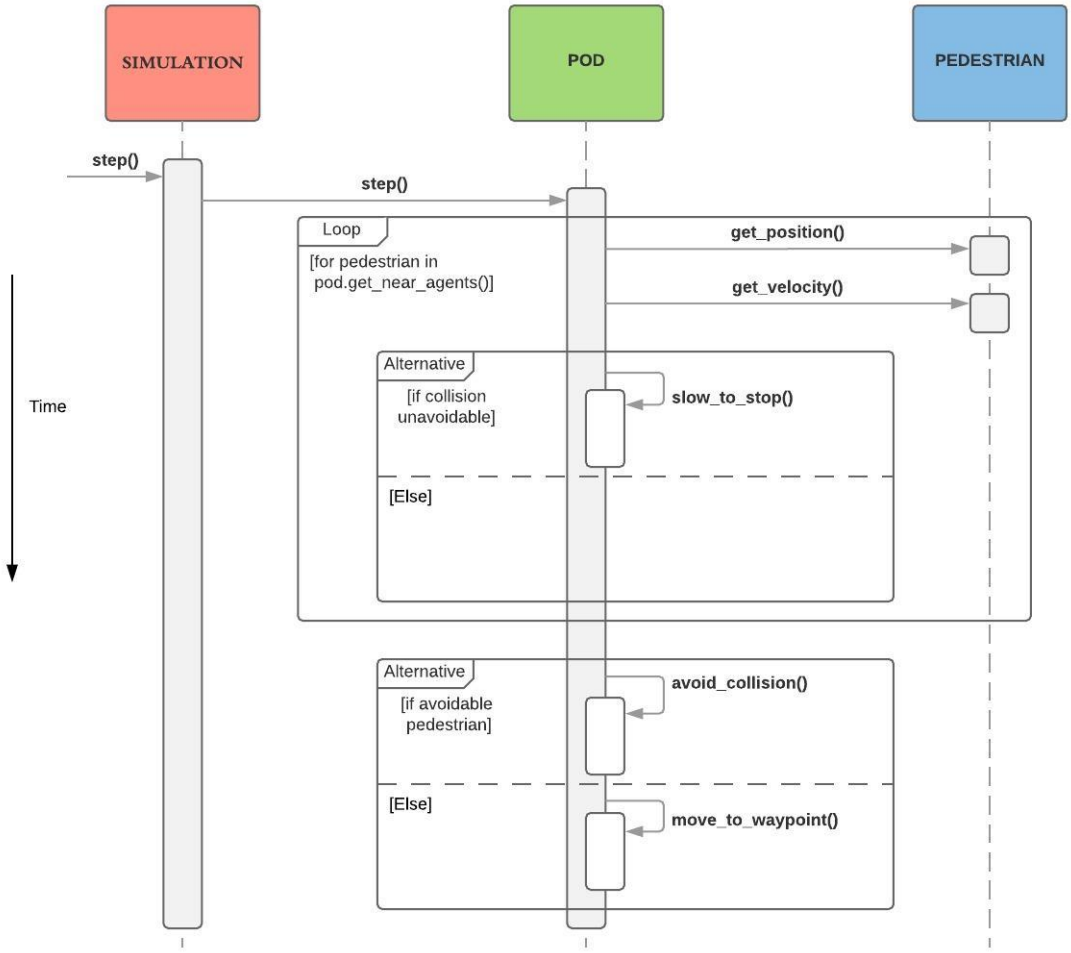

Figure 12: Sequence diagram to illustrate the pod agent control algorithm

shown in Figure 9. If the pod does not need to slow down to prevent a collision, then the alternative statement is executed. This statement ensures that pods heads directly to the next waypoint if there are no pedestrians near to it, else the nearest pedestrian is avoided by the pod. The nearest pedestrian can either by the actual location of a pedestrian using the in-field test or a projected pedestrian position using the out-of-field test.

The second sequence diagram illustrates the execution of a pedestrian agent's step method when interacting with a pod agent. Once again, the simulation step method calls the step method for each agent in the simulation. As with the pod agent, the pedestrian agent loops through nearby agents checking to see if any of them are pods, getting their positions and velocities. If there is no pod located close by, the pedestrian executes its default behaviour. Else if there is set to be a collision with a pod, the pedestrian agent avoids it using one of two distinct collision avoidance algorithms, based on the in-field avoid collision algorithm illustrated in Figure 10.

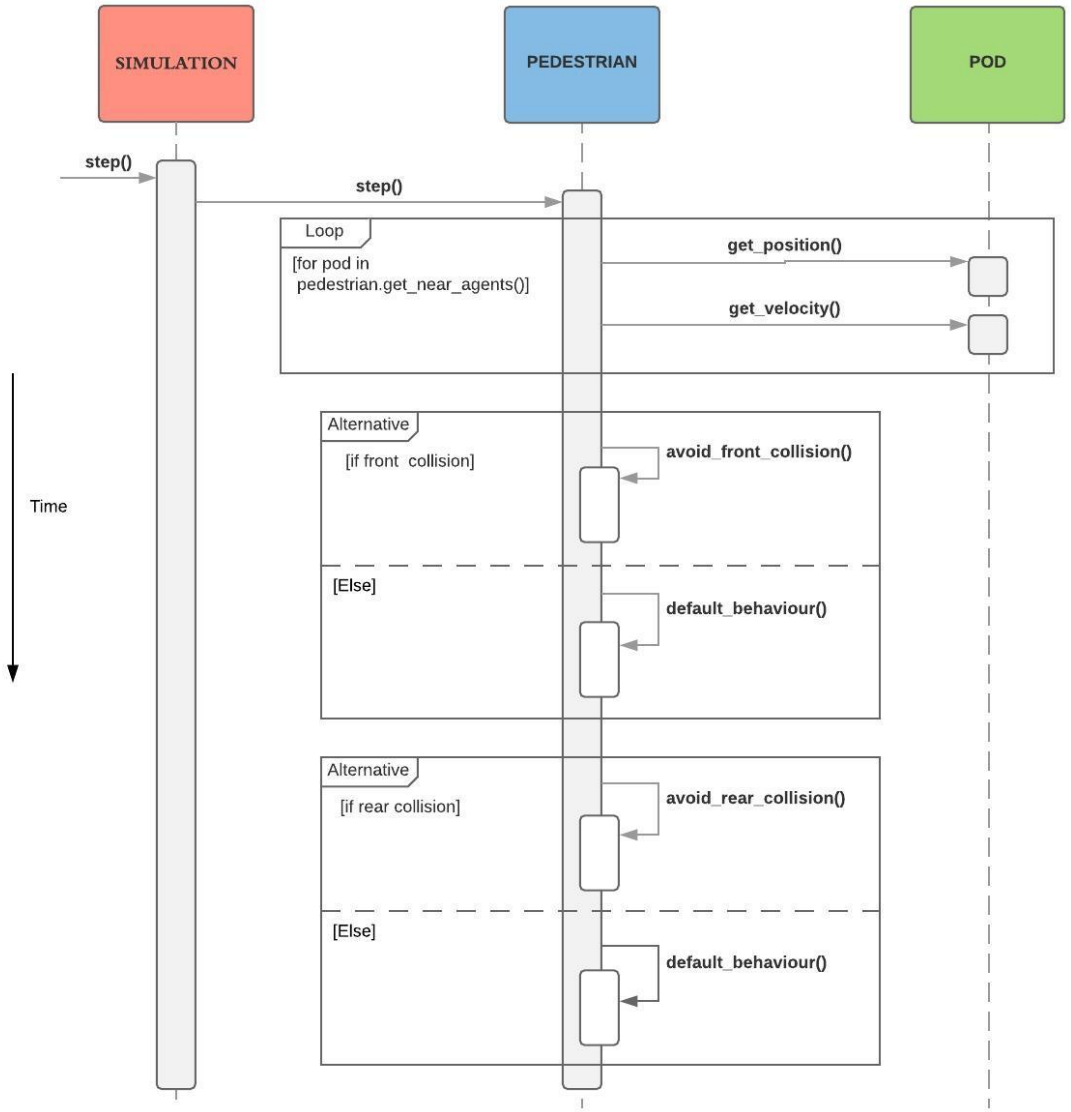

Figure 13: Sequence diagram to illustrate pedestrian control algorithm

2.5 Integration of pedestrian data into simulation

Given that these simulations seek to evaluate autonomous pods operating in Milton Keynes, the flow rates of pedestrians in the simulation should mimic the flow rates in the real world. Sensors installed in Milton Keynes record the number of pedestrians that pass them and the direction in which they travel. This data provides a complete picture of the pedestrian flow profiles in the locations where the sensors are situated but they provide an incomplete picture of the pedestrian flow profile throughout the entire city centre. To attempt to estimate the pedestrian flow profile at all locations that the pods operate in around the city centre, flow rates were interpolated for the pathways with no data associated with them.

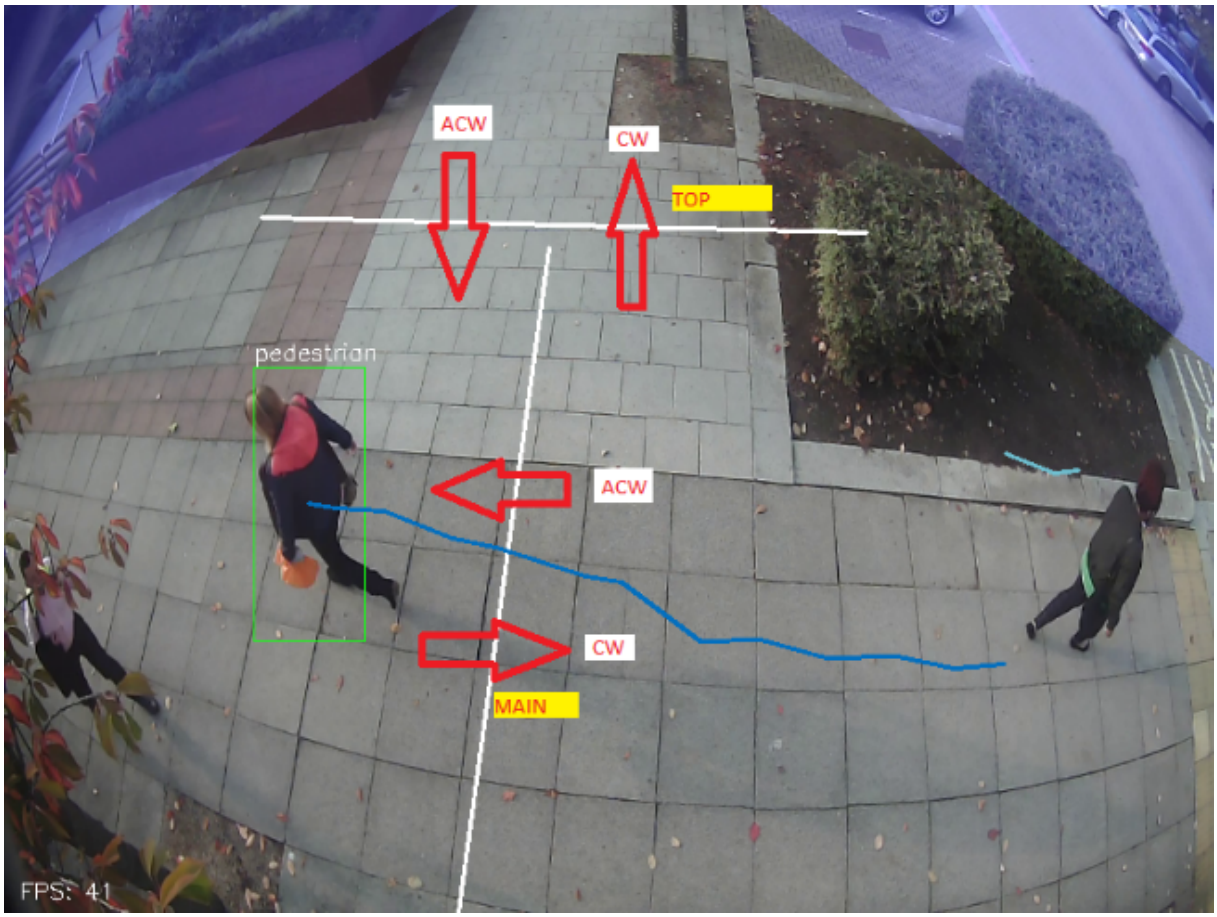

Figure 14: Image from a pedestrian sensor

Figure 14 shows an image taken from one of these pedestrian sensors. The sensor calculates pedestrians travelling clockwise (CW) and anti-clockwise (ACW) through lines across the different paths in the image. The pedestrian sensors were well distributed throughout Milton Keynes city centre, allowing the most busy and quiet pedestrianised areas to be identified. This made interpolation for pathways with no data relatively simple because there are no large areas in the city centre which have no data.

The full details of the hand calculations used to estimate the pedestrian flow rates for areas with no sensors were too lengthy to include in the Appendix for this report. However the Appendix contains a number of 'trip networks' which illustrate the journey times of autonomous pods in Milton Keynes, on a specific type of day and at a specific time of day. These 'trip networks' were developed by integrating the real-world pedestrian data into a series of agent-based simulations. Figure 15 illustrates how the pedestrian data read by a sensor can be compiled into a histogram. These histograms were used to identify the pedestrian flow rates along the pedestrianised walkways in Milton Keynes, so that pod journeys could be simulated for each pathway.

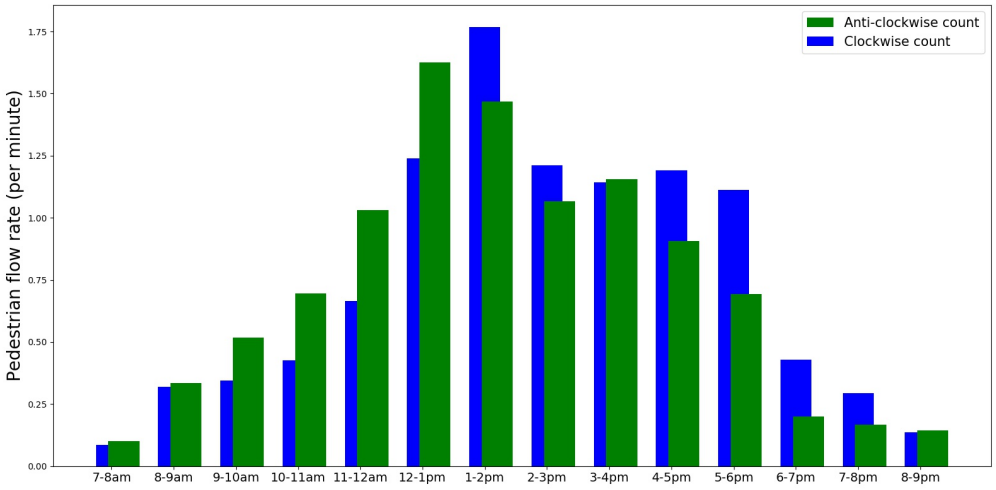

Figure 15: Example data recorded by a pedestrian sensor installed in Milton Keynes

3 Methods

3.1 Overview

The primary aim of this project is to simulate the autonomous pod L-SAT operating throughout Milton Keynes city centre. This involves estimating the journey time for an autonomous pod travelling between any two pair of points in the city centre at specific times of day. However, before undertaking this large-scale simulation, multiple simpler simulations were run to evaluate specific characteristics of the transport system such as the pod's control algorithm and the attributes of the environments they travel in.

The simulation environments were created within MassMotion and were designed to meet the requirements of the environments the autonomous pods will operate in in the real world. There are two main environments that autonomous pods would be expected to encounter in Milton Keynes. The first environment is a narrow pathway with a typical width of 3m and length of the order of 150m. These pathways are designed to allow pedestrians and cyclists to travel in the city centre without having to share the roads with cars. Since MassMotion does not model cyclists these experiments only consider pedestrians travelling along these walkways. The development of a cyclist agent class could be considered for future work but is beyond the scope of this project.

The second main environment is an open square which contains numerous pedestrians and obstacles. There are a much smaller number of these environments than the narrow

pathways; the open square environments are typically found around the train station and shopping centre in Milton Keynes. Obstacles pose a different challenge to pedestrians for the autonomous pod agents and so experiments were run to identify a control algorithm that could satisfactorily navigate both kinds of objects. A simulation environment of the entirety of Milton Keynes city centre was also created. It turned out to be too computationally expensive to run experiments in this very large environment however it is a useful tool for visualising the L-SATS operating as it would in practice. This environment was created using Sketchup Placemaker, a tool that converts maps into 3D models, and then simply imported into MassMotion.

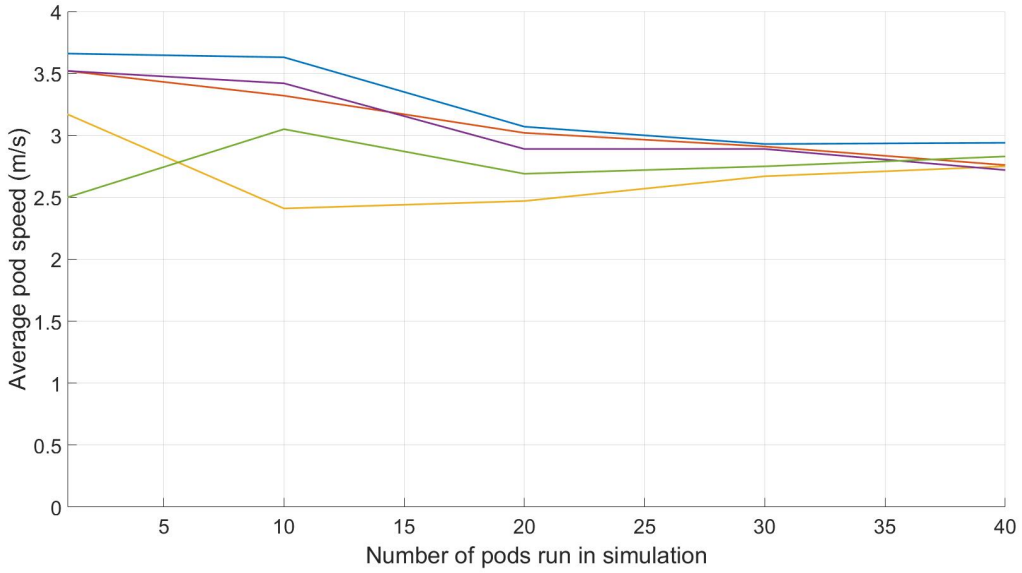

Figure 16: The influence of the number of pods run in each simulation on average pod speed for different random seeds

For each simulation a threshold number of pods is required to be run in order to produce consistent results. This is important because pod speeds are highly sensitive to the number and spacing of pedestrians in the simulation. Pedestrians are pseudo randomly distributed in MassMotion with a random seed determining their distribution. The larger the number of pods run in each simulation the more consistent the results. However running a large number of pods in a single simulation is computationally expensive. Figure 16 demonstrates how increasing the number of pods that are run reduces the variance in the average pod speed calculated in the simulation. It was decided that 40 pods per simulation would be a sufficient threshold number, since this yields consistent results accurate to approximately ± 0.1 m/s. This amounts to one hour of pods travelling down busy pathways in the simulation in real time, which would be a realistic test in the real world.

3.2 Simple vs complex pod control

The specification for the autonomous pods is yet to be finalised and so there is still uncertainty relating to how the pod should interact with pedestrians. In general, there is a trade-off between having a simple pod that is cheap to implement vs a complex and expensive pod with can interact in a more sophisticated manner with pedestrians. These competing solutions are approximated with simple and complex pod specifications in this first simulation.

Simple pod specification:

- Cannot avoid pedestrians.
- Can only detect pedestrians in its line of sight (a 60-degree circle segment of radius 22m).
- Slows down to a speed slightly greater than walking pace (2m/s) when a pedestrian is within its line of sight but is more than 6m away from it.
- Stops when a pedestrian is less than 6m in front of it.

Complex pod specification:

- Can estimate collision points with pedestrians.
- Has a line of sight which is a semi-circle of radius 22m.
- Only slows down when collisions are unavoidable at the current speed.
- If there is a group of pedestrians in its way it attempts to avoid the pedestrian closest to it.

To test both specifications, 40 pods were run in the path environment in MassMotion, which is a rectangle of length 150m and width 3m, and their average speed was then calculated. Complex pod agents were free to move across the path in a manner that minimised their journey time by avoiding collisions with other agents. However since the simple pod agents are unable to avoid pedestrians they travelled in a straight line along the walkway, leaving room for pedestrians to pass them either side. This was repeated for different pedestrian flow rates in the simulation, ranging from 0 to 5 pedestrians per minute, which were observed as being approximately the quietest and busiest flow rates experienced by the walkways in the real world. Pedestrians were uniformly randomly

distributed in the simulation and travelled along the path equally in both directions. It is also assumed that 5% of pedestrians will refuse to make way for pods and will only avoid collisions at the last possible moment, accounting for rebellious behaviour.

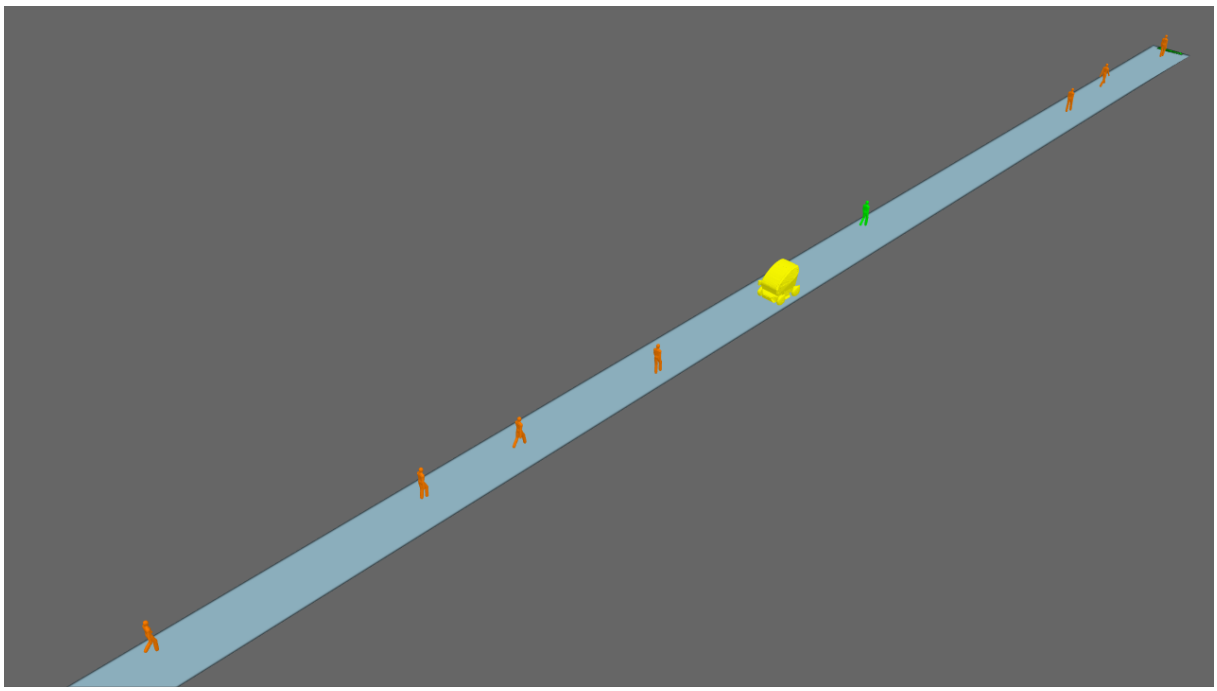

Figure 17: Illustration of a pedestrianised walkway path test in MassMotion

3.3 The split lane arrangement

An important preliminary experiment in this project was to analyse the effect of dividing the path into separate lanes on average pod speed. It was postulated that running a pod in its own lane could boost its average speed, because pedestrians would no longer be in its way so it would not have to slow down as much. Figure 18 illustrates this split lane arrangement whereby a pod travels in a 2.2m wide lane, which is approximately its width, and pedestrians are encouraged to stay within their own 0.8m wide lane.

The performance of a complex pod agent operating on a path with no lanes was compared against its performance on a path with the split lane arrangement. However since the width of the pod's designated lane is approximately the width of the pod there is no space for the pod agent to avoid collisions with pedestrians. In the case of the split lane arrangement the pod agent's collision avoidance behaviour was de-activated, if the agent calculates it is collide with a pedestrian then it can only slow down to a stop to prevent this.

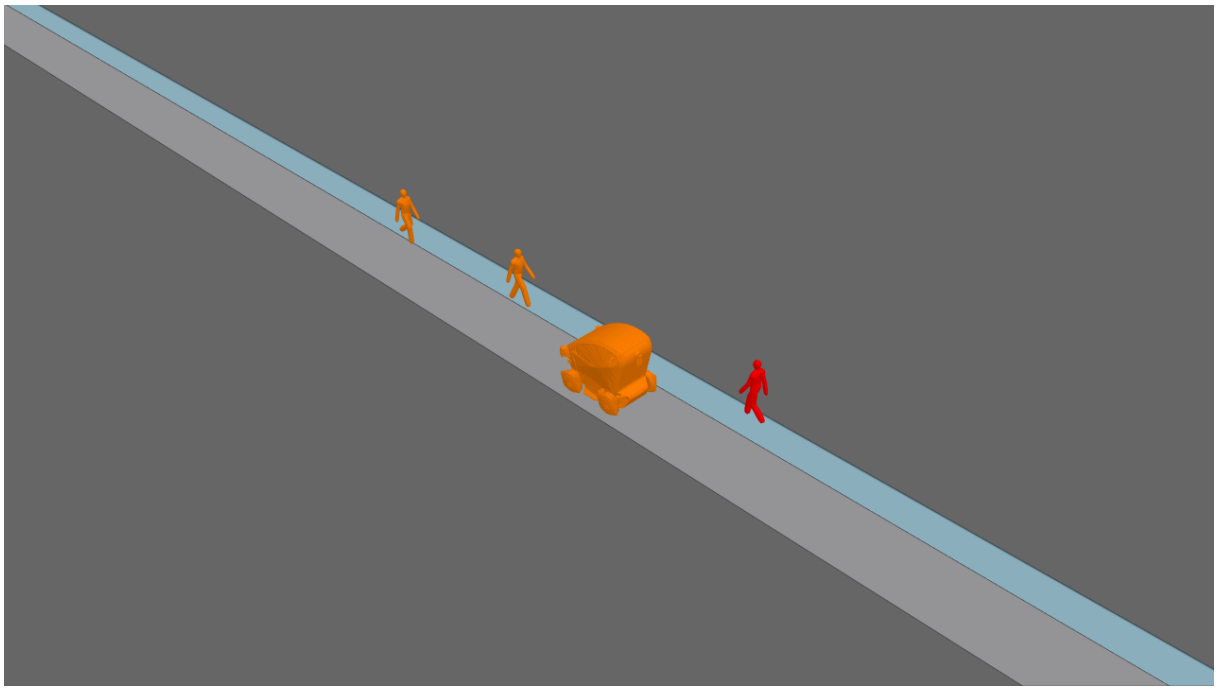

Figure 18: Illustration of the split lane arrangement

In order to account for realistic pedestrian behaviour it was assumed that 20% of pedestrians would wander into the pod's designated lane for the split lane arrangement. Even if pedestrians are encouraged to stay out of this lane, it was considered that it would be likely that a significant number of people would choose to disobey these instructions. Additionally pedestrians which are attempting to overtake each other will also walk into the pod's designated lane. It was assumed that pedestrians would assume they have right of way over a pod.

3.4 The influence of path width on average pod speed

A key parameter that governs the average pod speed along a path is the width of the path. A series of simulations were run to evaluate how path width influences average speed for a variety of pedestrian flow rates. The range of flow rates was selected to be typical to those experienced by the pathways in Milton Keynes, ranging from 0 to 5 pedestrians per minute. As discussed earlier, 40 pods were run for the selected flow rate in each of these simulations. The minimum path width was chosen to be as small as possible whilst still allowing pedestrians and pods to pass each other. The maximum path width was selected to be half a metre wider than the standard path width, because this was considered to be a reasonable amount by which they could be extended.

The typical width of the pedestrianised pathways in Milton Keynes is 3m, however this

can vary significantly. In some areas the pathways have bottlenecks due to constraints from the neighbouring buildings and roads. In addition obstacles such as lampposts, trees and bins in the path effectively shrink the width of the paths in certain places. since there is the possibility of widening the pathways in some places, these simulations set out to provide a means of evaluating if undertaking this work would be a worthwhile investment.

3.5 Cocurrent vs counter flow

Cocurrent flow refers to when pedestrians travel in the same direction as pods along paths. Counter flow is the opposite of cocurrent flow and occurs when pedestrians travel along the pathway in the opposite direction to the pod. It was observed that, for the actual pathways in Milton Keynes, these two flow rates are often not balanced. For example at rush hour in the morning there are large crowds of people travelling from the train station to the business parks. This results in pathways which, at that specific moment in time, contain pedestrians almost exclusively travelling in one direction.

The effect of cocurrent and counter flow on average pod speed is of interest to this project. A simulation was designed to evaluate how different proportions of cocurrent and counter flow affect pod speed. The overall pedestrian flow rate was kept constant, whilst the proportion of counter flow along a path was steadily increased from 0% to 100%.

3.6 Obstacle avoidance

Milton Keynes has several large, open pedestrianised areas in which pods can operate. However, these areas typically contain a variety of obstacles which pods must interact with. These obstacles are expected to reduce a pod's average speed because they must navigate around them, as well as possibly slowing down to avoid collisions with them. Simulations were conducted to evaluate if a pod control algorithm could be created that could navigate these environments realistically. These large obstacles cannot be avoided by the normal collision detection algorithms employed to avoid pedestrians, since obstacles such as bicycle sheds are far larger than pedestrians.

MassMotion's social forces algorithm handles both obstacle and pedestrian interactions for pedestrian agents. The social forces algorithm considers the geometry of the floor, the presence of other agents on it and the whereabouts of obstacles to create a route for the pedestrians to navigate to their next waypoint. This experiment attempts to use the social forces algorithm to allow the pod agent to avoid obstacles. Although this algorithm is calibrated for pedestrian agents it is combined with the pod agent's existing behaviour

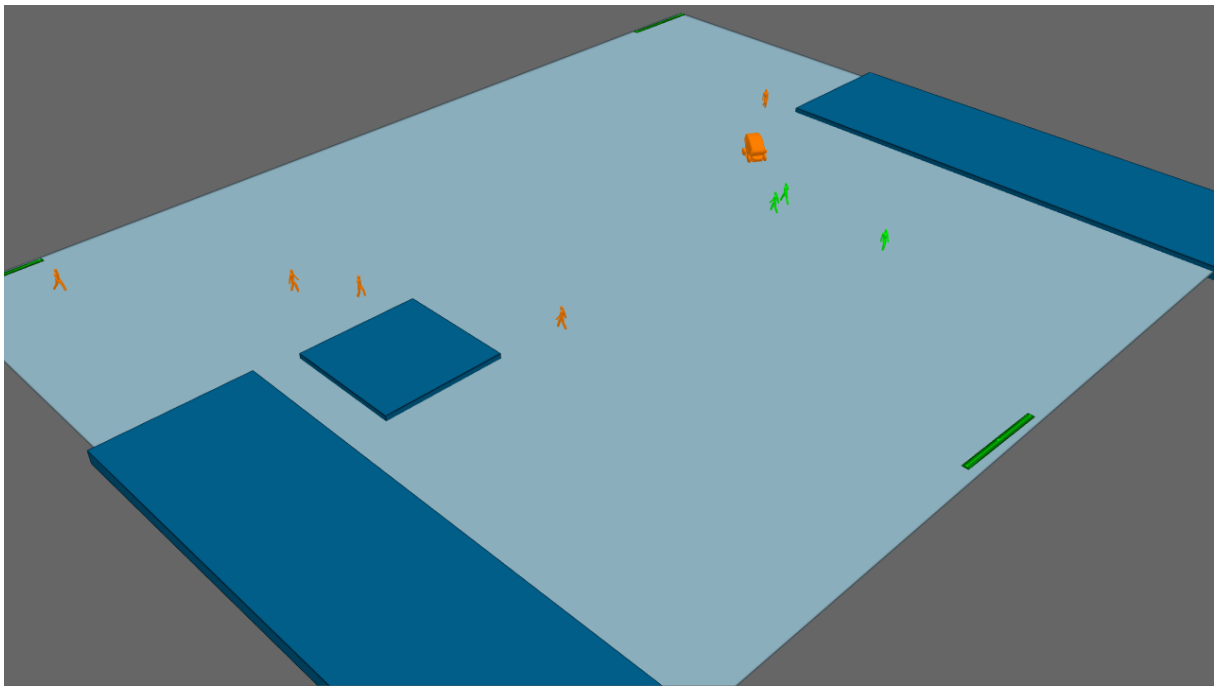

Figure 19: Simulation of pod agents avoiding obstacles in an open environment

in a way that allows it interact with both obstacles and pedestrians. Creating a unique social forces algorithm for the pod would be greatly time consuming and is not easily facilitated by MassMotion's software development kit.

The aim of this test is to evaluate how realistically a pod can navigate obstacles and pedestrians in a large, open space. Although this assessment is primarily qualitative, there are some quantitative measures that can also be used. For example, any collisions with other agents or obstacles results in an unacceptable simulation of a pod's journey. The qualitative assessment is based on a judgment if a pod traveled to its goal in a realistic fashion. For example it is unrealistic that pods can pass close to pedestrians travelling at its maximum speed, even if it doesn't collide with them, because this would be considered unsafe in real life.

3.7 Large-scale simulation of the city centre

The most important simulation evaluated in this project is the large-scale simulation of Milton Keynes city centre. It was found to be too computationally expensive to conduct this as single simulation run in MassMotion. This is primarily due to the fact that the city centre covers a very large area and contains a considerable number of pedestrians; simulating all these agents simultaneously takes significant time. In addition, this project makes use of MassMotion's software development kit, which is optimised for ease of use

not computational efficiency, to extend the simulation using a high-level programming language which is not efficient to run.

There can be tens of thousands of people within Milton Keynes city centre on a busy day. Although MassMotion is handled to model very large crowds, often in models of stations and skyscrapers, this is a computationally expensive simulation without access to high performance computing resources. This simulation requires 40 pods to be run when the simulation is at a steady state. A pod journey to and from the furthest points in the city centre could be several kilometres long. First pedestrians must have travelled this distance at walking pace to allow the simulation to have reached a steady state and then 40 pods must also travel this distance. This example illustrates how the time to run this simulation starts to become unacceptably large.

These limitations resulted in the decision to break down the city scale simulation into multiple components, which could be simulated individually, and then later combined. Milton Keynes city centre was broken down into the individual pathways and open spaces that it is comprised of. Pods were simulated travelling in each individual environment, with the correct number of pedestrians also travelling in the environment. The overall journey time of a pod could then be calculated by summing the journey times of the environments that it travelled in. The results of the large-scale simulation were summarised by several 'trip networks'. These are a set of graphical representations of the paths that autonomous pods can travel along, expressing the journey time for each path for the specific day and time of day that the 'trip network' corresponds to. These 'trip networks' are shown in the Appendix.

4 Results and discussion

4.1 Simple vs complex pod

As was to be expected the complex pod, with more sophisticated control algorithms than the simple pod, has the superior performance and travels faster in pedestrianised areas. The greatest difference in average speed occurs for intermediate pedestrian flow rates of approximately three per minute. When the paths are empty the pods travel at the same speed because they have the same top speed. When the paths are very busy, the difference between the average speeds of the pods reduces, because the complex pod is not able to avoid collisions effectively. However, when the paths are moderately busy the complex pod travels significantly faster than the simple pod. In this instance the pod is able to

take evasive action, to avoid collisions with pedestrians, enabling it to maintain a higher speed than the simple pod.

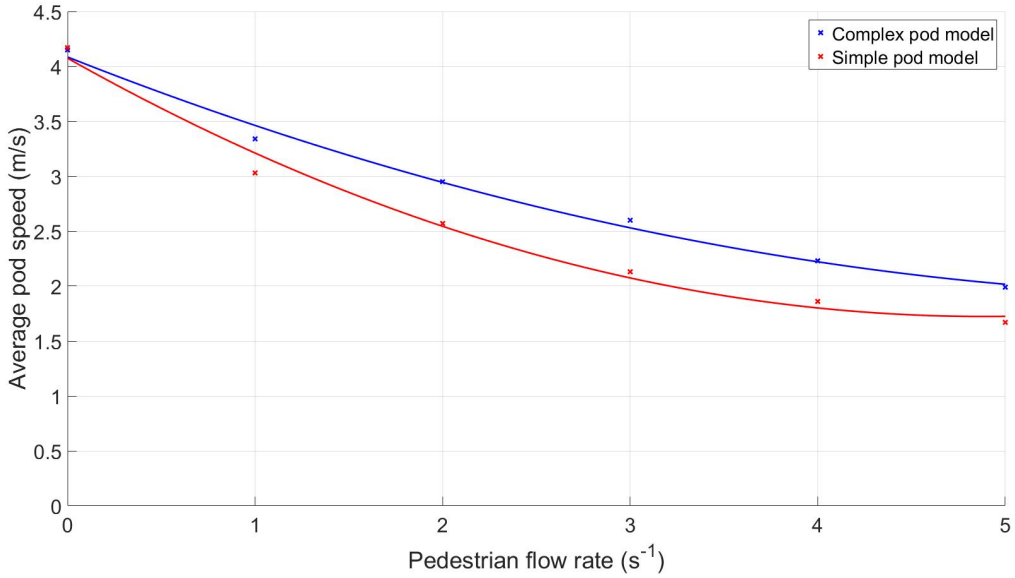

Figure 20: How the performances of the complex and simple pod models vary with the busyness of a path

Most of the pathways in Milton Keynes were measured as having pedestrian flow rates between 0.5 and 2 pedestrians per minute, at peak times on the most heavily used routes this can rise to 4 to 6 pedestrians per minute as shown in Figure 21. This experiment is useful for expressing the performance trade-off for the two designs of pod at a range of different pedestrian flow rates. Complex autonomous pods with sophisticated pedestrian avoidance algorithms require more time and resources to develop. Once an estimation for the difference in cost of the two pods has been made then a decision can be reached regarding which design would be the most profitable.

It was assumed that the complex pod would prove to be the superior design in practice. The complex pod has few additional hardware requirements and is not expected to be significantly more difficult to implement. This assumption lead to the large-scale simulation, undertaken in this project, simulating complex pods rather than simple ones. However this was still an important simulation to run, since as the autonomous pods are tested in Milton Keynes may have to follow a simple pod specification during this testing period, until engineers are confident of their safety.

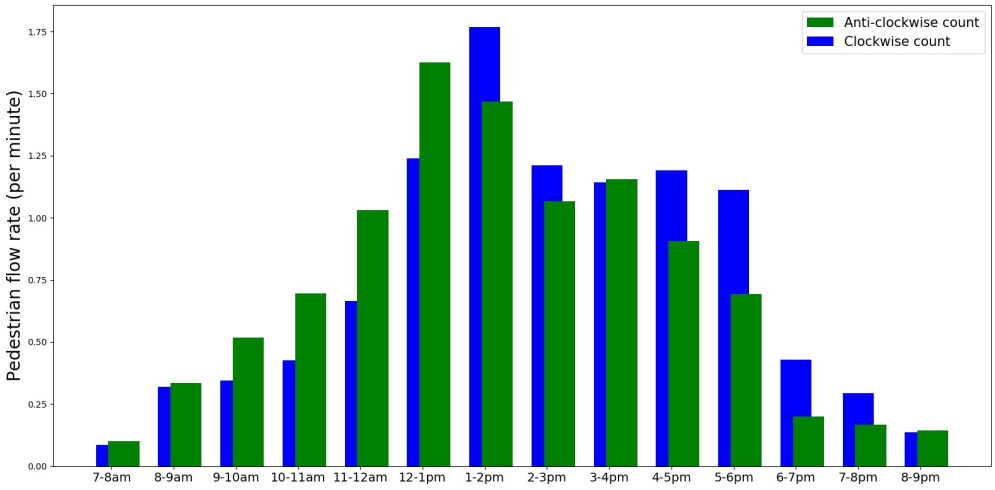

Figure 21: Pedestrian flow rate data for a principal pedestrianised pathway in Milton Keynes

4.2 The split lane arrangement

It was hypothesised that splitting the path into two lanes, one for autonomous pods and one for pedestrians, could increase the average speed of the pods since it encourages pedestrians to stay out of their way. However, it was interesting to note that experimental results disagreed with this hypothesis. When designing the pod agent model, a rule was created for a pod to treat the edge of the floor as a collision point, so that when the pod reaches the end of the pathway it slows down to prevent it running off the floor. This rule also allows the pod to transfer to a new pathway, if one follows the pathway it has just traversed, since it requires the pod to slow down to a speed where it has a small turning radius and so it can change direction to travel along the new path without running off the floor. This was rule was created independently of the split lane arrangement test, but it is the rule responsible for the pods unexpected behaviour in this simulation.

In MassMotion when a pod evaluates the space surrounding it, the space is analysed as a circle comprised of 32 segments [5]. The radii of the different segments represent the distance from the pods position to the nearest agent or obstacle along the vector corresponding to that segment. This discrete approximation results in the fact that a pod's heading never points directly to its next waypoint. Hence when a pod travels along a narrow pathway, the edges of the floor parallel to its direction of travel are evaluated as obstacles since the pods heading intersects them. The pods maximum speed is then restricted because its collision avoidance algorithms are activated. The narrower the pathway the greater the reduction in the pods top speed due to the presence of this rule.

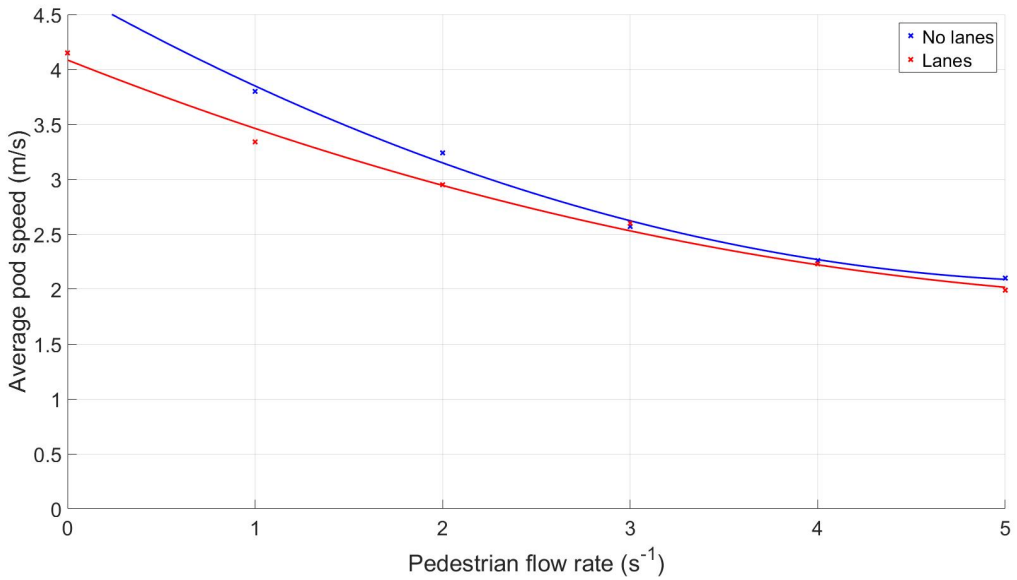

Figure 22: A comparison of a pod's performance on pathways with and without a split lane arrangement

It was decided to keep this rule for the pod's behaviour because it is realistic to expect that pods travelling in narrower lanes should have a reduced top speed. As a pod travels it must constantly make small corrections to its heading to stay in the middle of the lane. There is less room for error in a narrow lane, so it is reasonable to assume a pod should travel slower. This effect is responsible for the reduced top speed of the pod for the dual lane pathway design.

It was expected that Figure 22 would show that when the path gets busy the benefits of the split lane arrangement would become clear. By restricting pedestrians to their own lane, it was thought that the pod would be able to maintain a higher average speed than if there were no lanes. However, it was observed that since pedestrians are randomly distributed and have a variety of walking speeds they are prone to clustering on the pathways; forming small groups and travelling together. This occurs since pedestrians walk side by side if they are walking together or if one is overtaking another. This results in pedestrians straying into the designated pod lane and nullifying any advantages to the split lane arrangement. This experiment suggests there is no benefit to designing the pathway with separate lanes, since there are no pedestrian flow rates for which it outperforms the standard configuration.

4.3 The influence of path width on average pod speed

The simulations evaluating how path width influences pod speed found that the relationship was approximately linear for a variety of different pedestrian flow rates. As was to be expected, average pod speed increased with path width. When pods had more space available to avoid collisions, they were able to pass pedestrians at greater speeds, since they could pass pedestrians with greater space separating them from each other. It was also interesting to observe that the gradients of the different lines plotted for different flow rates are nearly identical (apart from the line corresponding to a zero-flow rate of pedestrians). This highlights how the gain in pod speed due to an increase in path width is approximately the same for all pedestrian flow rates, apart from when the walkways have negligible flow rates.

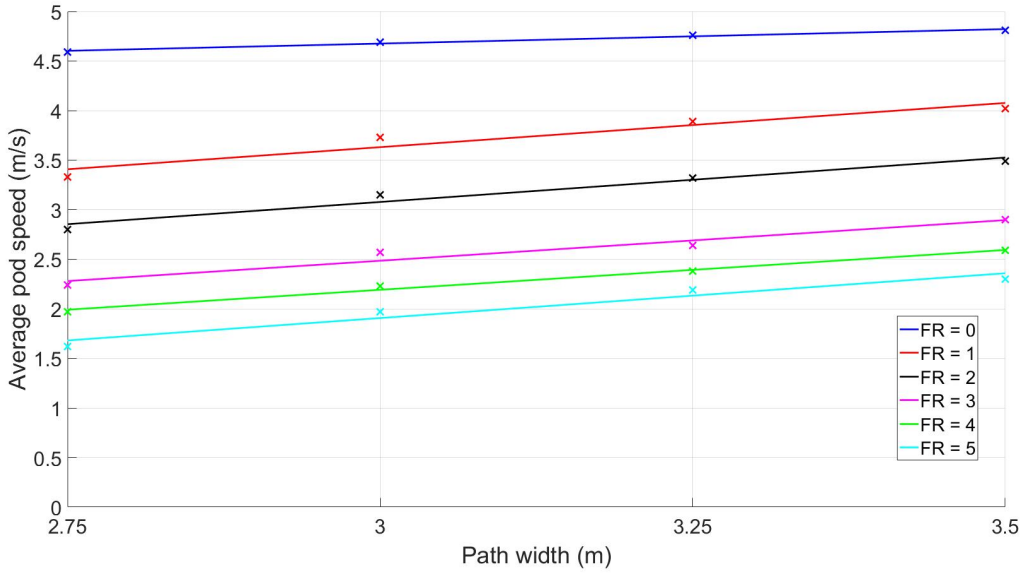

Figure 23: The influence of path width on pod speed for a variety of different flow rates

In the case of negligible pedestrian flow rates the pod travels slower along narrower walkways because of the effect of path width on the pod's collision avoidance algorithm. As mentioned earlier, a narrow pathway reduces a pod's top speed because at high speeds the pod sees the edges of the pathway parallel to its direction of travel as obstacles. This causes a reduction in average pod speed on narrow pathways even when there are no pedestrians sharing them with pods. However this reduction in speed is smaller than for the cases when pedestrians populate these paths. For these cases an increase in path width of 0.75m yields a gain of approximately 0.5m/s. Augmenting pathways is a time-consuming and expensive task and results of this experiment suggest it is not worth undertaking.

4.4 Cocurrent vs counter flow

Experiments into the effect of the ratio of cocurrent flow to counter flow on average pod speed found no clear relation between the two variables. Figure 24 demonstrates a slight increase in average pod speed for high percentages of cocurrent flow (and correspondingly low percentages of counter flow). However after analysing the simulation carefully it became clear that this slight increase was down to highly subjective characteristics of the pod agent.

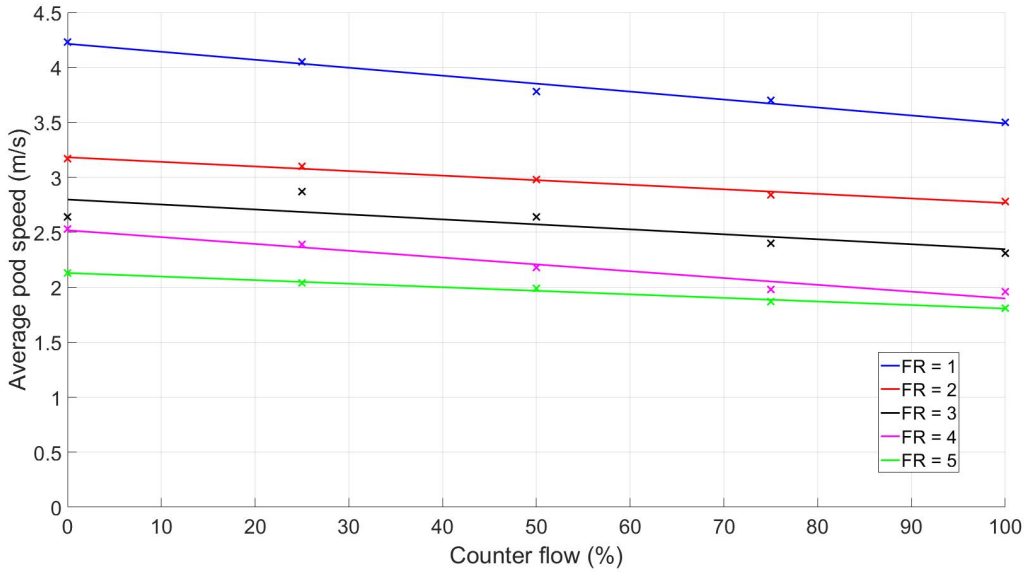

Figure 24: The effect of the ratio of cocurrent flow to counter flow on average pod speed

95% of pedestrian agents move aside to let a pod pass. When travelling in counter flow to a pod agent, these pedestrians start moving aside when the pod is about 6 seconds away from colliding with them. Pedestrian agents travelling in cocurrent flow to the pod start moving aside when the pod is also a similar time away from them. This may seem unrealistic given pedestrians face the same way as a pod when travelling in cocurrent flow and so can't see the pod approaching. However, it was assumed that an autonomous pod would emit a noise to warn pedestrians, travelling in cocurrent flow of its presence. This results in these pedestrians moving aside for the pod, despite the fact that it cannot see them.

In cocurrent flow pods can slow down less than in counter flow since the velocity of the pedestrian is away from them rather than towards them. Pods are able to travel faster 'going with the flow' rather than'going against it'. This effect results in the trend illustrated in Figure 24. However any tweaks to the pod specification significantly alter the

relationship between the ratio of cocurrent flow to counter flow and average pod speed. For example Figure 25 illustrates how this relationship becomes much weaker when a pod is no longer able to emit a noise to warn pedestrians of its presence. For this simulation pedestrians travelling in cocurrent flow only made way for pods travelling a couple of metres behind them.

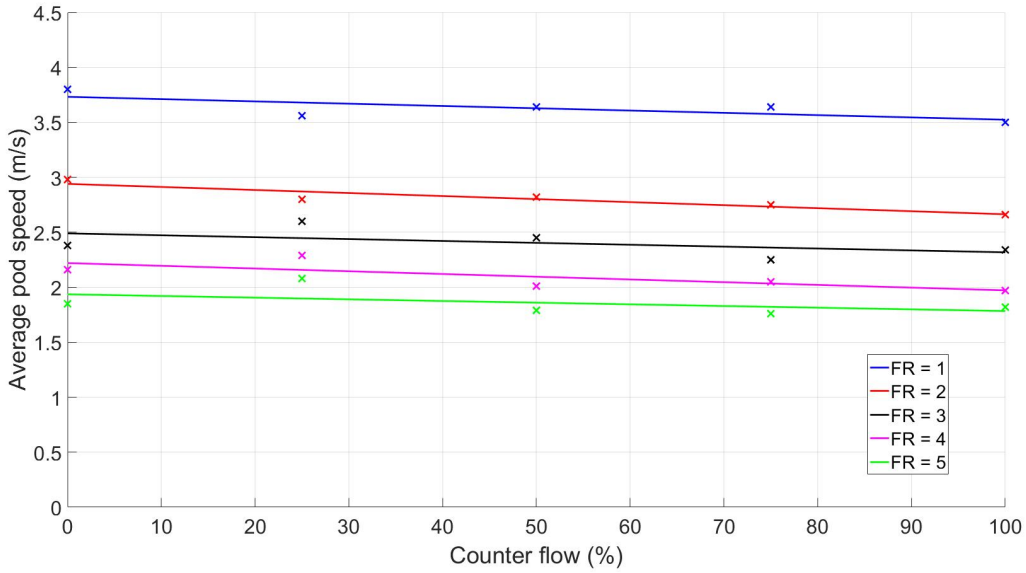

Figure 25: The effect of the ratio of cocurrent flow to counter flow on average pod speed (when a pod can no longer emit a noise)

4.5 Obstacle avoidance

Simulations into using MassMotion's social forces algorithm with the pod agent, to allow it to avoid obstacles in the environment, had mixed success. Since the social forces algorithm is not calibrated for large fast moving agents it suggests sharp changes in direction to the pod agent. However, at high speeds the pod is unable to make sharp changes in direction due to its turning radius constraint. Pedestrian agents are always able to make sharp changes in direction since they travel at walking speed and have zero turning radius.

The simulations were successful in demonstrating how the majority of pod agents could could navigate from their origin to their goal, using the social forces algorithm, without colliding with any objects, Approximately 80% of pods run in the simulation were successful in avoiding with collisions, however a significant number of pods ended up colliding with obstacles in the simulation environment. Pedestrians agents, unlike pod agents, do not need to give obstacles a wide berth since they have a small radius and are able to

change direction instantaneously. By making use of the social forces algorithm pod agents ended up not giving obstacles in the environment sufficient space, often brushing against them. In addition to this the social forces algorithm caused the pods to wander in the fashion of a pedestrian, instead of travelling in straight lines, which harmed the effectiveness of its collision detection and avoidance algorithms.

Since pods are still required to be simulated in open environments, for the large-scale simulation of the city centre, a decision was made to simulate agents in these environments without the presence of obstacles. Too many of the pods run in this simulation failed to complete their journey correctly for the simulation's results to be considered consistent. Instead, pods were simulated in these environments without using the social forces algorithm and with all the obstacles removed from the environment. Safety factors were added to the pod's average journey time, to account for the effect of the obstacles. It was assumed that the magnitude of the safety factor would approximately correspond to the total area of the environment that the obstacles take up.

The simulations of the pod's obstacles avoidance algorithm did prove to be useful for visualisation purposes. Adding obstacles into the simulation environments made them more easily recognisable, since it makes them better match the locations they represent in the real city centre. Hence when presenting videos of the simulations, it was useful to demonstrate the operation of the autonomous pods in Milton Keynes, using videos of the pods operating in environments populated with obstacles.

4.6 Large-scale simulation of the city centre

Milton Keynes city centre was broken down into the individual pedestrianised pathways and spaces that the pods can operate in. The distance that pods would travel in these environments was estimated so that, using the simulation results of the pod's average speed, the estimated time taken for a pod to travel through each environment could be calculated. Time delays were added to environments with junctions, where pedestrians and pods have to slow down to cross roads, since they must check for other traffic. It was found to be easiest to express the results of this simulation using a set of 'trip networks'.

This report contains six of these trip networks in total. Three of the trip networks show the average journey times for pods travelling in each individual environment at a range of times during the average weekday in Milton Keynes. The range of times was selected to be 8-9am, 1-2pm and 5-6pm so that the different journey times of pods in the morning, afternoon and evening could be compared. The remaining three trip networks shows the average journey times for pods on the average weekend day. The average weekday and

Weekday: 1-2pm

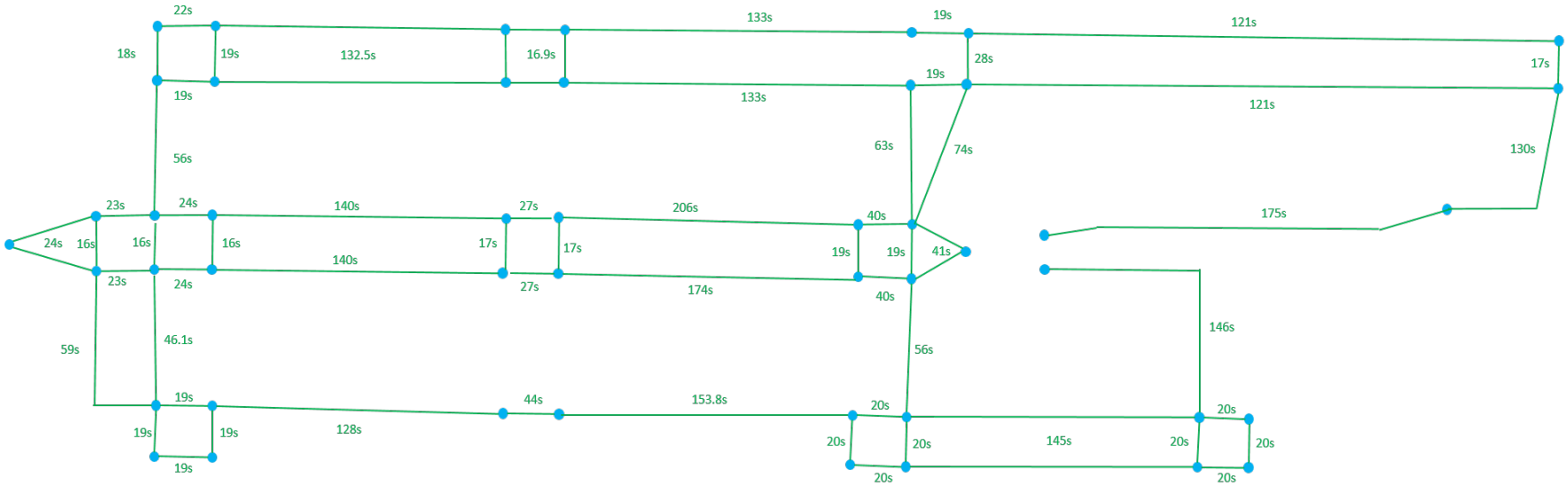

Figure 26: Trip network corresponding to a weekday at 1-2pm

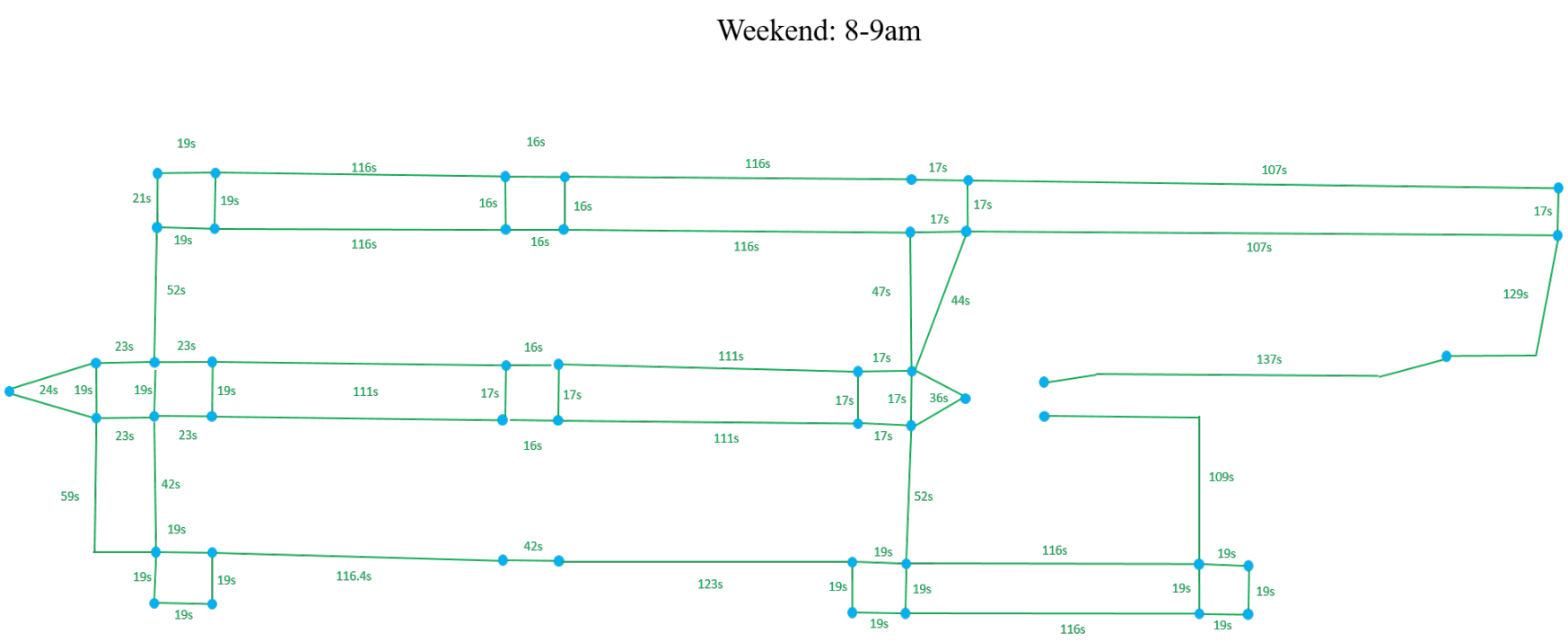

Figure 27: Trip network corresponding to a weekend day at 8-9am

weekend day were created by averaging data collected over a period of just over a month in February and early March 2018.

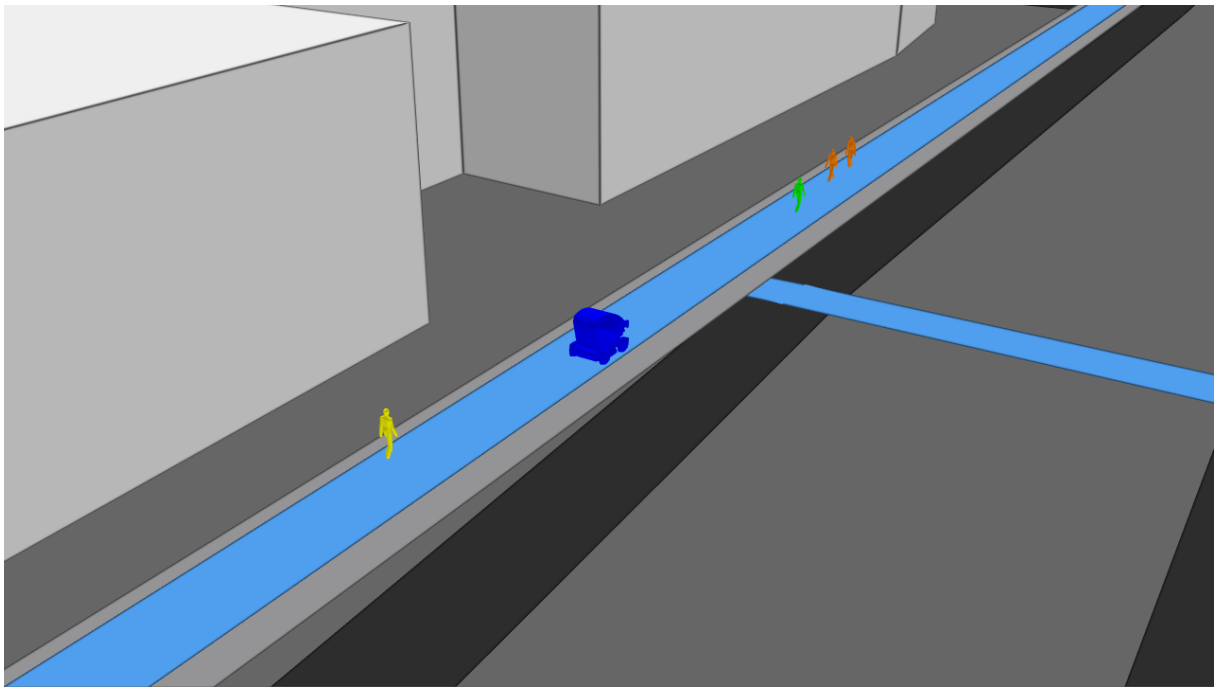

Figure 28: Image from the large-scale simulation of Milton Keynes city centre

A trip network illustrates the sensitivity of journey duration to the time and type of day. A typical journey for a shopper in Milton Keynes has been assumed to be the walk from the train station to the shopping centre. On a typical weekend morning this is estimated to take around 6 minutes by autonomous pod, which is one of the times at which this part of the city is quietest. This same journey takes approximately 8 minutes by pod in the early afternoon on a week day, which corresponds to one of the times at which this part of the city is busiest. Figures 26 and 27 were used to estimate these journey times as they illustrate the relevant trip networks. A breakdown of how the trip networks relate to the geography of Milton Keynes is shown in the Appendix.

The journey from the train station to the shopping centre is estimated to be a 17 minute walk for pedestrians. Pedestrian journey times are not significantly affected by the busyness of the pedestrianised walkways so this journey time can be considered to be constant. A simple acceptance test for the L-SATS is to check whether pod journeys are at least twice as fast as walking. The trip networks can be used to evaluate the L-SATS against this measure and generally it can be seen that the autonomous pod journey times meet this criteria for success. For example even along a main route in Milton Keynes, from the train station to the shopping centre, at a busy time it takes autonomous pods 8 minutes to complete a 17 minute journey by foot.

The large-scale simulations of the L-SATS operating in Milton Keynes city centre can also be used to produce revenue estimates for the transport system. This project has created a tool that allows journey times for the L-SATS to be estimated. This is useful to analysis of the business viability of the transport system, since it allows the number of journeys autonomous pods would be expected to complete each day to be calculated. Given each journey costs a small fee, then an estimate of the revenue created by each individual pod can be derived. This is an important parameter required to calculate the total fleet size, since the system must be profitable to run.

Although the trip networks were produced by breaking down the large-scale simulation into a series of components, and then simulating each component individually, simulations were also run in the city scale model of Milton Keynes. It was found that the simulation of the city scale model was useful for visualisation purposes. It is often difficult to convey the concept of an L-SATS that consists of autonomous pods, to those unacquainted with the concept. Although the trip networks provides a better means of estimating pod journey times in the city centre, due to the computational cost of simulating the entire city model at once, videos of pods operating in the city in its entirety demonstrate how the L-SATS would operate in practice.

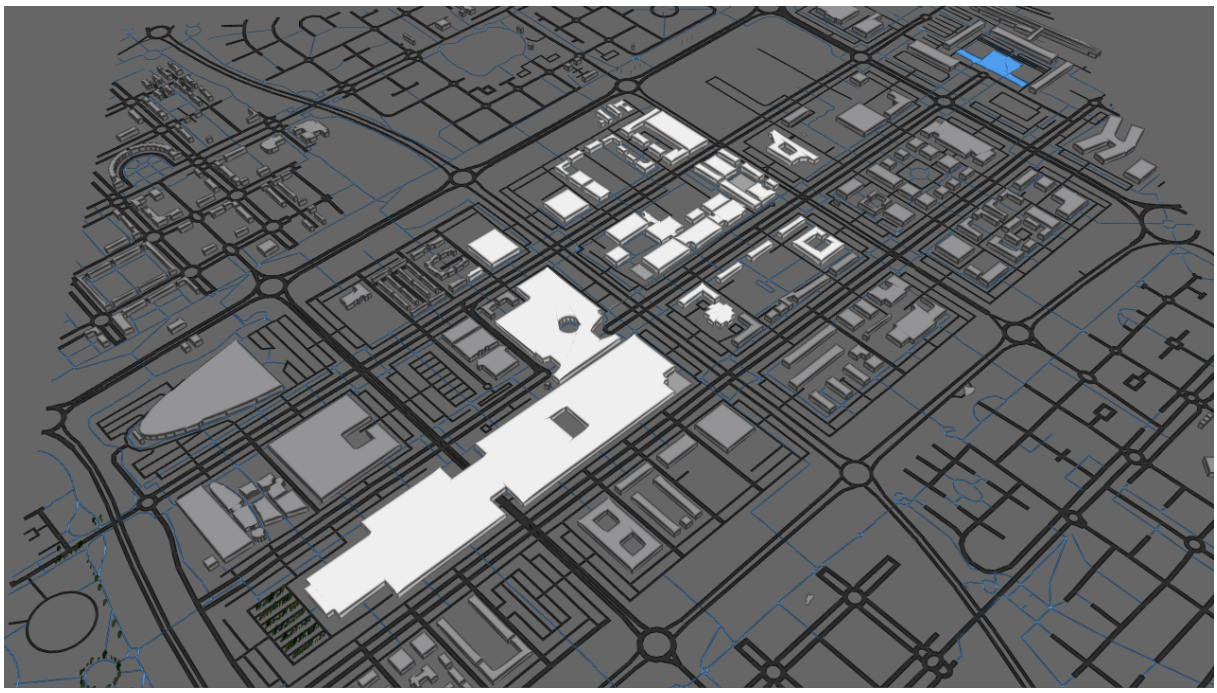

Figure 29: Aerial view of the large-scale simulation of Milton Keynes city centre

5 Conclusions

This project was successful in creating realistic simulations of autonomous vehicles interacting with pedestrians. This demonstrates how MassMotion can be extended to include both autonomous vehicle and pedestrian agents. MassMotion is an ideal software package on which to build these types of simulations. It is a world leader in its agent based modelling of pedestrians, it is well maintained and user friendly but most importantly it has a well-document software development kit so that its functionality can be easily modified. However there are also challenges and limitations with creating autonomous vehicles in MassMotion. The default agent class has few attributes and methods that can be recycled in a pod agent class since the software has not been designed to accommodate last, fast-moving agents such as autonomous vehicles.

Through preliminarily experiments it was shown that a complex pod specification may be worth the extra investment over one for a simple pod. A complex pod, that can avoid collisions by taking evasive action and can calculate collision points with pedestrians so it only slows down when necessary, has significantly better performance over a simple pod. This is particularly apparent in moderately busy environments, where their ability to navigate around pedestrians allows them to maintain a higher average speed. Additionally, in the real world autonomous pods are likely to encounter unexpected obstacles in their path. Whereas a simple pod would struggle to deal with such scenarios a complex pod would be much more effective in finding a new route around the obstacle.

Preliminary experiments in this project also found that there is unlikely to be any benefits to splitting the pathways in Milton Keynes into individual pod and pedestrian lanes. Although this idea appeared intuitive at the start of the project, simulations have suggested that this approach would be detrimental to pod performance. A split lane arrangement lowers a pod's top speed and, due to the tenancy of pedestrians to cluster as groups on pathways, the arrangements yields no benefit when the pathways are busy. It would be an expensive and time consuming task to paint theses lanes in the pedestrianised areas in Milton Keynes, so it has been useful to confirm that this task is not worth undertaking.

This project has created a useful tool for analysing how alterations to an autonomous pod's specification and its environment affect its performance. It has been demonstrated that autonomous pods have higher average speeds when they travel along wider walkways. Experiments have also quantified how increases in path width increase their average speed, so an estimate for the return on investment for widening the pathways can be calculated. In addition it has been shown that assumptions about a pod's behaviour can have unexpected impacts on its performance, for example, pods must have a reduced top speed

when travelling in narrow lanes.

An autonomous pod's agent specification is highly subjective and it requires continuous communication with the engineers who design the vehicle in order to create an accurate agent specification. There has been little experimentation into how pedestrians might interact with autonomous pods in the real world, and so many assumptions have been made in this project that might not reflect reality. For example, it was demonstrated that if autonomous pods can emit noises, to warn pedestrians travelling in cocurrent flow to it, of its presence, then pods can pass them at a greater speed. However it is unclear how often autonomous pods will be able to emit noises since this action may be classed as noise pollution.

Although a simulation of the entire model of Milton Keynes city centre proved to be too time consuming to undertake, small simulations run in the city centre model have proved to be useful for visualisation purposes. It is often difficult to convey the concept of an L-SAT to engineers and business people unacquainted with the system. Although still images do help peoples conceptualize the idea of a taxi service comprised of autonomous pods, the videos produced from the simulation of the city centre have proved to be far more useful. Similarly, although a robust algorithm, which allows pods to avoid pedestrians and obstacles in open spaces, was unable to be developed the work in this area has not been futile. The majority of pods run using this algorithm can be simulated successfully and so video illustrations of pods operating in prominent pedestrianised locations ,such as outside stations and shopping centres, can be made.

The large-scale simulation of the L-SATS operating in Milton Keynes city centre, has been summarised using the set of trip networks. These trip networks are a useful tool which allows the business viability of the L-SATS to be evaluated. The trip networks allow journey times for the autonomous pods, travelling between any two locations in the city centre, to be estimated which can be used to produce revenue calculations in the future. The trip networks also allow the usefulness of the transport system to be assessed, since it can be checked whether journey times by autonomous pod take at least half the amount of time it would take to walk the journey.

A fundamental assumption of this project is that 5% of pedestrians will refuse to move out of a pod's way, and will only decide to deviate from their course at the last possible moment. This type of behaviour greatly reduces average pod speed in simulations but the accuracy of this assumption is unknown. There have been no real world tests to collect data to evaluate this assumption and so this will form a key part of the future work for this project. In addition the pedestrian data used in the large-scale simulation was

collected over a period of a month. It would be useful to re-run these simulations with data corresponding to other months in order to cross validate the simulation results.

Other important future work includes optimising the programs used to run the agent-based simulations, so the entirety of Milton Keynes city centre can be simulated simultaneously. This would allow journey times for the L-SATS to be assessed without using trip networks, which were very time consuming to produce. In addition, the agent models for the pods and pedestrians should be refined. In particular, it would be useful to develop a variant of the social forces algorithm for pod agents, so they are able to avoid both pedestrians and obstacles in a satisfactory manner.

6 Appendix

6.1 Risk assessment retrospective

Back pain proved to be a risk needed to be mitigated. Making use of correct chairs. Eye strain at times, taking regular breaks clearly helped. No issues with repetitive strain injuries. No electrical hazards

6.2 Trip networks

6.3 Code for pod and pedestrian sequence diagrams

References

[1] McKinsey&Company
`Automotive revolution perspective towards 2030` McKinsey&Company, 2016.

[2] Fagnant, D. *The travel and environmental implications of shared autonomous vehicles, using agent-based model scenarios.* Transportation Research Part C: Emerging Technologies, 1-13, 2014.

[3] Schelhorn, T.
`Streets: an agent-based pedestrian model` UCL Discovery, 1999.

[4] Crooks, A. *Introduction to Agent-Based Modelling* Agent-Based Models of Geographical Systems (pp. 85-105). Springer, 2012.

[5] Oasys Software Limited *MassMotion Help Guide* 2017

[6] Morrow, E. *Personal communication* Product Director, MassMotion