A Hierarchical Pedestrian Behaviour Model to Reproduce Realistic Human Behaviour in a Traffic Environment

by

Scott Larter

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2022

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Understanding pedestrian behaviour in traffic environments is a crucial step in the development and testing of autonomous vehicles. As the environment's most vulnerable road users, pedestrians introduce an element of unpredictability that can lead to dangerous scenarios if their behaviours are unfamiliar to or misinterpreted by vehicles. In this thesis, we present a hierarchical pedestrian behaviour model that interprets high-level decisions through the use of behaviour trees to produce maneuvers that are executed by the low-level motion planner using an adapted Social Force Model. The presented hierarchical model is evaluated on two real-world data sets collected at separate locations with different road structures. The first data set provides a busy four-way intersection with signalized crosswalks, while the second location provides an unsignalized crosswalk across a two-way road at a Canadian university. Our model was shown to replicate the real-world pedestrians' trajectories and decision-making processes with a high degree of accuracy given only high-level routing information (start point, end point, and average walking speed) for each pedestrian. The model is integrated into GeoScenario Server, extending its vehicle simulation capabilities with pedestrian simulation. The extended environment allows simulating test scenarios involving both vehicles and pedestrians to assist in the scenario-based testing process of autonomous vehicles.

Acknowledgements

First and foremost, I would like to thank my supervisor, Professor Krzysztof Czarnecki, for guiding and supporting me through my research and always providing invaluable advice and feedback. This work would not be possible without the support from Rodrigo Queiroz who assisted me in extending his GeoScenario Server code with my pedestrian model. His expert knowledge and advice influenced my design decisions and frequently saved me hours of debugging. I would also like to thank and acknowledge all the members of the Waterloo Intelligent Systems Engineering (WISE) Lab, including but not limited to Michał Antkiewicz for his expertise on everything technical, Atrisha Sarkar for all her help in creating and maintaining clean and useful data sets, and Sean Sedwards for his insight and experience with the Social Force Model. Thank you all for your constant support and guidance. I owe a great debt of gratitude to all of you.

Dedication

This thesis is dedicated to my ever-supportive partner, Melissa, and my friends and family without whom none of this would be possible.

Table of Contents

List of Figures **ix**

List of Tables **xi**

1 Introduction and Motivation **1**

2 Related Work **5**

2.1 Agent-based Microscopic Pedestrian Models 5

2.1.1 Social Force Models 6

2.1.2 Cellular Automata 7

2.1.3 Game Theoretic Models 8

2.1.4 Machine Learning Models 10

2.2 Behaviour Trees 11

3 Background **12**

3.1 Social Force Model 12

3.2 Behaviour Trees 16

3.2.1 Selector and Sequence Nodes 17

3.2.2 Maneuver and Condition Nodes 18

3.2.3 Simple Behaviour Tree Example 18

3.3 GeoScenario 20

3.4 Lanelet2 Maps 21

4 Hierarchical Pedestrian Behaviour Model **22**
4.1 Model Design . . . 22
4.2 Model Architecture . . . 25
4.2.1 World Model and Pedestrian Representation . . . 26
4.2.2 Traffic State Estimation and Pedestrian Motion . . . 29
4.2.3 Behaviour Layer . . . 30
4.2.4 Maneuver Layer . . . 34
4.2.5 Motion Planner Layer . . . 42
4.3 Model Implementation . . . 43

5 Evaluation **44**
5.1 Configuration of Evaluation Scenarios with Prerecorded Trajectories . . . 44
5.1.1 Evaluation Scenario Configuration . . . 44
5.1.2 Intersection Data Set . . . 45
5.1.3 Full and Segmented Evaluation Scenarios . . . 47
5.2 Producing Realistic Low-Level Trajectory Movements (RQ1) . . . 48
5.2.1 Trajectory-Matching Metrics . . . 49
5.2.2 Empirical Results . . . 52
5.3 Replicating High-Level Decisions (RQ2) . . . 59
5.3.1 Decision-Based Metric . . . 59
5.4 Extending Model to Different Road Structures (RQ3) . . . 61
5.4.1 Single Crosswalk Data Set . . . 62
5.4.2 Evaluation Metrics and Empirical Results . . . 63

6 Limitations **65**
6.1 Research Scope Limitations . . . 65
6.2 Model Limitations . . . 65
6.3 Evaluation Limitations . . . 66

7 Conclusions and Future Work **67**

7.1 Conclusions 67

7.2 Future Work 68

7.2.1 Improvements to Model 68

7.2.2 Future Research Directions 69

References **70**

List of Figures

3.1 Visualizing the components of the body compression and sliding friction forces acting on a pedestrian (i) in close proximity to another (j). 14

3.2 Example of a simple behaviour tree used to determine the appropriate maneuver for a pedestrian waiting to enter and cross a signalized crosswalk. . 19

4.1 Using the Java Open Street Map editing tool to edit a scenario. 23

4.2 Tree (a) and text (b) representations of the same *enter_xwalk_on_green* behaviour tree 24

4.3 Model diagram with process and information flow. 26

4.4 Default behaviour tree assigned to model pedestrians. 31

4.5 Enter Crosswalk sub-tree - Aggressiveness Level 1 32

4.6 Enter Crosswalk sub-tree - Aggressiveness Level 2 34

4.7 Enter Crosswalk sub-tree - Aggressiveness Level 3 35

5.1 Side-by-side comparison of the intersection data set location with a shot taken from Google Earth (a) and its simplified representation (b). 46

5.2 A visual representation of the set of defined points (red) with which a baseline trajectory is generated to approximate each empirical trajectory. An example baseline trajectory is shown by the blue path. 50

5.3 A trace of the trajectories from the empirical (red) and simulated (blue) pedestrians overlaid on the intersection map from one evaluation scenario. 53

5.4 Example trajectories resulting from both corner cutting and non-corner cutting behaviours when entering a crosswalk. 54

5.5 Sub-figure (a) shows evaluation scenario where SP pedestrian (blue) does not apply the corner cutting procedure. Sub-figure (b) shows the same evaluation scenario except where SP pedestrian applies the corner cutting procedure to match the empirical trajectory (red) more closely. 57

5.6 Sub-figure (a) shows evaluation scenario where SP pedestrian (blue) does not apply the jaywalking maneuver. Sub-figure (b) shows the same evaluation scenario except where SP pedestrian applies the jaywalking beside the crosswalk maneuver to match the behaviour of the empirical pedestrian (red). 58

5.7 Side-by-side comparison of the single crosswalk data set location with a shot taken from Google Earth (a) and its simplified representation (b). 62

5.8 Trajectory traces of three example evaluation scenarios with varying paths from the single crosswalk data set. 64

List of Tables

5.1 Counts of road users by type in the intersection data set. 46

5.2 Results of the Euclidean distance (ED), Fréchet distance (FD), and Hausdorff distance (HD) measures averaged and grouped by full and segmented evaluation scenarios at the intersection data set location. 52

5.3 Summary of decision-based metric results on full length evaluation scenarios using the intersection data set. 60

5.4 Counts of road users by type in the single crosswalk data set. 63

5.5 Trajectory-matching metrics results for all evaluation scenarios at the single crosswalk data set location. 63

Chapter 1

Introduction and Motivation

Understanding pedestrian behaviour in traffic environments is a crucial step in the development and testing of autonomous vehicles (AV) and automated driving systems (ADS). As the environment's most vulnerable road users, pedestrians introduce an element of unpredictability that can lead to dangerous situations if their behaviours are unfamiliar to or misinterpreted by vehicle drivers. Through rigorous testing of a wide range of traffic scenarios, AVs can begin to learn and understand pedestrian behaviours in order to better handle interactions with traffic's human participants. This introduces the problem of finding a systematic approach for setting up and testing the interactions between AVs and human pedestrians and identifying key situations relevant to the AV's decision-making process while ensuring the safety of all participants.

Our work aims to address this problem through the integration of simulated pedestrians into the scenario-based testing process of autonomous vehicles. If pedestrians within a traffic scenario have customizable and, more importantly, explainable actions and behaviours, then certain desired behaviours can be reproduced in testing. This allows testers to isolate events of interest (e.g. a collision between a vehicle and pedestrian) and trace them back to the sequence of pedestrian decisions and environmental contexts that led to it.

This thesis presents a hierarchical pedestrian behaviour model for traffic environments that incorporates a variation of the Social Force Model (SFM) to control low level movements and behaviour trees to handle higher level decision making processes. This model allows scenario designers to create pedestrian agents that can dynamically navigate various traffic environments with realistic decisions and behaviours while interacting with other pedestrians and vehicles in the scene. The main goals of building the model are to facilitate the scenario-creation process in scenarios where autonomous vehicles interact with pedestrians, analyze real-world traffic data to identify and isolate unusual pedestrian behaviours, and determine which aspects of pedestrian behaviours are relevant to the development and testing of AVs. With these goals in mind, our research is driven by the following questions regarding pedestrian behaviours in traffic environments and why it is important to study them:

Q1: Can the complexity of human behaviours in a traffic environment be modelled in a relatively simple way with a predetermined set of basic maneuvers analogous to vehicle maneuvers?

Q2: Which aspects of pedestrian behaviour are relevant to AV testing and development?

Q3: How can pedestrian simulation be integrated into the AV development process to effectively improve AV's capabilities and intelligence?

This thesis attempts to provide complete and comprehensive answers to these research questions. As the scope of questions 2 and 3 are intentionally broad, we use them as motivation for our work and acknowledge that this thesis does not completely cover all aspects of the questions. In these cases, we propose future research directions that build on our contributions to continue the pursuit of complete and satisfactory answers.

In terms of Q1, an important task within the scope of understanding pedestrian behaviours is parsing individual and sequences of maneuvers performed by pedestrians throughout their journey. An ideally complete list of maneuvers provides many advantages to researchers. In human prediction tasks, common maneuver sequences and patterns can be analyzed to predict likelihoods of future actions or determine if certain maneuvers are more likely to lead to dangerous situations. Sequences of atomic maneuvers can be used to construct composite pedestrian-specific maneuvers, such as crossing a crosswalk or jaywalking. Composite maneuvers can help intuitively tag and categorize real-world scenarios derived from data. With respect to the pedestrian simulation model presented in this thesis, a list of maneuvers is directly beneficial as it helps construct the output space of the decision-making processes for each agent.

The concept of extracting and analyzing pedestrian maneuvers has been applied within the context of traffic analysis. There are existing reports on traffic accidents involving pedestrians in the US [81] [71] in which investigators note the vehicles' and pedestrians' maneuvers at the times of the accidents. Noting pedestrian actions has also played a role in designing safer temporary traffic control processes at construction sites [76]. However, the maneuvers provided in these reports are generally high-level and broad (e.g.

crossing/in/adjacent to road, walking with/against traffic) and therefore not necessarily suitable to compile a complete list possible maneuvers to be used in simulation. There are existing works that make use of maneuvers for vehicles within AV navigation systems [10] [77], rule-based planning systems for AVs [11], and numerous other AV-specific applications. Within the context of pedestrian simulation, we have not found an analogous set of maneuvers to be used as the output space of pedestrian decisions. To fill this gap, a sufficiently complete set of maneuvers was extracted from naturalistic data sets and its completeness was validated through our evaluation process.

With respect to Q2 and Q3, we attempt to capture an explicit representation of pedestrian behaviours through a set of maneuvers formed into behaviour trees. Having this representation lends itself well to the scenario-based testing of AVs process since engineers can inject desired behaviours into scenarios. Often these desired behaviours induce rare or dangerous events that cannot be easily found in data. This is advantageous for the development of AVs since their behaviours can be thoroughly tested on high-risk events in a safe simulation environment. It is left for future work for the scenario design and testing process to further explore the aspects of pedestrian behaviours that lead to important events for the AV. We present the framework to facilitate this process.

There is an important distinction to be made between pedestrian *prediction* models and *behaviour* models. The goal of the former is to, as accurately as possible, predict a pedestrian's future trajectory over a relatively short horizon given the past trajectory up to the current time. The metrics used to evaluate such models are highly concerned with minimizing precise distance metrics such as the average Euclidean distance and the average displacement error. Conversely, behaviour models are more concerned with producing realistic decisions made by pedestrians rather than exactly matching each low-level trajectory point. For example, if a behaviour phenomenon is observed, such as lane formation when two groups of pedestrians are passing each other in opposite directions in a narrow corridor, then a behaviour model would be expected to replicate this behaviour without explicitly being told to do so. In a traffic environment, one of these decisions could be when and where a pedestrian decides to enter a crosswalk. The model presented in this thesis is intended to be a behaviour model and, while trajectory-matching distance metrics are still relevant to our evaluation process, the model's performance is also judged on its ability to replicate certain behaviours and decisions observed in the data.

Our model was evaluated on two different data sets extracted from drone recordings at two separate locations in Waterloo, Canada: a high-traffic intersection with a higher density and number of vehicles than pedestrians and a single crosswalk at a university with a higher density and number of pedestrians than vehicles. The locations of the two data sets were selected for their differing road geometries and distributions of traffic participants. The first is a busy four-way intersection containing traffic lights, pedestrian crossing signals, and six zebra crossings including two right-turn merge lanes (or slip lanes), while the second is at a single unsignalized zebra crossing with heavy foot-traffic across a less busy two-lane road.

The evaluation process is driven by three explicit goals for our model: accurately generate and reproduce low-level trajectories observed in data, replicate the same high-level decision-making processes made by real-world pedestrians in a traffic environment, and prove its extensibility and generalization to environments with differing road structures. We generate evaluation metrics to measure each one of these requirements and our model is shown to meet and exceed the given evaluation criteria.

The novel contributions presented in this thesis are as follows:

1. A hierarchical pedestrian behaviour model that incorporates behaviour trees and an adapted Social Force Model that is capable of producing realistic high-level decisions and low-level trajectories to navigate through traffic environments with varying road structures.

2. Extension of the GeoScenario Server simulation environment to integrate dynamic pedestrians into scenarios along with the existing simulated vehicle agents.

3. A complete set of basic pedestrian maneuvers sufficient for replicating the observed human actions in two real-world data sets.

4. A suite of behaviour trees and subtrees that provides scenario designers and testers with customizable and reproducible pedestrian behaviours through three defined levels of aggressiveness observed in real-world data.

The structure and content of the remainder of this thesis is as follows. Chapter 2 explores existing pedestrian models and their varying approaches in literature. Chapter 3 discusses the necessary background information needed to understand the components and concepts integrated into our work. The design, architecture, and implementation of our presented model is described in Chapter 4 and the evaluation methods and results are explored in Chapter 5. Finally, Chapter 7 discusses the conclusions drawn from our work and possible future research avenues that can build on our contributions.

Chapter 2

Related Work

Given the non-trivial task of handling the dynamic nature of pedestrians, researchers have theorized and applied a great number of different approaches in order to detect, predict, reproduce, and explain pedestrian behaviours. Our proposed model incorporates a low-level motion planner layer driven by an adapted Social Force Model with a decision making layer that applies customized behaviour trees to determine appropriate maneuvers and decisions. Since the model considers each pedestrian to be its own individual agent that adopts a separate instance of the model, we compare our proposed approach to other agent-based microscopic models. While macroscopic pedestrian models exist in literature [63] [58] [79], they have historically been used to study crowd dynamics and they consider the collection of pedestrians in the scene as a continuous medium characterized by properties determined by the crowd instead of each individual participant. This approach does not lend itself well to the scenario-based testing process where the interactions between individual or small groups of pedestrians and vehicles need to be isolated and analyzed.

2.1 Agent-based Microscopic Pedestrian Models

An agent-based microscopic model considers the individual dynamics and characteristics of each agent and how it interacts with other agents and objects within the environment. This type of model does not limit itself to simulating only large group crowd dynamics and allows researchers to study the interactions within small groups of agents as well. One of our objectives is to create a model that accurately replicates individual behaviours of pedestrians while maintaining the capability to be extended to model any arbitrary group size.

In the following sections, we compare existing pedestrian models to our research to explain the gaps in current literature and how our model can help fill them.

2.1.1 Social Force Models

Arguably one of the most popular approaches to pedestrian behaviour modelling is through the use physics-based models. These models consider physical and "social" forces acting on the agent. An agent's movement is driven by a combination of all the forces acting on it at each time step. This approach gained recognition with the introduction of Helbing and Molnár's Social Force Model [29]. The authors originally applied their model as a way to simulate the collective behaviour of a panicking group of people evacuating various room types in an emergency situation [28]. The basis of their model relied on three main collections of forces acting on an agent: an attracting force drawing the agent towards their goal, a repulsive force away from each other agent in the scene, and repulsive forces from all walls and borders in the environment. The motion resulting from the sum of these forces was shown to accurately model human movements in their presented scenarios.

While this model seemed to accurately represent human behaviour in the scenarios presented by the authors, researchers began applying Social Force models to other situations. Lakoba et al. [34] propose modifications to the original model to accurately replicate behaviours of individual or small groups of pedestrians in contrast to large crowds. Up to this point, however, the Social Force Model had not been applied to pedestrians within a traffic environment.

Suh et al. [68] explored how Helbing and Molnár's model could be used at a signalized crosswalk setting which allowed for novel conclusions to be drawn regarding pedestrian behaviours at a crosswalk. They observed that the majority of pedestrians disregarded the signal state when making their crossing decision and instead employed a gap-acceptance approach to crossing. While we observe instances of this behaviour in our intersection data set, the location is usually too busy for pedestrians to rely on gap-acceptance to cross and instead we observe that the crossing signal states are much more influential in their decisions.

Variations of the Social Force Model have since been applied to the signalized crosswalk setting across multiple countries including the US, China, and Japan. Notable modifications focus on calibrating optimal parameter values via a maximum log-likelihood estimation to improve collision avoidance between pedestrians and vehicles [85], taking into account the dynamics and phenomena that occur with counter-flow and leading pedestrians [46], and studying the effect of a flashing green signal (or countdown to stop signal state) on pedestrians already in the crosswalk [88].

Multi-layer approaches to model the movements and behaviours of road users have also made use of Social Force models within their layers, especially within the context of shared spaces. These approaches include Anvari et al.'s shared space model [8] in which both vehicles and pedestrians adopt a variation of a SFM with different constraints based on their physical restrictions. The model incorporates a trajectory-planning layer to find shortest routes, a Social Force layer to generate feasible trajectories, and a rule-based layer to define constrains that cannot be modelled by the other layers. Calibration of their model was performed with a data set of a shared space location in the UK [7]. Rinke et al. propose another multi-layer approach for shared spaces with a SFM layer [64]. This particular model introduces a SFM implementation for cyclists as well as pedestrians and vehicles. Similar to the previous approach, this model contains a trajectory planning layer, a conflict prediction layer, and a Social Force layer to model interactions between their three types of road users.

In our work, we propose a similar multi-layer approach involving route planning from high layers and feasible trajectory generation from lower layers using a Social Force model. However, our model employs behaviour trees in the upper decision-making layer to enable a kind of "mode switching". Behaviour trees can track the stages of a pedestrian's journey and apply the appropriate inputs and constraints to the model's lower layers accordingly.

The Social Force Model and its variations has appeared in literature a number of times in the past two decades and has repeatedly been applied in traffic environments, ensuring that appropriate repulsive forces from vehicles are added and calibrated [8] [64] [19] [73]. In our work, we wish to extend the functionality of the Social Force Model through the use of behaviour trees on a higher decision-making level. With our approach, we are not limited to one set of parameters per agent per scenario. Instead, we take into account the environmental context of a given pedestrian and apply the most appropriate instance of the Social Force Model to the motion planner layer.

2.1.2 Cellular Automata

Apart from physics-based models, another popular approach to microscopic pedestrian simulation is through cellular automata (CA). A general cellular automaton model discretizes the physical environment into a grid of cells. For most CAs, a cell is designed to allow at most one occupying pedestrian, although the exact size of the cells vary between models. Rules are devised to define how agents may move between neighbouring cells. A common approach to define these rules is using a floor field, first introduced by Burstedde et al. [13], which maps neighbouring cells to probabilities of travelling between them.

Like the Social Force Model, a natural application of cellular automata was found to be in evacuation scenarios [72] [27] [83] [39] [57]. Within the scope of traffic environments, researchers have also found success in modelling vehicle traffic flow with cellular automata. The focus of these models range from studying lane-changing behaviour [42], improving traffic flow when crossing pedestrians are introduced [87] and when there are changes in vehicle speeds [60], studying the effects of autonomous vehicles on traffic flow [47], to modelling traffic flow at intersections with an Internet of Vehicles [86].

For comparisons to our work, we are interested in the use of cellular automata to model pedestrian flow and behaviour within a traffic environment. In recent years, CAs have been adopted to model vehicle-pedestrians interactions with the intent of modelling realistic pedestrian flows on crosswalks and roadways. Similar to Social Force Models, CA models have been localized to study pedestrian behaviours and interactions with vehicles at signalized intersections [43] [23]. These models showed that cellular automata models were effective at modelling complex pedestrian phenomena at signalized crosswalks and could potentially assist engineers in designing areas shared by pedestrians and vehicles in an urban setting. More recently, fuzzy logic was incorporated into CA models to model crowd dynamics at signalized intersections [18] and vehicle-pedestrian conflicts at unsignalized midblock crossings [38]. Lastly, Layegh et al. developed a cellular automata specifically designed to study interactions at a multi-lane roundabout [35] with the intent of extending the capabilities of CA models to different road structures.

Overall, cellular automata have been proven to effectively model pedestrian flow and even display complex pedestrian phenomena within traffic environments. However, defining the transition rates between cells in the discretized grid is heavily dependent on the specific road structure of the environment and does not generalize well to arbitrary traffic locations. We found that a Social Force Model approach was better suited to work with behaviour trees, as the decision-making layer could inform a small set of model parameters, and required less dependency on specific characteristics and properties of the environment.

2.1.3 Game Theoretic Models

Game theory has been accepted as a useful approach to modelling human behaviour due to our inherent utility-optimizing nature. Humans naturally tend to find ways to expend the least amount of energy during a task, travel the shortest route available, or even try to maximize personal gain in competition with other players which can all be intuitively modelled using game theory.

As we have seen with Social Force and cellular automata models, game theoretic models have been applied to the popular emergency evacuation scenario in which a large group of panicking people evacuate through a small number of relatively narrow exits [48] [72] [12] [26] [66] [45]. These scenarios have proven to be a natural application for game theory because, when evacuating, players are constantly competing for space against multiple other agents and usually have a stronger drive to advance their position than they would under normal conditions due to the severe nature of the scenario.

Within the scope of pedestrian motion and behaviour modelling, game theoretic models are often used to supplement other microscopic models, most notably, cellular automata [72] [26] [45] [78]. In these models, game theoretic concepts are applied to define the transition rates between neighbouring cells. Games are also used to resolve conflicts between two agents who both want to travel to the same cell. Note the overlap of these models with the evacuation scenario.

Game theoretic concepts have also been applied to Social Force models within a few applications. The emergency evacuation scenario is revisited by Zhu et al. [89] in their model where users compete against each other in optimization games to reach the best exit given multiple options. Johora et al. [33] propose a game theoretic Social Force model aimed at capturing realistic behaviours and interactions of pedestrians and vehicles in shared spaces. Their approach incorporates multiple layers responsible for trajectory planning, game theoretic decisions, and force based modelling with a focus on complex interactions between road users. Existing literature combining the use of game theory and Social Force models is currently limited.

Finally, even more relevant to the research in this thesis, game theory concepts have been employed to study and model the interactions between pedestrians and autonomous vehicles. A common scenario in which game theory is applied is when a pedestrian intends to cross the road and the vehicle must decide to either yield to the pedestrian or assertively pass the pedestrian before they cross [51] [15] [50] [14] [16]. This competition for space forms a game theoretic interaction between pedestrian and vehicle agent in which each player attempts to maximize their utility function while following the rules of the game.

Game theory has been proven to be a viable approach to modelling complex interactions between rational agents. In comparison to our use of behaviour trees with a SFM, each approach comes with its own advantages depending on the context and scope of the test scenario. Game theoretic models have their advantages in resolving non-trivial interactions between agents; whereas our approach allows a finer level of control over the behaviours of the agents through explicit representation. This provides benefits in scenario design where testers can force desired behaviours and situations.

2.1.4 Machine Learning Models

With the emergence of machine learning (ML) techniques to help solve real-world problems, pedestrian behaviour simulation has not escaped its rapidly expanding list of applications. A range of ML methods have been employed to develop agent-based pedestrian models. Torrens et al. [75] showed that synthetic non-sampled pedestrian trajectories and behaviours can be generated in a simulation environment through training a machine learning scheme on a combination of naturalistic data and agent-generated movements. More recently, an agent-based pedestrian behaviour model, HAIL, was developed to learn human behaviours through Imitation Learning of virtual agents [6].

Nasernejad et al. [55] considered a more localized scope in their work on studying pedestrian reactions to near-misses with vehicles in order to better understand evasive maneuvering. Their approach uses a continuous Gaussian Process Inverse Reinforcement Learning (GP-IRL) method to infer reward functions alongside a Deep Reinforcement Learning (DRL) algorithm to learn optimal policies. DRL has also been used to simulate virtual crowds in a dynamically changing environment [36].

There is a wealth of machine learning models to solve the pedestrian trajectory prediction problem in which future trajectories are inferred from a combination of past ground-truth trajectories and environmental inputs [4] [80] [69] [67] [49] [31] [25]. These prediction tasks have even been extended to full simulation environments. TrafficSim [70] uses naturalistic data to learn a multi-agent model capable of simulating all road users in traffic, including pedestrians. From the data, TrafficSim identifies complex agent maneuvers and interactions that can be leveraged and reproduced in simulation. As emphasized in Chapter 1, our interests do not lie in creating predictive models. Rather, our focus has been to produce natural and realistic pedestrian *behaviour* in longer-duration test scenarios.

In terms of the design choices for our model, machine learning techniques simply do not fully align with our research goals. As further explored in Section 4.1, explicit customization of pedestrian behaviours is a high priority in designing our model. An important objective is the ability to inject certain desired behaviours into existing test scenarios involving pedestrians. This proves to be difficult with learned models, since one must have sufficient existing data of the desired behaviour. This is especially amplified when testers want to inject rare events and behaviours that are not frequently found in data sets. For these reasons, we do not incorporate machine learning techniques into our pedestrian simulation model.

2.2 Behaviour Trees

Existing literature regarding the incorporation of behaviour trees into pedestrian simulation is currently limited. However, behaviour trees are a popular tool in the realm of game development for defining non-player characters' (NPC) movements and actions [32] [44] [65] [9] [3]. We employ a similar strategy to the development of in-game AI players, while adding degrees of intelligence through our motion planner layer.

In recent years, a few works have explored the use of behaviour trees to improve our understanding of pedestrian behaviours. Tomai et al. showed the benefits of Learning Behaviour Trees to simulate a human-like population [74]. Perhaps the work closest in similarity to ours was done by Paranjape et al. [56] in which simulation-based testing for AVs was performed by creating virtual cities and agents in order to find rare and interesting scenarios not yet existing in data sets. These approaches lack sufficient evaluation against real-world data to validate their models. To the best of our knowledge, behaviour trees have not been used in conjunction with an agent-based microscopic pedestrian behaviour model to simulate realistic pedestrian motion and behaviour.

Chapter 3

Background

3.1 Social Force Model

The Social Force Model (SFM) is responsible for driving low-level basic motion through its use of "physical" attracting and repelling forces to influence a pedestrian's next position. At its base version [28], the model applies three main collections of forces to the agent: an attracting force drawing the agent towards their destination, a repelling force from each of the other agents in the scene, and a repelling force from each wall or border in the environment. These three forces are summed into an acceleration equation that describes the pedestrian's change in velocity across consecutive time steps. Helbing's model has been shown to reproduce human behaviour phenomena such as the bottle-necking effect that occurs when a group of people with the same destination all attempt to exit a room through a narrow doorway and the single file lane formation behaviour when two groups walking in opposite directions towards each other meet in a narrow corridor. Throughout this thesis, Helbing's 2000 Social Force Model will be referred to as the classic or original SFM as our adaptation is based heavily on this version. We will discuss the basic formulas that make up the classic Social Force Model and how they work together to drive an agent's movements.

The attracting force component of the model is composed of a force vector directed towards the pedestrian's current destination or waypoint. Waypoints, which will be further discussed in Section 4.2.2, are intermediate destinations the pedestrian visits between its starting point and final destination point. Pedestrian i's acceleration over a given characteristic time τ is

$$\frac{dv_i}{dt} = \frac{v_i^0(t)e_i^0(t) - v_i(t)}{\tau_i}$$

where $v_i^0(t)$ is the pedestrian's desired speed, $e_i^0(t)$ is the unit vector pointing towards the current destination, and $v_i(t)$ is the current velocity. The first term of the force acting upon pedestrian i is therefore

$$f_{adapt} = m_i \frac{v_i^0(t)e_i^0(t) - v_i(t)}{\tau_i}$$

As pedestrians have a tendency to avoid contact with and maintain a comfortable distance from vehicles and other pedestrians while they are walking, the SFM incorporates a collection of repulsive forces between pedestrian i and all other agents in the environment. This second term of the equation is composed of two sub-forces, $f_{otherPeds}$ and $f_{vehicles}$, each a sum of force vectors describing the repulsive effect from other pedestrians and vehicles, respectively. Starting with other pedestrians, the force acting on pedestrian i is given by

$$f_{otherPeds} = \sum_{j, j \neq i} f_{ij}$$

where f_{ij} is the repulsive force of pedestrian j on pedestrian i. This interaction force implements three intuitive concepts that describe the natural interactions between pedestrians. The first ensures that pedestrian i maintains a comfortable distance from pedestrian j given their respective velocities and radii which is described by

$$A_i \exp[(r_{ij} - d_{ij})/B_i] n_{ij}$$

where A_i and B_i are coefficients, r_{ij} is the sum of pedestrian i and j's respective radii, r_i and r_j, d_{ij} is the distance between the pedestrians' centers of mass, and $n_{ij} = (n_{ij}^1, n_{ij}^2)$ is the unit vector pointing from pedestrian j to i.

The next two concepts are activated when pedestrians i and j are sufficiently close (i.e. $d_{ij} < r_{ij}$). In order to counteract body compression, a "body force" is included as $\phi g(r_{ij} - d_{ij}) n_{ij}$ where the function $g(x)$ is x when $x > 0$ and 0 when $x \leq 0$. To impede the tangential motion of two close moving objects, a "sliding friction force" is applied, described by $\omega g(r_{ij} - d_{ij}) \Delta v_{ji}^t t_{ij}$ where $\Delta v_{ji}^t = (v_j - v_i) \cdot t_{ij}$ and $t_{ij} = (-n_{ij}^2, n_{ij}^1)$. In the these forces, ϕ and ω are constants. Figure 3.1 visualizes the components of the body compression and sliding friction forces, leaving out constant values for simplicity. In the diagram, these forces are activated because the distance between the pedestrians' centers of mass is less than the sum of their radii. The body compression force is fairly simple and in the direction of n_{ij} in the figure. The resulting sliding friction force is represented by $\Delta v_{ji}^t t_{ij}$ which impedes pedestrian i's tangential movement due to j's position and velocity.

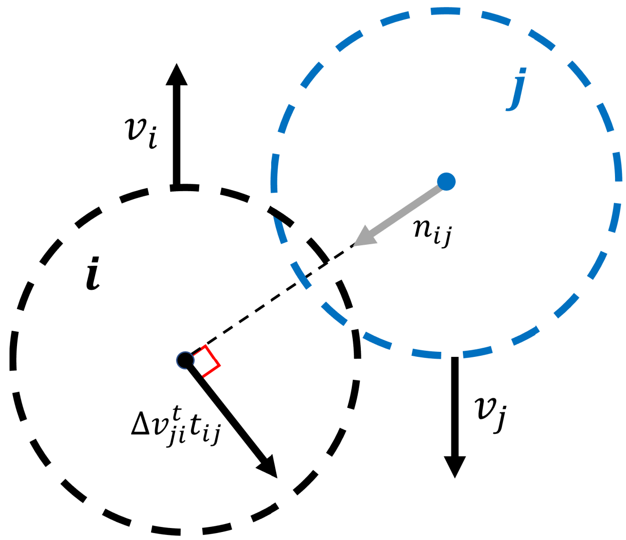


Figure 3.1: Visualizing the components of the body compression and sliding friction forces acting on a pedestrian (i) in close proximity to another (j).

Accumulating all the discussed interaction forces acting upon pedestrian i when pedestrian j is in close proximity into the final interaction force, f_{ij}, gives

$$f_{ij} = \{A_i \exp[(r_{ij} - d_{ij})/B_i] + \phi g(r_{ij} - d_{ij})\}n_{ij} + \omega g(r_{ij} - d_{ij})\Delta v_{ji}^t t_{ij}$$

Similarly, each vehicle in the scene produces an effect acting on pedestrian i shown in the sum

$$f_{vehicles} = \sum_k f_{ik}$$

The vehicle interaction force, f_{ik}, is comparable to the pedestrian interactions but it has a slightly simplified equation as presented by Anvari et al. [8]. The pedestrian treats vehicles as having an elliptical shape, or at least considers an ellipse around the body of the vehicle that defines the space it occupies. To construct the force term f_{ik}, we must first define terms describing each agent's shape and relationship to the other. The dimensions of the ellipse are defined by w and l representing the semi-minor and semi-major axes respectively. The "radius" of the vehicle, r_k, is the distance from its center to the edge of the ellipse in the direction of the pedestrian. We also determine the angle, ϕ_{ik}, between the vehicle's heading and the vector pointing from it to the center of the pedestrian. Given these defined terms and the constants A_k, B_k, and λ_i, the force acting on pedestrian i by vehicle k is described by

$$f_{ik} = A_k \exp[(r_{ik} - d_{ik})/B_k] n_{ik} F_{ik}$$

where $r_{ik} = r_i + r_k$, d_{ik} is the distance between centers of the pedestrian and vehicle, n_{ik} is the normalized vector pointing from the center of the vehicle to the pedestrian. Finally, the last term

$$F_{ik} = \lambda_i + (1 - \lambda_i)\frac{1 + \cos(\phi_{ik})}{2}$$

explains the behaviour of the pedestrian when facing the vehicle given its shape and heading.

Summarizing the collections of repulsive forces between agents and other pedestrians and vehicles around them gives the two sums

$$\begin{aligned} f_{otherPeds} &= \sum_{j, j \neq i} f_{ij} \\ &= \sum_{j, j \neq i} \left(A_i \exp\left[\frac{r_{ij} - d_{ij}}{B_i}\right] + \phi g(r_{ij} - d_{ij}) \right) n_{ij} + \omega g(r_{ij} - d_{ij}) \Delta v_{ji}^t t_{ij} \end{aligned}$$

and

$$\begin{aligned} f_{vehicles} &= \sum_{k} f_{ik} \\ &= \sum_{k} A_k \exp\left[\frac{r_{ik} - d_{ik}}{B_k}\right] n_{ik} \left(\lambda_i + (1 - \lambda_i)\frac{1 + \cos(\phi_{ik})}{2} \right) \end{aligned}$$

Similar to behaviours towards vehicles and other pedestrians, pedestrians tend to avoid walls and borders in their environment, which is implemented in the Social Force Model with the $f_{borders}$ force. The agent pedestrian i calculates an individual force, f_{iW} with each wall in its environment and sums all these forces to generate $f_{borders}$. Each wall force is determined by the below formula similar to the pedestrian interaction force.

$$f_{iW} = \{A_i \exp[(r_i - d_{iW})/B_i] + \phi g(r_i - d_{iW})\}n_{iW} - \omega g(r_i - d_{iW})(v_i \cdot t_{iW})t_{iW}$$

where n_{iW} and t_{iW} are the unit vectors perpendicular and tangential to the wall, respectively, relative to the pedestrian.

At each time step, the Social Force Model for pedestrian i receives as input the positions and velocities of all other agents, the relative locations of each wall or border segment in the scene, as well as its own desired velocity vector pointing towards its destination point and determines i's change in velocity with the acceleration term

$$\begin{aligned} m_i \frac{dv_i}{dt} &= f_{adapt} + f_{otherPeds} + f_{vehicles} + f_{borders} \\ &= m_i \frac{v_i^0(t)e_i^0(t) - v_i(t)}{\tau_i} + \sum_{j,j\neq i} f_{ij} + \sum_k f_{ik} + \sum_W f_{iW} \end{aligned}$$

3.2 Behaviour Trees

Behaviour trees are powerful tools that can be used to concisely model a wide range of decision making processes. One of their main advantages is that they are easily understandable and can be designed without extensive expert knowledge. While the trees can grow with as much complexity as the user desires, we have found through evaluation of our model that behaviour trees of a manageable size ($<$ 15 leaf nodes) are quite sufficient in modelling the non-trivial behaviours of pedestrians in traffic. Within our model, each pedestrian agent in each scenario is assigned a personalized behaviour tree to determine which maneuver to perform at each execution time step.

The behaviour trees used in the presented model are composed of four types of nodes: selectors, sequence, maneuvers, and conditions. In the context of our model, the leaf nodes of the tree are maneuvers and conditions and the internal nodes are selectors and sequences, otherwise known as composite nodes. Each internal node can have one or more children.

The basic process of traversing and evaluating a behaviour tree involves "ticking" the tree. When a behaviour tree is ticked, the tree is traversed in a depth-first fashion starting at the root. It is not necessary that every node is visited with each tick of the tree and the tick's path is dependent on the evaluation of the visited conditions and maneuvers. After a leaf node has been visited and evaluated, a status is returned back up the tree with the tick. Based on their type (selector or sequence), internal nodes use the returned status to determine the next path of the tick. Within the scope of our model, the valid statuses are SUCCESS, FAILURE, and RUNNING. In general, conditions return a status of SUCCESS or FAILURE and maneuvers return SUCCESS or RUNNING.

An example of a simple behaviour tree can be seen in Figure 3.2. We will explain in detail the process of ticking this example tree in Section 3.2.3.

3.2.1 Selector and Sequence Nodes

Selector nodes, denoted by a question mark [**?**], are essentially equivalent to a logical OR. When they are reached by the tick, they continue ticking their child nodes, in order of left to right, as long as they receive a status of FAILURE. If they receive a status of SUCCESS or RUNNING, then the selector node returns that same status to their parent node. If they tick through all their children and each child returns FAILURE, then the selector node returns a status of FAILURE to their parent. The idea of a selector node is to *select* one of its children to be executed. This is useful when the tree must select an action to take from an ordered list of actions with decreasing priorities.

Sequence nodes, as their same suggests, are used to execute a sequence of actions as long as each action executes successfully. They can be thought of as analogous to a logical AND. Sequence nodes are denoted by an arrow [$\rightarrow$]. When a sequence node is visited, it proceeds to tick each child in order until it receives a status of FAILURE at which point it returns FAILURE to its parent. If no children return FAILURE, it returns the tick to its parent node with the status of its last executed child, either SUCCESS or RUNNING.

Since the root node is internal, it will either be a sequence or selector node. As it does not have a parent node, the status that would normally be returned to its parent node is used as the return status of the entire tree. The leaf node that generated the status returned by the root node is selected to be executed. The trees used in our model are designed such that the selected leaf node will always be a maneuver node, which is the maneuver chosen to be performed by the pedestrian.

3.2.2 Maneuver and Condition Nodes

Each leaf node of the behaviour trees used in our model is either a condition node or a maneuver node. The trees of our model are intentionally designed in such a way that ensures a maneuver node is the last leaf node visited before returning from the tree for any given tick sequence. The last maneuver node visited describes the maneuver selected to ultimately be executed by the pedestrian.

Condition nodes are designed to return a binary status of either SUCCESS or FAILURE, which is then processed by their parent node. These nodes provide context and information about the current state of the environment, such as the state of a given pedestrian crossing signal, and are used to guide the tick to the most appropriate maneuver. The specific maneuver and condition nodes used in the model's behaviour trees are covered in detail in Section 4.3.

Throughout this thesis, maneuver nodes are visually represented by ellipses and condition nodes by diamonds.

3.2.3 Simple Behaviour Tree Example

We will illustrate the process of ticking a behaviour tree with a simple example for clarity. Suppose that a pedestrian within our model is following the behaviour tree shown in Figure 3.2. Note the shapes of the condition and maneuver leaf nodes. Suppose the pedestrian is currently waiting at a crosswalk entrance with the intention of crossing. They must consider the crossing signal state before making their crossing decision.

Let's first assume the crossing signal is red. The following steps walk through the tick's traversal of the tree for the given context.

1. The tick's path starts at the root selector node and travels down to the leftmost leaf node in a depth-first fashion.

2. From the environment's state, this condition node determines whether the pedestrian crossing signal for the relevant crosswalk is green. Since it is not green, the node returns the tick back up to its parent with a status of FAILURE.

3. We know that sequence nodes, [$\rightarrow$], tick their children successively *until* they receive a FAILURE status, so the tick then returns to the root node with the same FAILURE status.

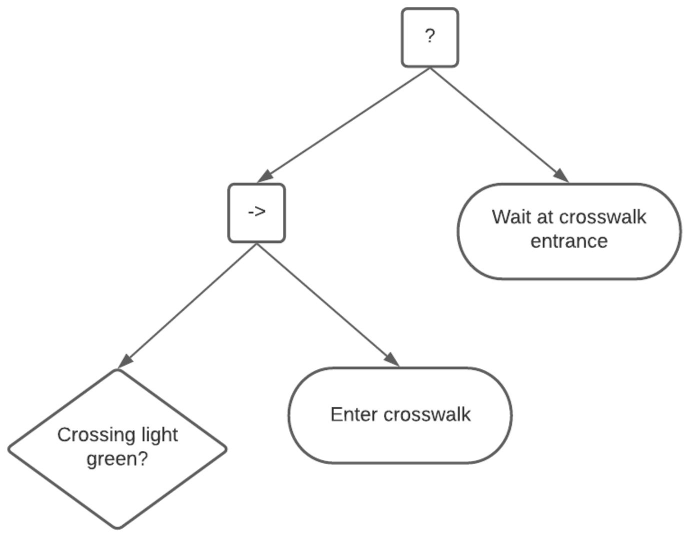


Figure 3.2: Example of a simple behaviour tree used to determine the appropriate maneuver for a pedestrian waiting to enter and cross a signalized crosswalk.

4. Selector nodes, [**?**], on the other hand, tick their child nodes in order *as long as* they receive a status of FAILURE. Therefore, the root will direct the tick to its right child, a maneuver leaf node.

5. In our model, we designed maneuvers to always be immediately executable by the subsequent layers so the tick returns back to the parent node (the root), this time with a status of SUCCESS.

6. Finally, the tick returns from the entire tree with the maneuver **Wait at crosswalk entrance** since the selector at the root received a status of SUCCESS (and also ran out of unvisited child nodes). This maneuver is the one selected to be executed by the pedestrian. It is important to note that the Behaviour Layer does not actually execute the maneuver. The selected maneuver is passed to the Maneuver layer to plan the resulting trajectory and finally the Motion Planner Layer handles the execution of the pedestrian's decision.

Now suppose the crossing signal is green. The condition node in Step 2 would evaluate to true and return SUCCESS to its parent, a sequence node, which would direct the tick to the maneuver **Enter crosswalk**. Because the parent sequence node has run out of children, it returns the SUCCESS status to the root selector node and causes the tick to return from the entire tree with the maneuver that was last visited. Therefore, the pedestrian decides to enter the crosswalk.

3.3 GeoScenario

The scenarios run by the model are defined and designed in GeoScenario [62], a Domain-Specific Language built on top of the Open Street Map (OSM) standard. First, we must define how a scenario is expressed within GeoScenario. Scenarios are composed of a set of elements such as a road network, vehicle and pedestrian agents, agent start positions and goals (destinations), paths, triggers, and actions, though not all of these are necessary for our model. The scenario is formed by representing the included elements in a formal language following the OSM standard.

GeoScenario adopts two core building blocks from OSM: the node and the way. A node is an elemental point within the scenario that can be used to either represent a single entity, such as a pedestrian agent or an agent's destination point, or be used to build composite elements, such as a way that may be part of a road boundary. A way is simply an ordered sequence of nodes that can be used to compose, for example, a lane boundary or even define the path of an agent in the scenario. A way can either be open or closed based on its intended use. A closed way can be used to define an area of particular interest, such as a traffic island. Nodes and ways are used to construct the components of each scenario within GeoScenario.

While nodes and ways are low-level elemental components, they can be augmented with additional information with the use of tags. Tags have a *key-value* format where the key is selected from a known list of valid keys and the value should be within acceptable range or set of options for the given key. Tags can add information valuable to the scenario definition such as each agent's type (vehicle or pedestrian) or the underlying map structure used by the scenario. For example, the specific key to define the type of an agent is gs, so the key-value tag $\langle gs, pedestrian\rangle$ is assigned to each pedestrian agent within the scenario.

Each scenario is required to specify a map file to define the underlying road structure. Like scenario files, map files also follow the OSM standard and are constructed with a network of nodes and ways. Additional information on the map files is provided in Section 3.4.

3.4 Lanelet2 Maps

The map files that form the underlying road structure of scenarios are constructed following the standard Lanelet2 framework [59]. At their core, Lanelet2 maps are composed of lanelets, areas, and regulatory elements. Following the OSM standard from Section 3.3, lanelets are defined by their borders, made up of nodes and ways. They define drivable and walkable sections of the traffic environment, such the road and sidewalk. However, a lanelet is just one elemental section of the roadway, so sequences of lanelets joined by their borders' end points are formed to construct the entire road structure of the environment. For example, a single lane of a roadway is represented by a sequence of connected lanelets, with each following lanelet sharing their end points with its successor. Lanelets can also have tags assigned to them that augment the relevant information associated with that particular section of the road or sidewalk. Lanelet tags follow the same format as scenario tags in GeoScenario. Common lanelet tags include intended users of the lanelet (pedestrian, vehicles, etc.), whether the lanelet is bidirectional, and the speed limit within the defined section of roadway.

Less commonly used than lanelets, areas are closed polygons with their borders defined by three or more points. Areas can take on any shape and are used to represent a space that can be occupied by road users but have no inherent direction of travel, such as a traffic island or a parking space. In the data sets used in our evaluation, areas are only used to represent pedestrian traffic islands.

Regulatory elements provide the designer with a way to express traffic rules. Regulatory elements generally represent a physical (or at least observable) element of the environment such as a stop sign or traffic light, but users are free to extend their use to express any traffic rules required in a specific scenario. For our purposes, regulatory elements are used to represent stop signs, traffic lights, and pedestrian crossing signals. Specifics on how regulatory elements are used within our model are provided in Section 4.2.1.

Chapter 4

Hierarchical Pedestrian Behaviour Model

4.1 Model Design

This section discusses the design goals of a practical agent-based microscopic pedestrian simulation model and how our presented model addresses and achieves each one.

Facilitation of Scenario Creation (G1)

An essential component of scenario-based testing is a clear, straight-forward, and easily explainable scenario creation process. With a highly customizable model, the necessity of this requirement is even stronger as the tester is provided with a wide range of design choices. To address this, we represent scenarios in the Domain Specific Language, GeoScenario, and employ the Open Street Map scenario editor tool, JOSM [2], to create and edit scenarios. JOSM provides a drag-and-drop style interface to edit Open Street Map files by allowing users to quickly add, move, and tag nodes and ways (Figure 4.1). Recall from Section 3.3, nodes and ways are the scenario building blocks and are used to construct many different elements of the scenario including dynamic agents, static objects, and even entire Lanelet2 maps. Testers are able to create and edit tags for one or multiple elements at a time within JOSM, allowing them to set a pedestrian agent's name, type, goal point, and personal behaviour tree file. Customizing behaviours with behaviour trees is discussed in the following section.

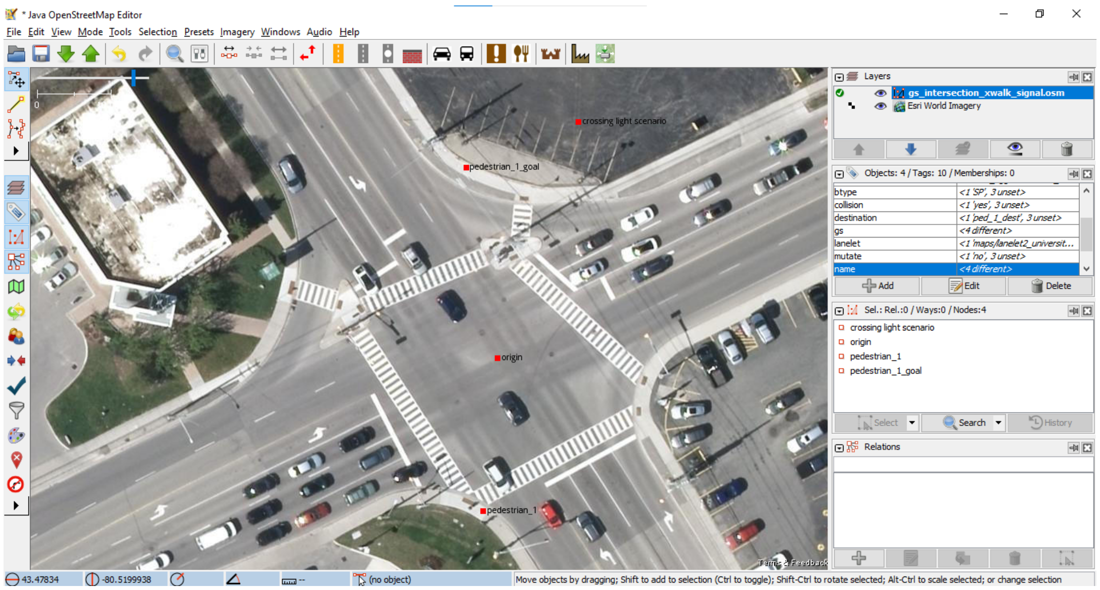
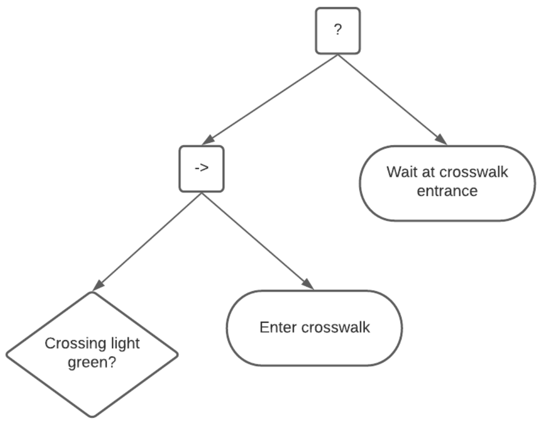


Figure 4.1: Using the Java Open Street Map editing tool to edit a scenario.

Customizable Pedestrian Behaviour (G2)

Pedestrians in the real world display a spectrum of behaviours, so it is not reasonable for our model to assume each pedestrian agent will act the same and make the same decisions at all times, even given identical contexts. Behaviour trees provide the tester with a fine-level of control over how each pedestrian behaves and are an integral part of each scenario. Each pedestrian is assigned a behaviour tree file at scenario creation, though multiple pedestrians may be assigned the same file. This file is a text representation of the behaviour tree that dictates the pedestrian's decisions throughout their journey. We will revisit the behaviour tree from the example in Section 3.2. Figure 4.2 displays the text file representation of the example behaviour tree, *enter_xwalk_on_green*, where each line of text signifies a different node in the tree (except for the title on the first line). Note the parameters passed to each maneuver and condition node and how the indentation corresponds to the tree's structure. The details of these parameters is discussed further in Section 4.3.

As an example of how different behaviour trees can influence different decisions, we consider the case of a pedestrian approaching a crosswalk that they intend to cross, where the crossing signal displays a countdown of 7 seconds to the stopping "red hand" state. An able-bodied pedestrian in a hurry might choose to enter the crosswalk and increase their walking speed as they cross, whereas an elderly pedestrian may opt to wait for the



(a) Tree representation

```
behaviortree enter_xwalk_on_green:
    ?
        ->
            condition c_ped_greenlight ( pedestrian_light_green() )
            maneuver m_enter_crosswalk ( MEnterCrosswalkConfig() )
        maneuver m_wait_at_crosswalk ( MWaitAtCrosswalkConfig() )
```

(b) Text representation

Figure 4.2: Tree (a) and text (b) representations of the same *enter_xwalk_on_green* behaviour tree

next green "walk" signal to start crossing. These two decisions can both be modelled with either completely different trees or the same tree with different parameters that influence the crossing decision. This exact scenario and others are further explored in the evaluation section 5.2.2.

Dynamic Interactions Between Agents (G3)

The interactions involving pedestrian agents within a scenario is largely handled by the Social Force Model component of the motion planner layer. These interactions can be between pedestrians or between a pedestrian and a vehicle. There is a collection of repelling physical forces between each pedestrian and all the other dynamic agents that makes up an essential component in the SFM acceleration equation. These forces ensure that agents maintain a comfortable distance from and do not run into each other while ensuring pedestrians produce natural movements when close to other agents. The Social Force Model has been proven to effectively produce a realistic representation of human behaviour and interactions between human agents. This is also supported by our observations during the model's evaluation process.

The handling of dynamic interactions is also taken on by the behaviour tree component. A wide range of interaction behaviours can be modelled based on the specific condition and maneuver nodes within the tree. For example, pedestrians may employ a gap-acceptance approach to decide whether to enter a crosswalk. Condition nodes can check the speeds and positions of the relevant vehicles and the tree would make a crossing decision depending on the pedestrian's acceptance parameters. If conditions change throughout the interaction, behaviour trees handle the discrete switching behaviour which allows the pedestrian to dynamically change their decision based on the new information. Behaviour trees provide an explicit representation of dynamic interactions between agents which proves useful in scenario design.

Realistic Human Movements and Decisions (G4)

A practical pedestrian behaviour model needs to be able to simulate realistic movements and rational decision-making processes. We evaluate this requirement on our model against two naturalistic data sets with different road structures. Our hierarchical approach of handling high-level decisions with customizable behaviour trees whose outputs inform low-level trajectory movements is shown to be effective in producing realistic movements and decisions that map to real-world scenarios.

4.2 Model Architecture

In this section, we will discuss the structure of the presented model. Since we employ an agent-based approach, each pedestrian within a scenario contains an individual instance of the model. The model receives both static environmental inputs upon startup of the scenario that remain constant throughout execution (e.g. map structure, static objects) and dynamic inputs from changing aspects of the scenario (e.g. other agents, traffic light states). At each scenario time step, each pedestrian perceives the environment around them and forms a simplified state representation of it, which acts as the dynamic input to the model. This representation consists of state vectors for all agents, containing positional, velocity, acceleration, and heading information as well as traffic light state information for both vehicle traffic lights and pedestrian crossing signals.

The architecture is composed of three layers: the Behaviour layer, Maneuver layer, and Motion Planner layer. As an overview, the Behaviour layer receives the environment state representation and decides on an appropriate maneuver to execute. This maneuver is then passed to the Maneuver layer, which plans how best to execute the selected maneuver and forms instructions on how to adapt the current trajectory in the form of a vector containing the pedestrian's updated waypoint, direction vector, and desired speed, to pass to the Motion Planner layer. When the Motion Planner layer receives these instructions, it feeds the passed vector into the Social Force Model to determine the state information of the pedestrian for the next time step. Finally, this state information is updated and reflected in the environment. This process flow is visualized in Figure 4.3.

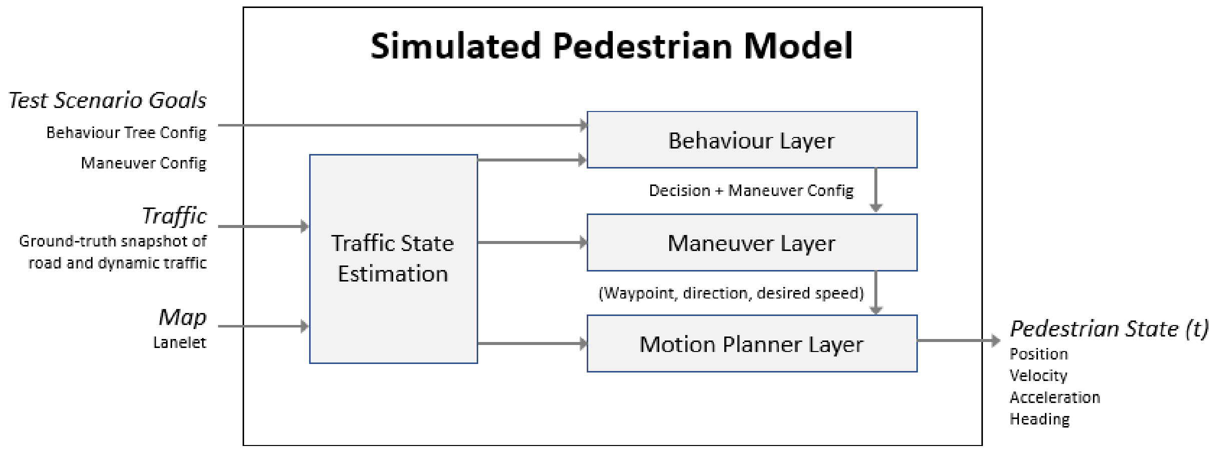


Figure 4.3: Model diagram with process and information flow.

4.2.1 World Model and Pedestrian Representation

The simulation traffic environment is represented in the three dimensional Cartesian coordinate frame and all computations for pedestrian movements are calculated in this frame. To reduce computational complexity, the z-coordinate for agent positioning is taken to be 0 so all calculations are reduced to be within the 2D xy-plane. The physical traffic structure of the world is represented by Lanelet2 compatible map files.

Lanelet2 Maps

As discussed in Section 3.4, our model relies on the Lanelet2 libraries to represent the traffic environment with OSM map files. These files must follow the Lanelet2 standards in order to be properly processed by the model.

To ensure compatibility between our model and Lanelet2 standards, we include pedestrian-specific lanelets and areas, with their own set of requirements, to the map file. In this section, we discuss these additional requirements we have defined. Any potential walking area in the scene must be represented and covered by a bidirectional pedestrian lanelet or area. This includes all sidewalks, crosswalks, and any pedestrian-specific areas, such as a traffic island. Intuitively, they are required to be bidirectional because, unlike vehicles, pedestrians can travel in any direction within their designated walking areas. In Lanelet2, we refer to the specific type of walking area (e.g. crosswalk, sidewalk, traffic island) as the lanelet or area's *sub-type*. Any Lanelet2 map file intended to be used with the model must follow the requirements listed below in order to ensure compatibility. The following lists define requirements for pedestrian lanelets, areas, and the pedestrian crossing signal regulatory element.

Lanelets

1. Any walkable space must be covered by one or more lanelet or area.

2. If a pedestrian-specific space, such as a sidewalk, has an accessible crosswalk (i.e. the pedestrian does not need to enter the roadway in order to enter the crosswalk), there must be a sequence of connected lanelets from that space to the crosswalk. In this context, two lanelets are *connected* if they share end points (nodes) such that the left border of the preceding lanelet shares an end point with the left border of the succeeding lanelet and likewise for the right border. This requirement is to ensure pedestrians can find valid routes of sequential lanelets from their position to all crosswalks accessible to them.

3. Any two points on the same segment of sidewalk must have a connected sequence of lanelets between them. Two connected lanelets composing a portion of sidewalk must share end points; however, the left border of one lanelet does not necessarily need to share an end point with the left border of the other lanelet. It may share an end point with the right border instead. This loosens the definition of *connected* lanelets only when all lanelets involved are part of the sidewalk. A single segment of crosswalk is defined as a continuous stretch of sidewalk such that a pedestrian is not required to leave the sidewalk to travel between any two points.

Areas

1. At minimum, an area must have a closed outer border defined by a set of points in a clockwise orientation.

2. In order for a crosswalk (or any other lanelet) to be *accessible* from an area, one of the crosswalk's pairs of end points (either both start points or both end points) must be shared with any two points of the area's outer border.

Pedestrian Crossing Signal

1. Each pedestrian crossing signal must have a corresponding regulatory element defined in the underlying map file.

2. A crosswalk is not required to have a crossing signal. If a crosswalk does not have a signal, the pedestrian is assumed to have right-of-way over vehicles.

3. Each crossing signal element must be tagged with a sequence of states and durations. The states are a list of characters that correspond to light states. The standard states are **G** for green, **R** for red, and **Y** for yellow. The list of durations define the number of seconds each corresponding state is active. For example, given the states $\langle G, Y, R\rangle$ and the durations $\langle 15, 5, 20\rangle$, the crossing signal will be green for 15 seconds, then yellow for 5 seconds, and finally red for 20 seconds before looping back to the beginning state of green for 15 seconds and so on.

Dynamic Elements

The world is also represented by its changing elements in the form of dynamic agents and states. Both dynamic and static elements are contained within the simulation environment. Static elements make up unchanging aspects of the world such as the physical road geometry, traffic lights, and road signs. On the other hand, dynamic elements can change throughout the execution of the scenario. Within our model's scope, these are primarily composed of dynamic road users and traffic light/pedestrian crossing signal states.

There exist two types of pedestrian agents: Trajectory Pedestrians (TP) and Simulated Pedestrians (SP). Trajectory Pedestrians follow a predefined sequence of trajectory points with preset time intervals and do not use our model to drive their motion. Simulated pedestrians do apply the presented model to determine their movements and are the focus of discussion of this thesis. Unless otherwise noted, "model pedestrian" refers to an agent of the SP type. All agents within a scenario share a common format to represent their current state at time t. Agents' state information contains two dimensional positional, velocity, and acceleration components as well as heading in the vector $|(x, \dot{x}, \ddot{x}, y, \dot{y}, \ddot{y}, \theta)|_t$. Vehicle agents have similar type options: Trajectory Vehicles (TV) and Simulated Driver

Vehicles (SDV). The SDV model is presented by Queiroz et al. (under submission). The state vector format is shared between all types of pedestrian and vehicle agents.

Unlike for vehicles, where one or more could be controlled by the automated driving system under test and are marked as "ego", there is no concept of an "ego" pedestrian. All model pedestrians are represented the same way and have access to all the same information from the environment. That being said, our method for the evaluation of the model employs a process where there is only one SP pedestrian per scenario who is, indeed, the main focus of these specific scenarios.

4.2.2 Traffic State Estimation and Pedestrian Motion

Every decision a pedestrian makes in traffic is dependent on the state of the environment around them. In order to make sense of the perceived world, pedestrians within the model form a simplified estimation of the traffic environment composed of properties and characteristics of the current traffic state at a given time. The traffic state estimation is accessible to the pedestrian at any layer of execution (Behaviour, Maneuver, and Motion Planner) and consists of the static lanelet map and the current states of the pedestrian crossing signals and all other agents in the scene. Pedestrians technically have access to the state of the vehicle traffic lights, although it is not taken into consideration for any decision by the model in its current implementation. Perhaps in future iterations of the model, the pedestrian may consider the traffic light states while deciding which crosswalk to take when all the crossing signals are red. The traffic lights may indicate which crosswalk will have a green walk signal next. However, this functionality is not yet implemented.

The overall pedestrian walking mission is defined by a starting point and ending point (destination) set in the Cartesian xy-plane. The SP agent does not plan the path for their entire journey when they are initialized upon startup of the scenario. Instead, the pedestrian plans one *segment* of their journey towards their destination at a time. These segments are defined by an ordered sequence of connected lanelets or areas (or a combination of both) starting with the space the pedestrian currently occupies. Once the pedestrian has traversed their current planned segment, the planning process is run again and a new journey segment is determined. The goal is to bring the pedestrian closer to their destination with each planned segment. Given the intended and tested traffic environments available to us, each segment either ends at the agent's destination or at a crosswalk the pedestrian must cross to reach their destination. For the latter case, the lanelet representing the crosswalk is not included in the segment, but will be a part of the next planned segment. This is working under the assumption that if the pedestrian's destination is not on the same sidewalk as the pedestrian, then there exists a sequence of crosswalks and other valid walking areas that will lead to the destination. We refer to the pedestrian's current journey segment as their *local path*.

The concept of local paths also introduces the idea of intermediate waypoints. Waypoints are required for the Social Force Model calculations in the Motion Planner layer in order for the pedestrian to travel their local path. The pedestrian determines one waypoint per planned local path and places it at the midpoint of the two endpoints of the last lanelet in the path. This provides a temporary intermediate goal point for the pedestrian while travelling their current local path. This waypoint is one of the few inputs to the Motion Planner layer required to determine the pedestrian's next state vector. In general, the attracting force of the SFM calculation is towards the pedestrian's waypoint, meaning the pedestrian's desired velocity vector, v, points to it. However, in some cases, this may be overridden by the Maneuver layer in which a different unit direction vector, d, may be passed to the SFM changing the pedestrian's desired velocity to be in the direction of d with the magnitude of v.

Throughout execution of the scenario, model pedestrians continuously iterate through the process of dynamically determining and traversing local paths until they reach their destination point.

4.2.3 Behaviour Layer

The Behaviour layer is tasked with determining an appropriate maneuver to execute given the current environmental context. It contains the pedestrian's specific assigned behaviour tree and consumes the current traffic state estimation as well as configuration parameters for the tree and each possible maneuver. The output of this layer is a selected maneuver with its specific configuration parameters, which is subsequently passed to the Maneuver layer.

At a higher decision making level, behaviour trees are designed to check certain conditions from the traffic state estimation and select a maneuver based on the particular set of conditions. At creation, each pedestrian is assigned one behaviour tree to determine which maneuvers can possibly be executed throughout their journey. This tree may have sub-trees in its structure, but the pedestrian still has a single main "root" behaviour tree. At each simulation cycle, the Behaviour layer ticks each model pedestrian's respective behaviour tree and selects an appropriate maneuver for each to execute. The behaviour trees used in our model are designed to always return a valid maneuver with a status of SUCCESS at each tick.

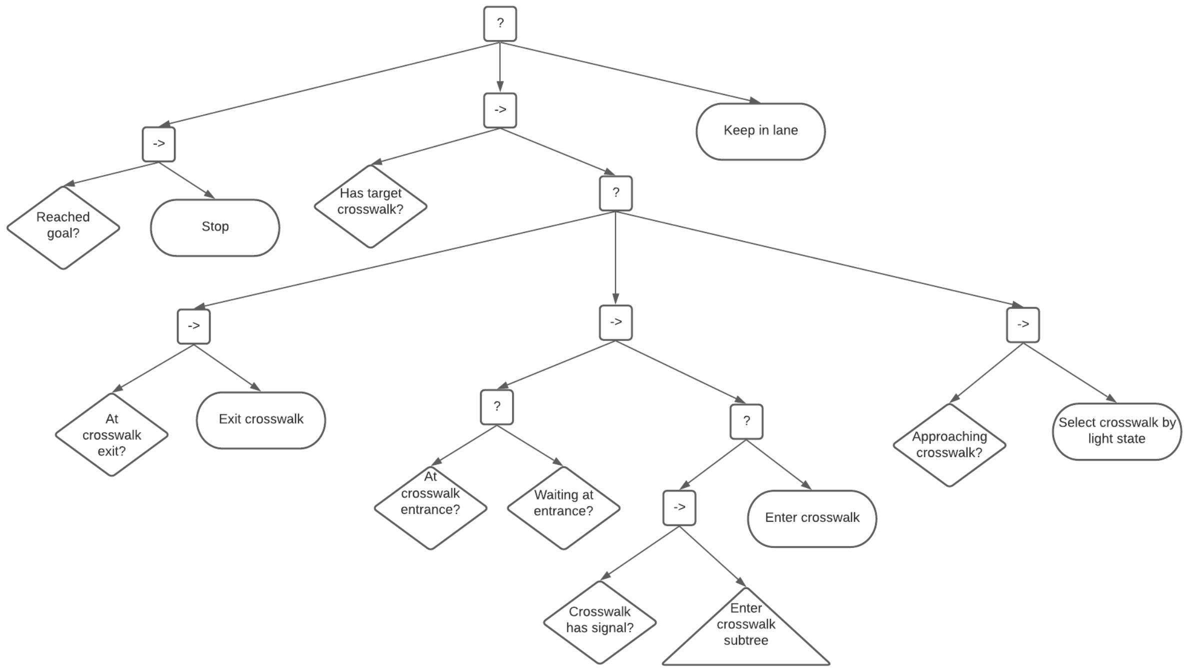


Figure 4.4: Default behaviour tree assigned to model pedestrians.

Sub-trees

Behaviour trees support the use of sub-trees within their structure. This provides the additional benefit of modularity and avoids trees growing too large in their complexity. A sub-tree is itself a separate behaviour tree that can simply be "plugged into" another tree, always at a leaf node, to extend the main tree's structure. As an example, Figure 4.4 shows a single sub-tree, labelled **Enter Crosswalk Sub-tree** and represented by a triangle, incorporated into the pedestrian's main behaviour tree in substitute of a leaf node. When the tick reaches a sub-tree node, it immediately begins to tick the sub-tree from its root. Sub-trees may not necessarily always output a maneuver and should be viewed as simply an extension of their base tree.

In this model, we use sub-trees to generate different variants of behaviour to control how different pedestrians may react in the same situation. Specifically, we have defined three levels of aggressiveness (Low, Medium, High) to describe pedestrians' varying decision processes when deciding the time at which they will enter their target crosswalk given the current crossing signal state. A pedestrian's *target* crosswalk is the crosswalk they have already decided to take as part of their local path. Each of these three levels of aggressiveness has a corresponding sub-tree that can be substituted in place of the sub-tree node to provide the pedestrian with a different decision-making process in the given situation. In order for the selected sub-tree to be invoked, the pedestrian must be at the entrance of their target crosswalk (within a threshold distance of the entrance point) and the target crosswalk must have a crossing signal. Below are tree representations of the three sub-trees, each corresponding to a defined level of aggressiveness with explanations on their respective decision-making processes.

Low - Only enter on green light (Figure 4.5)
At the lowest level of aggressiveness, the pedestrian will only enter the crosswalk if the corresponding crossing signal is green. If the signal is not green, the pedestrian will wait at the crosswalk entrance until a change in state of the signal.

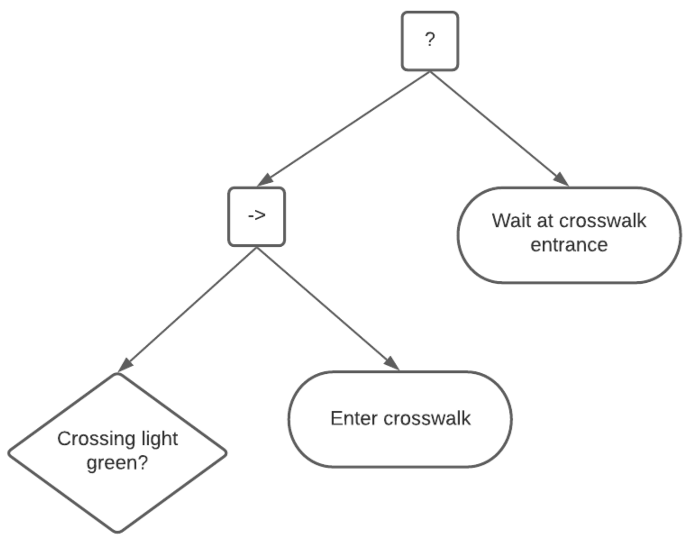


Figure 4.5: Enter Crosswalk sub-tree - Aggressiveness Level 1

Medium - Enter on yellow if pedestrian can cross before red (Figure 4.6)
The medium level of aggressiveness includes the "entering logic" of the low level (i.e. the pedestrian will enter the crosswalk on a green signal), but it adds extra conditions and maneuvers in the case of a yellow signal. Note that we are assuming the yellow state includes a countdown until the next red state that is made available to the pedestrian. If the crossing signal is yellow, the pedestrian must estimate if they can cross the crosswalk before the signal turns red within a reasonable margin. In order to make this estimation,

the pedestrian must take into account the time in seconds to the next red signal t_{red}, their own walking speed, a realistic increase in speed if necessary, and the distance they must travel to finish crossing. All of these factors are considered in the below calculation of Boolean $x_{\text{can\_cross}}$ to determine whether crossing before red is possible.

$$x_{\text{canCross}} = \begin{cases} \textbf{true} & \text{if } t_{\text{cross}} \leq t_{\text{red}} \\ \textbf{false} & \text{otherwise} \end{cases} \tag{4.1}$$

where

$$t_{\text{cross}} = \frac{d_{\text{entry}} + d_{\text{xwalk}} - d_{\text{fromExit}}}{s * (1 + k)}$$

In the t_{cross} equation above, the numerator represents the distance the pedestrian must travel to be considered finished crossing the crosswalk. The three components, d_{entry}, d_{walk}, and d_{fromExit} are the distances, respectively, from the pedestrian's current position to the crosswalk entrance point, from the crosswalk entrance to its exit point, and a configurable distance from the crosswalk exit the pedestrian must reach to be considered to have sufficiently crossed the crosswalk. This third distance is necessary to cover the observed case where pedestrians will start crossing a crosswalk knowing they cannot fully cross before the red state but they can at least travel an acceptable distance before the red. This risk-taking behaviour was frequently observed in our evaluation data sets so a configurable parameter was added to allow testers to set this distance. The denominator of t_{cross} is the product of the default desired speed of the pedestrian, s, and a factor by which the pedestrian can reasonably increase their speed in order to finish crossing before the next red signal, k. Note, $k = 1$ results in a 100% increase in walking speed.

High - Enter crosswalk unless oncoming vehicle presents danger (Figure 4.7)
The highest level of aggressiveness is reserved for risk-taking pedestrians that disregard crossing signals and simply try to optimize the time and distance travelled of their journey. As explained before, a higher level of aggressiveness will always enter the crosswalk given the same conditions a lower level also entered on. Thus, a high-aggressiveness pedestrian will enter the crosswalk on a yellow when they judge that they can finish crossing before the next red and always on a green signal. Additionally, if the signal is red or a yellow where they cannot cross before the next red, then the high-aggressiveness pedestrian checks for oncoming vehicles. If there is an oncoming vehicle that presents a danger to them if they cross (as described by the **Vehicle approaching crosswalk** condition), the pedestrian will not enter the crosswalk; otherwise, they will.

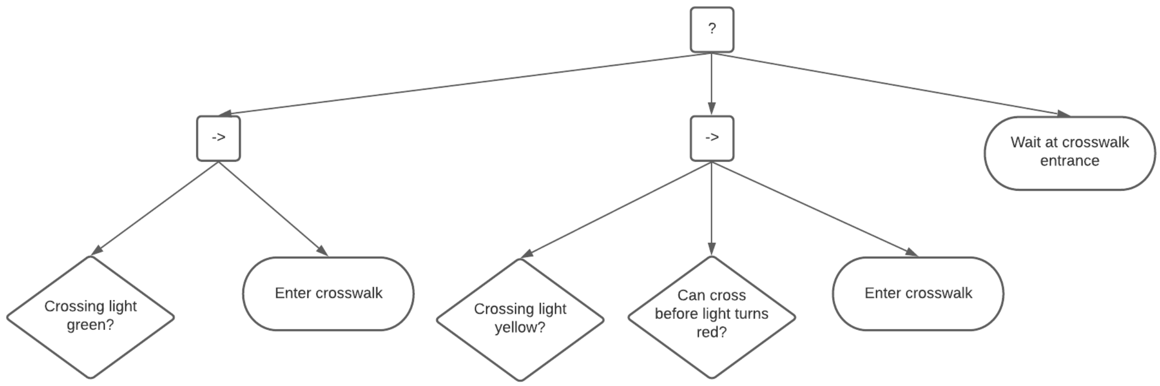


Figure 4.6: Enter Crosswalk sub-tree - Aggressiveness Level 2

The sub-trees described above provide the user of the model with customization freedom and a higher degree of control over the pedestrians' behaviour. A level of aggressiveness is assigned to each pedestrian at creation which determines the sub-tree that extends their main behaviour tree (Figure 4.4) at the leaf node labelled **Enter Crosswalk Sub-tree**.

4.2.4 Maneuver Layer

Given a valid maneuver from the Behaviour layer, the Maneuver layer translates the selected maneuver into low-level instructions on how to adapt the agent's trajectory. These instructions have three components: a unit direction vector d, a waypoint w, and a desired speed s. These three components are passed to the Motion Planner layer and are essential for the Social Force Model calculation. This section defines the set of available maneuvers and explores each one's process of determining the trajectory adapting instructions.

In order to model the wide range of complex pedestrian behaviours in the real world, a sufficiently complete list of pedestrian maneuvers is valuable to have at a tester's disposal. We construct such a list to draw from when designing each behaviour tree. The complete list of the maneuvers implemented in the model is presented below with brief explanations of their purpose and application. As different pedestrians display varying tendencies and behaviours, many of the below maneuvers have additional parameters that can be adjusted to fit the desired walking style.

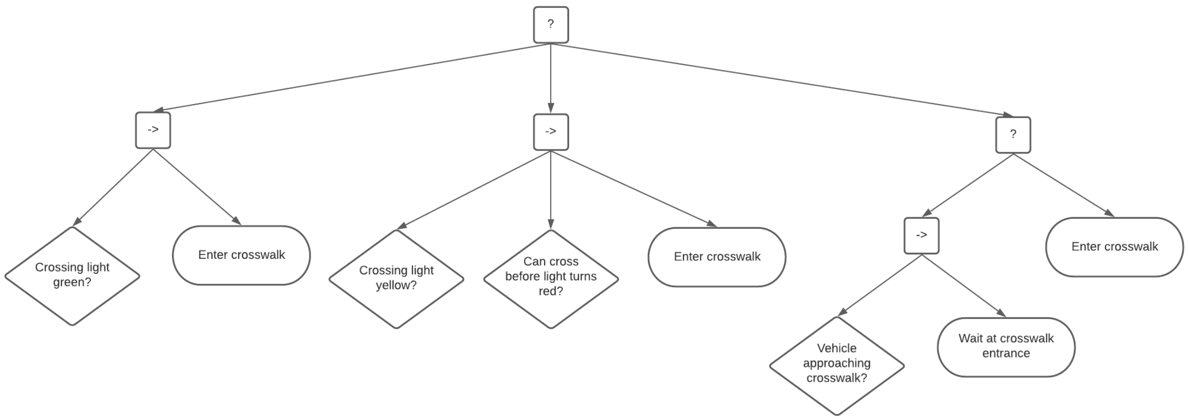


Figure 4.7: Enter Crosswalk sub-tree - Aggressiveness Level 3

Maneuvers

Keep in Lane

Default maneuver. The pedestrian walks towards their current waypoint following the borders of the sidewalk or crosswalk lanelet they occupy. Both the pedestrian's waypoint and desired speed do not change during this maneuver. The direction vector d, however, depends on the geometry of their current lanelet and the relative position of the waypoint. If the waypoint is within the line-of-sight of the pedestrian (i.e. a straight line from the pedestrian's position to the waypoint does not intersect any lanelet borders of their local path), then the direction vector simply points to the waypoint. Otherwise, the pedestrian will select a direction vector parallel to either the left or right border of their current lanelet to simulate "following" the selected border.

In order to select which lanelet border to follow, the agent considers their current waypoint and their currently occupied lanelet's exit point. Since their current lanelet is part of an ordered sequence of lanelets (the pedestrian's local path), there is an inherent order to the lanelet borders' points. The lanelet's exit point is defined as the midpoint between the last point of the left border and the last point of the right border. The agent then draws a line intersecting their position point and their current lanelet's exit point. If their waypoint w lies to the left of this line, the pedestrian selects the left border to follow and vice versa if w lies to the right. Generally, a lanelet's left and right borders are reasonably parallel making this border selection arbitrary; however, we present a simple method to account for unusual sidewalk or other walking spaces' geometries.

Stop
The pedestrian's desired speed is set to zero to allow the pedestrian to decelerate to a stop at a natural rate as determined by the Motion Planning layer. The waypoint and direction vector are not changed by this maneuver.

Enter Crosswalk
If this maneuver has been selected, the pedestrian has made the decision to enter their target crosswalk and is within a given threshold distance from the crosswalk's entrance point. A crosswalk's entrance point is defined in a similar way to a lanelet's exit point from the **Keep in Lane** maneuver, except using the first points of the left and right borders. The current waypoint is updated to be the exit point of the crosswalk. The pedestrian's desired speed is set as the agent's default desired speed determined at pedestrian creation. The direction vector is set to point at the updated waypoint.

Exit Crosswalk
The pedestrian has almost finished crossing their target crosswalk (i.e. within a threshold distance of its exit point) and must determine their next action. This maneuver invokes the function to plan the pedestrian's next local path. The waypoint is then updated to be the endpoint of the last lanelet in the updated path. The direction vector and desired speed are not changed.

Wait at Crosswalk
The pedestrian has selected a target crosswalk but cannot enter it yet due to the crossing signal state. The pedestrian remains within a threshold distance from the entrance until the pedestrian's behaviour tree results in a different maneuver. In the usual case, the next maneuver is **Enter Crosswalk** and is executed when the target crosswalk's signal turns to green. The desired speed is 0 and both the waypoint and direction vector are unchanged.

Increase Walking Speed
The pedestrian's desired walking speed is multiplied by a constant specified by a configuration parameter set in the behaviour tree's node for this maneuver. The waypoint and direction vector are unchanged.

Select Crosswalk by Light State
The pedestrian is approaching their target crosswalk and must re-evaluate their crosswalk selection based on the current state of the crossing signals. This maneuver invokes a function to select a new target crosswalk from the list of accessible crosswalks. Details of this function are given in Section 4.2.4. The waypoint is updated to be the entrance point of the selected crosswalk. The desired speed and direction are not changed.

Return to Crosswalk Entrance
After a pedestrian has entered the crosswalk, they may choose to return to the entrance

if remaining in the crosswalk or continuing to cross endangers their safety. They also may simply change their mind about needing to cross. In this case, the waypoint is updated to be the entrance point of the crosswalk they are currently crossing, the direction vector points to the new waypoint and the desired speed is unchanged. This maneuver is not included in the default behaviour tree, but it provides the tester with the option to configure unusual but completely plausible behaviours.

The output of the Maneuver layer, (w, d, s), is passed to the Motion Planner layer to determine the change in the pedestrian's state vector.

Conditions

Aside from maneuvers, behaviour tree leaf nodes can also be condition nodes. These Boolean nodes are evaluated and a status of SUCCESS is returned if the given condition evaluates to *true* and FAILURE if it evaluates to *false*. Within the scope of our model, condition nodes never return a RUNNING status. Below are examples of conditions used in our pedestrian behaviour trees. While we have found, through testing, that our set of conditions is sufficient in accounting for the dynamic nature of the test data sets, it is difficult to claim that a list of conditions is complete in this context since they are dependent on multiple factors such as available maneuvers, specific model implementation details, and physical structure of the environment.

Reached goal
Returns true if pedestrian is within a configurable threshold Euclidean distance from their goal point. Otherwise returns false.

Has target crosswalk
Returns true if pedestrian has selected a crosswalk to cross next, regardless of the target crosswalk's signal state or whether it even has a crossing signal. This condition will return true following execution of the target crosswalk selection function referenced in maneuver **Select Crosswalk by Light State**.

This condition will return false if there are no crosswalks that are advantageous for the pedestrian to take to bring them closer to their destination. In this case, the destination point is required to be within the same sidewalk segment as the pedestrian.

Approaching target crosswalk
Returns true if the pedestrian has selected a target crosswalk and is within a configurable threshold distance from its entrance point. Otherwise returns false.

At target crosswalk entrance
Returns true if the pedestrian has selected a target crosswalk and is within a configurable

threshold distance from its entrance point, otherwise returns false. Note that this condition is almost identical to **Approaching target crosswalk**; however, the threshold distance of this condition is smaller and they are differentiated by their roles in the behaviour tree shown in Figure 4.4.

At target crosswalk exit
Returns true if pedestrian has already entered their target crosswalk (i.e. executed the **Enter Crosswalk** maneuver) and is within a threshold distance from the exit point. Otherwise, returns false.

Waiting at target crosswalk entrance
Returns true if the condition **At target crosswalk entrance** is true but they cannot enter the crosswalk yet due to the crossing signal state. Otherwise, returns false.

Target crosswalk has signal
Returns true if there exists a crossing signal corresponding to the pedestrian's target crosswalk. Otherwise, returns false. This condition is dependent on the physical structure of the environment and does not change throughout execution of the scenario.

Crossing signal is green/red/yellow
This represents three distinct conditions that are written together for simplicity. Returns true if the crossing signal state is green/red/yellow respectively and otherwise, returns false. Note that, for our purposes, the yellow state represents a displayed countdown to the red state available to all pedestrians.

Can cross before crossing light turns red
Returns true if the crossing signal state is not red and the pedestrian judges that they can complete crossing their target crosswalk before the next red state. Otherwise, returns false. This judgement is made by considering their current desired speed, a factor by which they can realistically increase their speed, the time to the next red state (given by the yellow state), and the distance the pedestrian must travel to "finish" crossing the crosswalk from their current position. "Finish" is emphasized here because the scenario designer may choose to set a distance value that determines what distance from the crosswalk exit must be reached by the pedestrian for the crosswalk to be considered fully crossed. This is set through a configurable parameter in the behaviour tree node corresponding to this condition. For example, some pedestrians may choose to enter a crosswalk on a yellow state accepting that they can reach a point 1 metre from the exit before the red state. The conditions is determined by Equation 4.1.

Vehicle approaching crosswalk
Returns true if there is a vehicle with a trajectory that will intersect the pedestrian's target crosswalk before the pedestrian is able to cross. For the specific configuration of our model,

this condition is used in the behaviour sub-tree defining pedestrians with a high level of aggressiveness (Section 4.2.3) who may choose to jaywalk across the crosswalk on a red crossing signal. The condition is checked when the pedestrian faces a red crossing signal and is deciding whether or not to enter the crosswalk. Given this context, any approaching vehicle that may intersect with the crosswalk poses a threat to the pedestrian's safety.

The implementation of this condition can have varying levels of sophistication depending on what is deemed sufficient for the purposes of the model. For example, this condition could simply check if an vehicle will intersect the target crosswalk while the pedestrian occupies it (assuming both the vehicle and pedestrian proceed with their paths without stopping). Alternatively, layers of complexity may be added to improve the condition's accuracy regarding whether or not an approaching vehicle will have an effect on the pedestrian's decision to cross. Considering the rarity of the behaviour and context in which this condition is checked, we opted to implement a relatively primitive method in which the condition evaluates to true if there is any vehicle with a trajectory that will intersect the target crosswalk before the pedestrian can fully cross and false otherwise.

Local Path Planning Method

In this section, we discuss the method and logic behind planning the pedestrian's local path referenced in 4.2.2. This process is used by the Maneuver layer when the pedestrian has reached the end of their current local path (and has not reached their destination) and needs to plan a new path. It is crucial that this method emulates human decision-making as accurately as possible as it directly determines the pedestrian's long-range trajectory.

There are two options for the last lanelet of the selected sequence. A pedestrian's path either ends with a crosswalk leading them closer to their destination or the lanelet/area containing their final destination. In order for a scenario to be valid, the pedestrian's final destination must lie within a lanelet sub-type in which pedestrians are allowed (e.g. sidewalk, crosswalk) as defined by the map file. A pedestrian can either perform the local path planning method by taking into account the relevant crossing signal states or by disregarding their current states. Both cases are used in different scenarios. For example, if the pedestrian is initialized a far distance from any crosswalk entrances, then they must plan a path towards their destination; however, it does not make sense to consider the crossing signal states yet as they will be completely different by the time the pedestrian reaches their target crosswalk and is ready to enter. As the pedestrian approaches the accessible crosswalks at the intersection corner, they may run this path planning method again, this time taking into account the current state of the relevant crossing signals as the signals will now have an effect on their decision of which crosswalk to enter and at what time.

The path planning algorithm is simplified into the following steps describing its process.

1. Find accessible crosswalks
   - Accessible crosswalks are found using the lanelets defined in the underlying Lanelet2 map. If a sequence of connected lanelets can be found from the pedestrian's current position to a given crosswalk, then the crosswalk is considered accessible to the pedestrian. This step compiles a list of all crosswalks accessible to the pedestrian.
2. From all accessible crosswalks, determine candidate crosswalks
   - In general, a crosswalk is considered a candidate if it is advantageous for the pedestrian to take it strictly from a distance perspective. In other words, if the crosswalk's exit is closer to the pedestrian's destination point than its entrance (determined by the Euclidean distance), then it is considered a candidate.
3. From all candidates, determine target crosswalk. Recall the *target* crosswalk is the first (and only) crosswalk in the local path. If a local path contains a crosswalk, it will be the last in the connected sequence of lanelets that make up the path.
   - If the pedestrian is considering the current state of the relevant crossing signals in their decision, then their target crosswalk is determined by the rules in the next section. Note that the selection process is dependent on the pedestrian's level of aggressiveness.
   - If the pedestrian is not taking into account the crossing signals' states, then the target crosswalk is simply the candidate that takes the pedestrian closest to their destination if crossed.
4. If a target crosswalk is selected, find the shortest path to the target crosswalk
   - Lanelet2 provides built-in methods to determine the shortest path between two given lanelets. These methods are used to find a connected sequence of lanelets from the pedestrian's current position to their target crosswalk.
5. If there are no accessible or candidate crosswalks, construct a sequence of lanelets from current position to destination

- Using the built-in Lanelet2 methods, determine the order sequence of lanelets representing the shortest path from the pedestrian's current position to their destination point.
- We can assume that this sequence of connected lanelets exists because of the requirements on the map file. If no crosswalk is advantageous to take, then there must be a path along the sidewalk directly to the destination.

Rules for Selecting Target Crosswalk from Candidates

In this section, we discuss the separate set of rules for a pedestrian to select their target crosswalk given a list of candidates based on the pedestrian's level of aggressiveness. Recall that the level of aggressiveness is set when the pedestrian agent is created and the three levels are **Low**, **Medium**, and **High**. For simplicity, we use the term *green light* to refer to the pedestrian crossing signal "walking" state, *yellow light* to refer to the countdown to red state, and *red light* to refer to the "red hand" stop state.

In the case that a candidate crosswalk shows a yellow light, the pedestrian may need to determine if they can cross the crosswalk before the signal turns to the red state. Equation 4.1 is used as a quick calculation to judge whether or not this is possible.

Given a list of candidate crosswalks, the rules for selecting a target crosswalk by each level of aggressiveness are given below.

Low

- If there are any green lights among the candidates, select the best of the green candidates. The best candidate is the candidate whose exit point is closest to the pedestrian's destination point by Euclidean distance.
- Else if not all candidates have same the same signal state, select the best of the red candidates.
- Else, wait for a change in signal states and repeat process.

Medium

- If there are any candidates with yellow lights where it is determined that a crossing before red is possible or green lights, select the best candidate among them.

- Else if not all candidates have same the same signal state, select the best of the red candidates.
- Else, wait for a change in signal states and repeat process.

High

- Select the best candidate regardless of light state.

4.2.5 Motion Planner Layer

The Motion Planner layer is driven by an adaptation of the Social Force Model. It receives the three components passed by the Maneuver layer describing changes to the pedestrian's trajectory. These three components are a waypoint, a unit direction vector, and a desired walking speed. This layer also has access to the traffic state estimation and therefore all positions and velocities of the other agents in the scene.

Our model implements the classic SFM described in Section 3.1 extended by Anvari's method for handling pedestrian-vehicle interactions. The parameters of our SFM were manually calibrated through simulation testing. However, in a traffic environment, it was found there there are no relevant borders within the physical testing and simulation space that pedestrians need to account for (walking within a sidewalk or crosswalk's borders is handled through the **Keep in Lane** maneuver in the Maneuver Layer). Therefore, the wall forces are not included in our Social Force Model calculation as the list of walls is empty. This may be adapted for future environments containing physical walls the pedestrian must account for.

The Motion Planner layer is tasked with running an iteration of the Social Force Model's formula below to determine the model pedestrian's change in velocity at each simulation time step.

$$\begin{aligned} m_i \frac{dv_i}{dt} &= f_{adapt} + f_{otherPeds} + f_{vehicles} \\ &= m_i \frac{v_i^0(t) e_i^0(t) - v_i(t)}{\tau_i} + \sum_{j, j \neq i} f_{ij} + \sum_k f_{ik} \end{aligned}$$

Subsequently, this layer directly updates the pedestrian's state information before the next simulation cycle. It is important to note that, while the SFM is designed to describe interactions between human agents, we include relevant vehicles in the interaction forces. The physical interactions between pedestrians and vehicles are negligible most of the time as vehicles generally maintain a safe distance from pedestrians. However, we observed cases of pedestrians needing to walk around vehicles stopped in the middle of the crosswalk so it is necessary to include them as part of the sum of interaction forces.

4.3 Model Implementation

The presented model is written in Python and integrated into GeoScenario Server, an open-source full scenario simulation environment capable of running predefined traffic scenarios as a standalone application. GeoScenario Server provides the necessary infrastructure for full scenario simulation. It parses GeoScenario scenario files and initializes the necessary elements (agents, agent goal points, traffic lights, etc.) into the server, reads and loads the Lanelet2 map file to provide the underlying road structure for the scenario, maintains the static and dynamic environment and facilitates the information flow between dynamic agents and their surrounding environmental context, and finally runs the scenario by iterating through each of its agents to update their state in the environment at each simulation cycle. GeoScenario Server also implements Queiroz's SDV model for simulated vehicles to allow scenarios with dynamic interactions between pedestrian and vehicle simulated agents.

GeoScenario Server provides the optional integration with WiseSim, a simulator based on UnrealEngine [21] to run the Wise Automated Driving System. GeoScenario Server incorporates a shared-memory interface with WiseSim to integrate its dynamic agents and scenario environment into the simulator. This integration is an ongoing project and is provided as a sample integration intended to have future improvements.

Source code for GeoScenario Server which includes our model implementation can be found on GitHub at https://github.com/rodrigoqueiroz/geoscenarioserver.

Chapter 5

Evaluation

The following research questions guide our evaluation process as we evaluate our model in terms of how it can reproduce low-level trajectories and high-level decisions observed in a recorded data set and how well it extends to different road structures and geometries.

RQ1: Can our hierarchical model produce realistic low-level trajectory movements of a pedestrian's journey through a traffic environment?

RQ2: Can our model replicate high-level decision-making processes made by real-world pedestrians?

RQ3: Can the model be extended to different road structures without loss of realistic movements?

5.1 Configuration of Evaluation Scenarios with Pre-recorded Trajectories

5.1.1 Evaluation Scenario Configuration

To approach and answer our research questions, we need a standardized process for comparing a simulated pedestrian generated by our model against a real-world pedestrian from a naturalistic data set. We devise a process to create *evaluation scenarios*. These generated scenarios assist in validating our model against real-world data and provide a standard process that can be applied to any data set in a compatible format. A data set used to create evaluation scenarios must provide positional trajectory information for all vehicles and pedestrians as well as traffic light timing data.

The idea of an evaluation scenario is to replace a single pedestrian in a given recording with a model pedestrian and observe how the model pedestrian interacts with the other pedestrians and whether it follows the same trajectory as the empirical pedestrian it replaced. In terms of a GeoScenario scenario, one pedestrian is selected as the *evaluation pedestrian* and is created as an SP agent while all other pedestrians and vehicles in the recording file are created as TP and TV agents respectively, following their corresponding trajectories from the data set. We refer to the data-set pedestrian that the SP agent replaced as the *empirical pedestrian* or alternatively, the evaluation pedestrian's *empirical counterpart*. The scenario begins when the evaluation pedestrian enters the recording and ends when they exit.

The SP agent is created with three pieces of knowledge about its empirical counterpart: its starting position, its last position, and its average walking speed. In terms of the Social Force Model within the model, the last position is set as the SP's destination point and the average walking speed is set as the SP's desired speed. By default, evaluation pedestrians run the behaviour tree shown in Figure 4.4 and are assigned a Medium level of aggressiveness and therefore substitute the sub-tree node in their behaviour tree with sub-tree 4.6.

The evaluation scenario creation process is repeated for each individual pedestrian in the data set resulting in a unique evaluation scenario for each real-world pedestrian. This means that each data set pedestrian produces their own evaluation scenario in which they are replaced by a model pedestrian that dynamically interacts with the other agents. This process compiles a suite of evaluation scenarios on which we can perform analysis and draw conclusions in terms of our research questions.

5.1.2 Intersection Data Set

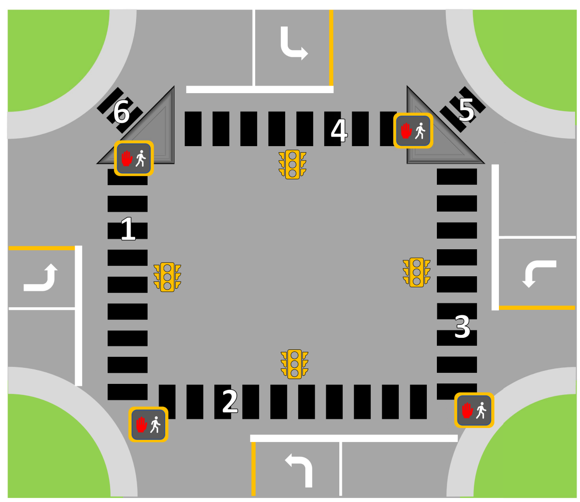
The evaluation scenarios described in Section 5.1.1 require a naturalistic data set to provide the prerecorded trajectories. For this purpose, we use a data set recorded by an overhead drone at a high-traffic four-way intersection in Waterloo, Canada. The drone collected a total of 63.85 minutes (approx. 1h:03m:51s) of video recordings across 14 separate video files. The video files were analyzed and annotated by DataFromSky [1] in which each vehicle and pedestrian agent was precisely tracked and tagged. From the annotated video files, each agent's trajectory was extracted into database files along with supplemental information about the recordings such as traffic light timings. Refer to Table 5.1 for the total number of road users by their types across all data set files.

Type	Count
Pedestrian	264
Bicycle	15
Car	3352
Medium Vehicle	182
Heavy Vehicle	61
Bus	22
Motorcycle	17

Table 5.1: Counts of road users by type in the intersection data set.

Figure 5.1 displays an overhead shot of the intersection taken from Google Earth alongside a simplified representation of the physical structure of the intersection location annotated with traffic lights and numbered crosswalks.

(a)



(b)

Figure 5.1: Side-by-side comparison of the intersection data set location with a shot taken from Google Earth (a) and its simplified representation (b).

Pedestrian Crossing Signal Timings

While the vehicle traffic light timings were provided with the intersection data set, the light timings for the four pedestrian crossing signals were not provided. These timings therefore needed to be inferred from the vehicle traffic light timings augmented with general knowledge about this particular intersection location. This section makes reference to the crosswalks by their numbers in Figure 5.1b.

In order to extract the pedestrian crossing signal timings, we require a set of rules known to be true for the recorded intersection.

1. There are only crossing signals for crosswalks numbered 1-4 in Figure 5.1b. Crosswalks 5 and 6 do not have crossing signals so pedestrians take the right-of-way.

2. The crossing signal turns green at the same time the corresponding traffic light turns green. Note that the corresponding traffic light is the light signalling the driving lanes parallel to the crosswalk. Additionally, note that when we say the traffic light is green, we are referring to the main light and not the left turn light (advanced green light) that may have a different state.

3. The crossing signal turns red at the same time the corresponding traffic light turns yellow.

4. The countdown to red (yellow state) is 16 seconds in length.

5. When one driving direction has both a green light and an advanced green light and all other traffic lights are red, the crosswalk to the right of the vehicles facing the green light will also have a green signal. All other crosswalks in this situation are red. This property is dependent on the specific intersection and was verified for this data set location.

Given the physical configuration of the intersection shown in Figure 5.1 and the above list of rules, we can extract the pedestrian crossing light timings and durations from the traffic light timings provided in the data set.

5.1.3 Full and Segmented Evaluation Scenarios

All evaluation scenarios are categorized into one of two groups: full or segmented. Each pedestrian in the data set produces exactly one full scenario that begins when they appear in the recorded scene and finishes when they are no longer in the scene. Each full scenario can be subdivided into two or more segmented scenarios. Each segmented scenario shows the pedestrian traversing a single crosswalk or section of sidewalk. In general, a full scenario is segmented each time the pedestrian passes from one walking area into another walking area with a different type (e.g. from a sidewalk to a crosswalk or vice versa). This approach splits a pedestrian's full journey into multiple sections of interest and creates a new evaluation scenario for each segment. We refer to whether an evaluation scenario is full or segmented as the scenario's *length* (as opposed to its *duration* in seconds).

Since full scenarios can have relatively long running times (average duration of 66.82 seconds), small deviations in trajectories can amplify and compound into large deviations further into the scenario. The purpose of segmenting full scenarios is to help mitigate this increase in error that is due to only a small difference in an early decision and does not necessarily reflect the model's performance throughout the pedestrian's entire journey. Segmented scenarios ensure that the evaluation pedestrian's position and velocity is reset to its empirical counterpart pedestrian at certain times to effectively test the model's performance over shorter intervals.

5.2 Producing Realistic Low-Level Trajectory Movements (RQ1)

It is crucial that our behaviour model produces pedestrian movements that are as natural and human-like as possible in order to be relied upon as a realistic representation of pedestrians in test scenarios. To evaluate our model's effectiveness at generating realistic motion, we run the evaluation scenarios (5.1.1) with the intersection data set (5.1.2) and record the evaluation pedestrian's generated trajectory. We compare this simulated trajectory generated by our model with the trajectory of the corresponding empirical pedestrian by three different trajectory-matching metrics: Euclidean distance, discrete Fréchet distance, and undirected Hausdorff distance. Given the dynamic nature of humans, it is unreasonable to expect a single configuration of our model to completely cover the varying behaviours of all the pedestrians in the data set. To accommodate this, we identify a custom configuration of parameter values for each evaluation scenario that best describes the corresponding real-world pedestrian's actions and behaviours. In the rare instances that our model behaves completely differently from its target empirical pedestrian, we propose solutions and extensions of the model to reduce the error.

5.2.1 Trajectory-Matching Metrics

In this section, we introduce the three trajectory matching metrics used to evaluate the simulated trajectory generated by the model against the ground-truth trajectory of the empirical pedestrian. We compute the Euclidean distance (ED), Fréchet distance, and Hausdorff distance. With the Euclidean distance, we aim to compare the trajectories when they are sampled by discrete time steps. We provide the mean ED and the worst case maximum ED averaged over the set of evaluation scenarios. Both the Fréchet and Hausdorff distances are a measure of geometric similarity which we use to evaluate the similarity between generated paths. We include two such metrics, despite their closeness, to emphasize the importance of producing the same path (and therefore the same decisions) as the real-world pedestrian.

Baseline Trajectory Method

To give context to our model's results, we introduce a baseline method to generate a trajectory-approximation for each empirical pedestrian. We produce a baseline trajectory for each evaluation scenario based on the empirical pedestrian's path. We first define a set of points on the intersection map that can be connected in various orders to approximate the pedestrian's path from the data set. These points, along with an example baseline trajectory, are shown in Figure 5.2. The selected map points closely resemble the waypoints a model pedestrian may select to navigate through the intersection.

For each empirical trajectory, the real-world pedestrian's start and end points are connected through a sequence of 2D points, placed on the map, in such a way that best approximates the actual path. A trajectory is then formed by filling in equi-distanced (and equi-timed) points along the approximated path such that the total number of points is equal to the total number of trajectory points in the empirical path. This produces a constant-speed baseline trajectory that has the same duration as the corresponding data set trajectory. Note that we only produce a baseline trajectory for the full evaluation scenarios due to the simplified nature of the segmented scenarios. This also avoids redundancy since the segmented scenarios are derived directly from the full scenarios.

Euclidean Distance

The Euclidean distance (ED) measures the spatial similarity between two trajectories in the 2D plane, in our case, between the simulated trajectory and the true points of the

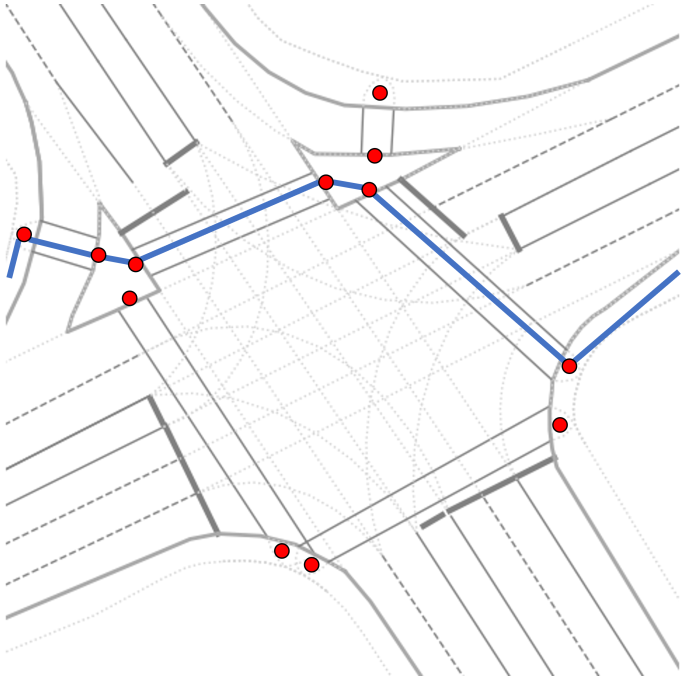


Figure 5.2: A visual representation of the set of defined points (red) with which a baseline trajectory is generated to approximate each empirical trajectory. An example baseline trajectory is shown by the blue path.

empirical trajectory. It is one of the most common and popular distance measures for its simplicity and is also known as the spatio-temporal Euclidean distance (STED) [54]. Given trajectories A and B, each with n 2-dimensional points, the Euclidean distance measure, d_{ED}, is calculated by

$$d_{\mathrm{ED}} = \frac{1}{n}\sum_{i=1}^{n} d(a_i, b_i)$$

where a_i and b_i are the i^{th} points of A and B respectively and $d(a_i, b_i)$ is the Euclidean distance between them.

It is possible for the compared trajectories to be of different lengths if the empirical and evaluation pedestrians reach their destinations at different times. The points of each trajectory are equi-timed, so if the trajectories are recorded over different durations, then they will each have a different number of points. Since it is required for calculation of the ED that both A and B contain the same number of points, we simply truncate the longer trajectory to match the length of the shorter one. This will show an increase in error in the situation where the destination point is reached at different times. This is valuable to record since a goal of the model is to replicate the empirical trajectory as closely as possible at every point in time. Classically, the Euclidean distance is not a function of time as it simply requires ordered sequences of points for comparison. However, since our compared trajectories represent moving objects in space over a period of time (but possibly two different durations of time) and are recorded at the same frequency, temporal information is inherently relevant to our Euclidean distance measure.

Fréchet Distance

A common analogy used to explain the Fréchet distance is the scenario where a person is walking a dog on a leash. Both the person and the dog may walk at different speeds but they cannot move backwards. The person traces one trajectory while the dog traces another separate trajectory. The Fréchet distance is the minimum length of the leash required such that both the dog and person can walk their respective trajectories. Mathematically, suppose both trajectories, A and B, are functions on the interval $[a, b]$. We define two arbitrary continuous non-decreasing functions $\alpha, \beta : [0, 1] \mapsto [a, b]$, then the Fréchet distance, d_F, is given by

$$d_F = \inf_{\alpha,\beta} \max_{t\in[0,1]} d(A(\alpha(t)), B(\beta(t)))$$ [20]

where $d(p, q)$ is the Euclidean distance between points p and q.

The Fréchet distance is measures the similarity between two trajectories without taking into account the time or speeds the trajectories were traversed, unlike the ED. This is advantageous to measure because even though a model pedestrian may choose a different desired speed than their empirical counterpart at a given moment, they can still make the same decisions and produce the same path through the traffic environment which should be rewarded.

Hausdorff Distance

As with the Fréchet distance, the Hausdorff distance is a measure of geometric similarity between the two trajectories. It does not consider the speeds of the pedestrians or the times at which the points of each trajectory are recorded in its calculation. Informally, given two sets of points (in this case, trajectories), the Hausdorff distance is the greatest distance from one set to the closest point in the other set.

The general undirected Hausdorff distance is given by

$$d_H = \max \left\{ \sup_{a \in A} d(a, B), \sup_{b \in B} d(b, A) \right\}$$

where $d(a, B) = \inf_{b \in B} d(a, b)$ and $d(a, b)$ is the Euclidean distance between points a and b. The emphasis on geometric-similarity measures between the trajectories is derived from the intention of our model as a behaviour model aimed at making the same high-level decisions as real-world pedestrians, rather than as a predictive model which attempts to perfectly replicate the low-level empirical trajectory points.

5.2.2 Empirical Results

For each evaluation scenario (full and segmented) generated by the data set, we compare the simulated trajectory from the evaluation pedestrian against the ground-truth trajectory from its empirical counterpart using the three trajectory-matching metrics. Table 5.2 displays the results of each metric averaged across all scenarios and grouped by scenario length. In the table, we also include the maximum Euclidean distance averaged over the evaluation scenarios.

	# of Scenarios	ED (m)	Max. ED (m)	FD (m)	HD (m)
Baseline	198	5.45	12.48	2.90	2.91
Full	198	2.23	5.21	3.04	2.99
Segmented	527	1.55	2.65	1.81	1.79

Table 5.2: Results of the Euclidean distance (ED), Fréchet distance (FD), and Hausdorff distance (HD) measures averaged and grouped by full and segmented evaluation scenarios at the intersection data set location.

While these results confirm our model produces synthetic trajectories with reasonable similarity to their target ground-truth trajectories, we must explore the individual evaluation scenarios to identify points of high error and investigate possible explanations. To assist with this search, we plot a trace of the simulated trajectory overlaying a trace of the empirical trajectory to easily identify points of separation visually. Figure 5.3 displays an example of such a plot with areas of high error. Through this process, we discover that the points of highest error occur when the pedestrian is entering or exiting the crosswalk. The layout of the plots make it easy to identify the high error at crosswalk entrances and exits. Visually, we can see that the data-set pedestrians often cut the crosswalk corner when entering instead of travelling all the way to the entrance of the crosswalk before entering. Similarly for exiting the crosswalk. This observance demands an alternative method for SP pedestrians to enter and exit crosswalks in order to more closely match the trajectories in our naturalistic data set.

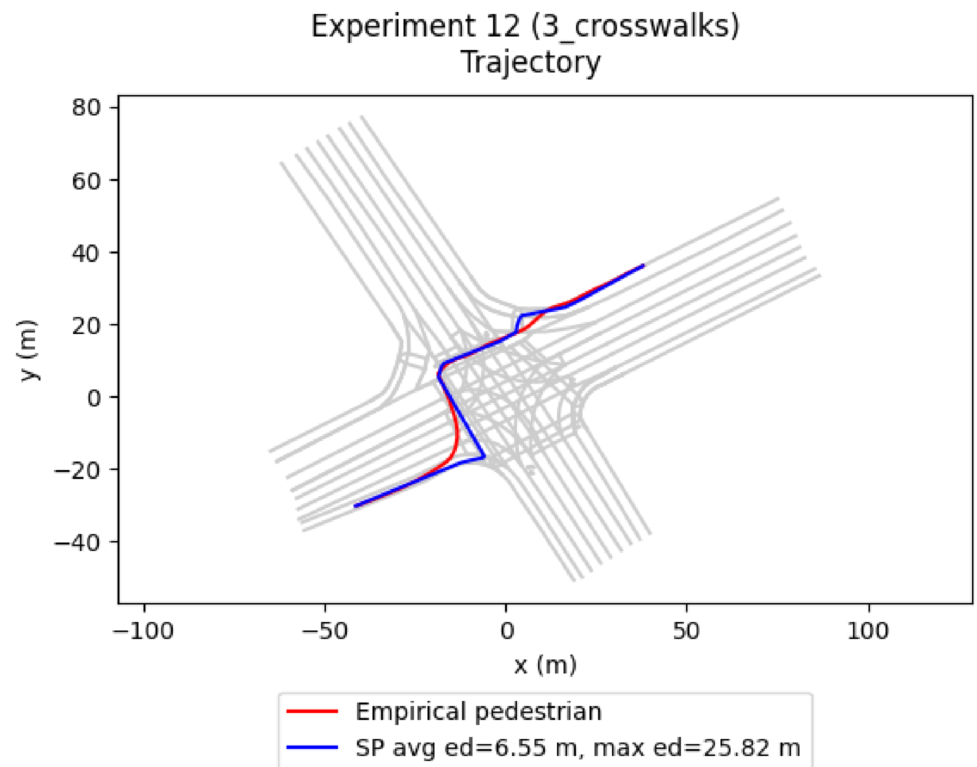


Figure 5.3: A trace of the trajectories from the empirical (red) and simulated (blue) pedestrians overlaid on the intersection map from one evaluation scenario.

Corner Cutting Behaviour

Corner cutting behaviour by real-world pedestrians was identified as a cause of major deviation between synthetically generated and ground-truth trajectories. Figure 5.4 clearly visualizes the corner cutting behaviour when entering a crosswalk and compares the resulting trajectory to one from a non-corner cutting entry.

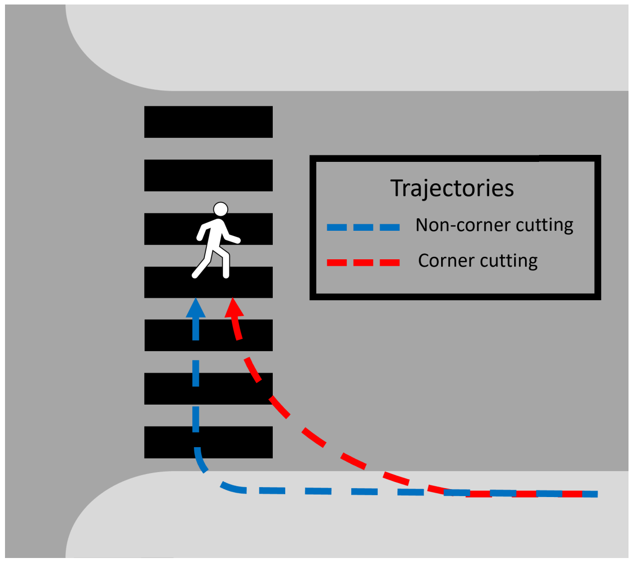


Figure 5.4: Example trajectories resulting from both corner cutting and non-corner cutting behaviours when entering a crosswalk.

It was determined that the **Enter/Exit Crosswalk** maneuvers must be adapted to apply a corner cutting decision. For this example, we will focus on corner cutting when entering a crosswalk, but the same process is applied to exiting as well. In a typical case where corner cutting is **not** applied, the model pedestrian follows the steps outlined below.

1. The pedestrian executes **Keep in Lane** maneuver as they approach their target crosswalk from the sidewalk.

2. The pedestrian is within a threshold distance from their target crosswalk's entrance point which is their current waypoint.

3. If their behaviour tree decides they can enter the crosswalk, then **Enter Crosswalk** maneuver is executed.

4. The current waypoint is updated to be the crosswalk's exit point so the pedestrian can enter the crosswalk freely.

Since this process often involves approaching the entrance point from an heading close to perpendicular to the crosswalk, reaching that point, then immediately updating the waypoint to the exit point that requires a heading parallel to the crosswalk, the resulting trajectory contains a sharp, almost 90°, turn which does not fit the natural curve of the real-world entry trajectory. It is clear that simply updating a waypoint once the entrance point has been reached is not sufficient in replicating realistic crosswalk-entering behaviour.

The solution for this problem was found by manipulating the pedestrian's desired direction vector over a given duration of time as they enter the crosswalk. The steps below are followed to achieve this.

1. The pedestrian executes **Keep in Lane** maneuver as they approach their target crosswalk from the sidewalk.

2. The pedestrian is within a threshold distance, $dist_{\text{entry}}$, from their target crosswalk's entrance point. Note that this threshold distance is greater than the threshold in Step 2 of the previous list of steps (approx. 5 m, but it is configurable).

3. Define a duration of time, d, and adjust the pedestrian's desired direction vector by following the steps below such that at the beginning of d, the direction vector points directly at the crosswalk's entrance point at by the end of the duration, it points to the exit point.

   (a) Define duration d in seconds, unit vectors v_{entry} and v_{exit} from the pedestrian's position to the crosswalk's entrance and exit points respectively, and a function of time $f(t) = t^n$ with degree $n > 0$.

   (b) At time $t_i \in [0, d]$, the desired direction vector, v, is given by

   $$v = f\left(\frac{t_i}{d}\right) * v_{exit} + \left(1 - f\left(\frac{t_i}{d}\right)\right) * v_{entry} \tag{5.1}$$

   (c) Since $t_i \in [0, d]$ and $f(t) = t^n$, then $f\left(\frac{t_i}{d}\right)$ and $\left(1 - f\left(\frac{t_i}{d}\right)\right)$ act as weights for the direction vectors such that $v = v_{entry}$ when $t_i = 0$ and $v = v_{exit}$ when $t_i = d$.

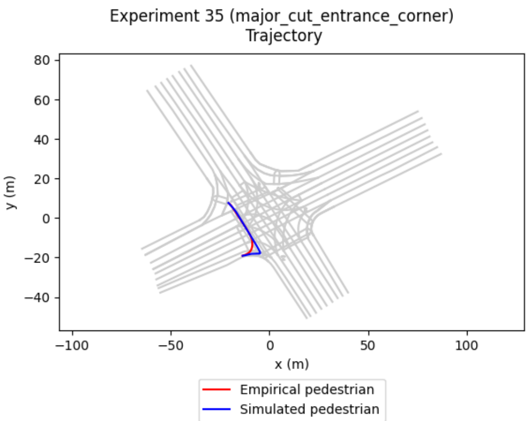
This approach for entering a target crosswalk produces a smoother, more natural trajectory curve from the sidewalk to the crosswalk. The configurable elements of the duration, the function degree n, and the threshold distance from the crosswalk entrance point at which this maneuver starts, provide more control over the corner cutting action for the scenario designer. For example, varying values of n controls the "turn rate". Lower values will cause the pedestrian to turn its direction vector towards the exit point at a slower rate than higher values of n.

Note that the same process can be applied when exiting the crosswalk by substituting v_{exit} for v_{entry} and the pedestrian's next waypoint as determined by the local path planning process for v_{exit} in Equation 5.1. Additionally, this corner cutting action can be applied any time the pedestrian approaches the entrance of their target crosswalk, not only when their approach is from a close to perpendicular angle to the crosswalk. The corner cutting effects may be less pronounced with a more "head-on" approaching direction, but the principles of gradually turning towards the exit point (or the next waypoint in general) are still relevant and applicable.

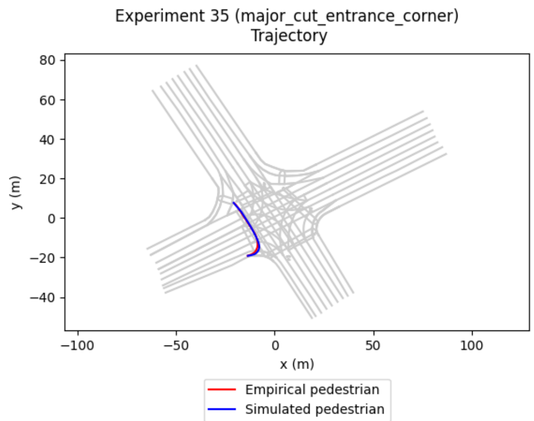
We found that this approach of adjusting the desired direction vector over a duration of time reduced the difference in trajectories at entering and exiting crosswalk events. To demonstrate how this method can be applied to reduce error in evaluation, we present an evaluation scenario in which the real-world pedestrian cuts the entrance corner of their target crosswalk. We will show two instances of this scenario. In the first, the model pedestrian does not apply any corner cutting process while entering the crosswalk and in the second, the steps above were implemented into the model and corner cutting parameters were derived to allow the model pedestrian to naturally cut the corner of the crosswalk entrance. Figure 5.5 shows these instances side-by-side to fully visualize the improvement in replicating the empirical trajectory.

The evaluation of the added corner cutting behaviour into the model was tested on a case-by-case basis by searching for optimal parameter values for the duration d and function degree n in Equation 5.1 for individual evaluation scenarios as necessary. Not every pedestrian applies the same corner cutting action in a given context. It is left for future work to determine the influencing factors that lead to a corner cutting event, automate the search for these optimal parameter values for all evaluation scenarios, and dynamically apply them during evaluation.

Another frequent point of high error observed in the evaluation scenarios occurs at the short unsignalized crosswalks that cross the designated "right-turn merge lanes". In Figure 5.1b, these are crosswalks 5 and 6. We found that pedestrians often avoid walking within these crosswalks altogether and simply walk from the traffic island directly to a point on the crosswalk that is closer to their destination than the crosswalk exit is (or vice versa if they are entering the intersection instead of leaving). We designate this action as *jaywalking beside the crosswalk*. While this falls under the umbrella of corner cutting behaviour, we can apply a different simpler approach to reduce differences in trajectories in these situations. Our model defines threshold distances, in meters, to a crosswalk's entrance



(a) No corner cutting



(b) Applying corner cutting with $d = 7$s, $n = 0.35$, $dist_{\text{entry}} = 7$m

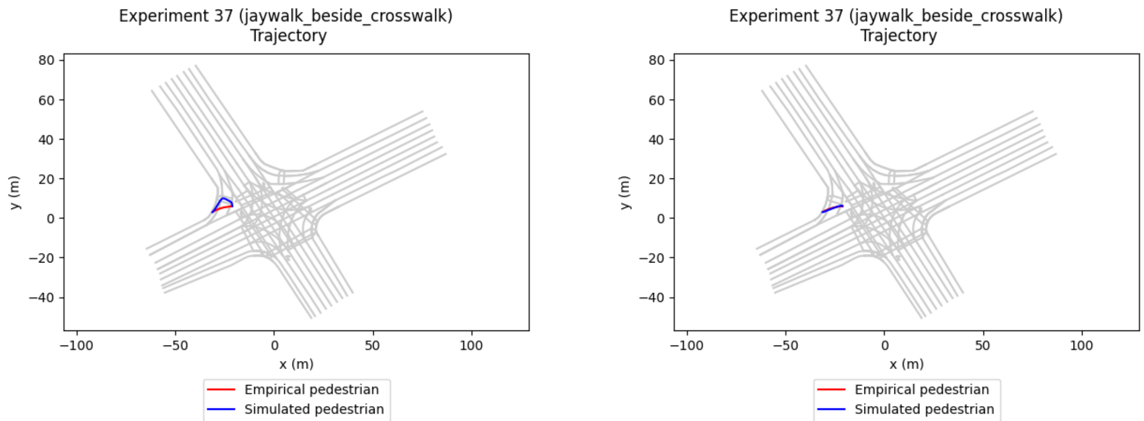
Figure 5.5: Sub-figure (a) shows evaluation scenario where SP pedestrian (blue) does not apply the corner cutting procedure. Sub-figure (b) shows the same evaluation scenario except where SP pedestrian applies the corner cutting procedure to match the empirical trajectory (red) more closely.

and exit points that determine when the model pedestrian has sufficiently "reached" the respective entrance or exit. By simply adjusting the threshold values of these parameters, we can accurately replicate the behaviour of jaywalking beside the unsignalized crosswalks. Figure 5.6 displays a side-by-side comparison of the resulting trajectories with and without this described maneuver.

As with the corner cutting process, the parameters within the evaluation scenarios containing this situation were determined manually on a case-by-case basis and it is left to future work to automate the search for optimal parameter values for a given evaluation scenario.

Reactions to Yellow Crossing Signal

Another key situation where we observed a major difference in behaviour and decision-making among pedestrians is when a pedestrian intends to cross a crosswalk but is faced with a yellow (countdown) crossing signal state. Since different pedestrians have different walking speeds, varying levels of aggressiveness, and simply different senses of urgency, some pedestrians may choose to enter the crosswalk while others may not. With the



(a) Without jaywalking maneuver

(b) Applying jaywalking beside crosswalk maneuver with $dist_{\text{entry}} = 3\text{m}$, $dist_{\text{exit}} = 6.5\text{m}$

Figure 5.6: Sub-figure (a) shows evaluation scenario where SP pedestrian (blue) does not apply the jaywalking maneuver. Sub-figure (b) shows the same evaluation scenario except where SP pedestrian applies the jaywalking beside the crosswalk maneuver to match the behaviour of the empirical pedestrian (red).

"birds-eye" view of the data set, we simply do not have the factors for determining a pedestrian's likelihood to enter on a yellow signal available to us. Instead, we leverage another level of customization in our model which allows testers to control a pedestrian's reaction to the countdown signal state.

The crossing decision in this situation is driven by whether or not the pedestrian determines they can sufficiently cross the crosswalk before the crossing signal turns to red. The customization parameters to control this decision are described in the **Can cross before crossing light turns red** condition node in Section 4.2.4. It is important to note that this decision is only relevant for pedestrians with a Medium level of aggressiveness. The tester is provided with 2 tunable parameters listed below.

1. Decimal factor by which the pedestrian can reasonably increase their walking speed
2. Distance in meters from crosswalk exit the pedestrian must reach to have sufficiently crossed the crosswalk

These parameters provide a grid of varying levels of acceptance for a crossing decision in this situation. During the running of our evaluation scenarios, these parameters were calibrated on a case-by-case basis to match the behaviours exhibited by the observed real-world pedestrians. In future test scenarios, designers and engineers can tune these parameters to manually force desired behaviours, potentially generating valuable high-risk and dangerous situations safely in a simulation environment.

5.3 Replicating High-Level Decisions (RQ2)

For a pedestrian behaviour model to properly represent the behaviours of real-world pedestrians, not only must it model the low-level trajectory movements, but it must also be able to replicate high-level decision-making processes. In order to measure this criterion, we need to define a metric by which we can conclusively declare that a decision made by a model pedestrian is the same decision made by its empirical counterpart. The hierarchical approach is well suited to model and control an agent's decisions through the design and calibration of the behaviour trees in the Behaviour layer.

5.3.1 Decision-Based Metric

First, we must define a list of decisions to be observed from the naturalistic data. It was determined that there are two notable decisions pedestrians make while navigating an intersection.

1. Given a set of accessible crosswalks and the requirement that at least one crosswalk must be crossed to reach the destination, which crosswalk is selected to cross
2. Given a target crosswalk that the pedestrian has already decided to take which displays a non-green crossing light state, will the pedestrian begin to cross or wait until the next green state

The above decisions were designed to ensure a binary response can be recorded for each instance of the decision across evaluation scenarios. The model pedestrian either makes the same decision as the data set pedestrian or they do not. In the rest of this section, we refer to these decisions as Decision 1 and 2 respectively. The process followed to measure this metric is as follows:

1. Run all full length evaluation scenarios within the intersection data set

2. For each evaluation scenario, identify *decision points* along the empirical pedestrian's journey where they make one of the two enumerated decisions above
3. For each decision point, determine if the evaluation pedestrian makes same decision as the empirical pedestrian
4. If the same decision is made, record a *Same* data point
5. If a different decision is made, search for parameter values that result in the same decision when the evaluation scenario is re-run
6. If no parameter combinations are found to produce the same decision, record a *Not Same* data point
7. Calculate percentage of *Same* data points from all recorded data points to measure the model's performance of this metric

The results of following the above collection process are summarized in Table 5.3. We record the number and percentage of *Same* decisions after individually calibrating the parameters of the scenarios that initially gave *Not Same* decisions. This allows us to view how many decision points in the data can be fully covered by some customized parameter value configuration of our model.

	# Decision Points	**# Same Decisions**	**% Same Decisions**
Decision 1	253	253	100%
Decision 2	148	145	97.97%

Table 5.3: Summary of decision-based metric results on full length evaluation scenarios using the intersection data set.

From Table 5.3, we can be reassured of our model's ability to select the same crosswalk a real-world pedestrian would pick given multiple candidates and the current environmental state. However, there is a slight difference in Decision 2, where the model pedestrian encounters a non-green crossing signal at their target crosswalk and must decide whether to enter it or wait until the next green signal to cross. These instances where the model pedestrian made the wrong decision can be explained by the real-world pedestrian's unpredictable behaviour in these particular scenarios and can be covered by our proposed extensions of the current model implementation.

There are three instances where the model pedestrian made the wrong choice for Decision 2. All of these instances occurred when the empirical pedestrian decided to enter the crosswalk on a red stop signal. Each instance shares a common situation: there is an advanced green light for the vehicles that will intersect crosswalk the pedestrian is waiting to cross; however, there are currently no left-turning vehicles that would make it dangerous for the pedestrian to cross, so the pedestrian decides to enter the crosswalk even though it is showing a red stop signal. Since our model pedestrians do not consider the light state of the vehicle traffic lights in their decisions, we do not cover this specific decision making process resulting in a difference of decisions in the three evaluation scenarios.

To cover these cases, we propose an extension of our model that augments the existing behaviour trees by adding condition nodes that check the light states of certain vehicle traffic lights in the intersection. There are multiple approaches and behaviour tree configurations that could account for this situation. One of which may involve the model pedestrians keeping track of the green traffic lights that correspond to lanes that intersect their target crosswalk. If there are no approaching vehicles in any of these lanes, then they can enter the crosswalk regardless of the crossing signal state. This particular behaviour would be categorized between the Medium and High levels of aggressiveness. Note that this extension does not require changes to the code of the model and rather the desired behaviour is achieved through extending behaviour trees. This process does not require expert knowledge on the internal technical details of the model and can be done by any user, further highlighting the model's extendibility and practicality for scenario based testing.

5.4 Extending Model to Different Road Structures (RQ3)

The majority of test cases during development of our model were derived from the previously discussed intersection data set. While this data set provides a substantial number of pedestrians exhibiting a wide range of behaviours, a goal of any useful pedestrian model should be extendibility to differing road structures and geometries. We test the generalization and extendibility of our model by extracting evaluation scenarios from a second data set. Specifically, we want to test whether our model pedestrians can navigate through a different road structure without making changes to our model. We evaluate the model's ability to perform this with the trajectory-matching metrics described in Section 5.2.1.

5.4.1 Single Crosswalk Data Set

The second data set, referred to as the single crosswalk data set, is located at an unsignalized crosswalk across a two-lane two-way road at a Canadian university. The point of view and collection process are similar to the intersection data set and the trajectories files were extracted by the same process. Figure 5.7 shows the road structure of this location.

(a) (b)

Figure 5.7: Side-by-side comparison of the single crosswalk data set location with a shot taken from Google Earth (a) and its simplified representation (b).

This location does not contain traffic lights or pedestrian crossing lights, so no preprocessing regarding light timings was necessary. Across the 17 recorded video files, there are a total of 1153 pedestrians, 242 motorized vehicles, and 38 bicycles (refer to Table 5.4 for a breakdown by agent type). Recall from Section 5.1.1, each pedestrian with a valid trajectory creates a unique evaluation scenario. An example of a pedestrian appearing in the data set that does not have a valid trajectory is one that only appears in the corner of the scene for a very short amount of time and does not cross the road. Valid trajectories are judged manually on a case-by-case basis.

Due to the simplified road structure of a single crosswalk at this location, we do not segment any evaluation scenario to create additional scenarios. All evaluation scenarios created with this data set have unaltered lengths and are considered full.

Type	Count
Pedestrian	1153
Bicycle	38
Car	193
Medium Vehicle	27
Heavy Vehicle	4
Bus	12
Motorcycle	6

Table 5.4: Counts of road users by type in the single crosswalk data set.

5.4.2 Evaluation Metrics and Empirical Results

Given the simplified layout of this data set location, we do not measure high-level decision-making as we did with the intersection data set. Instead, we evaluate the model's performance on this data set using the three low-level trajectory-matching metrics from Section 5.2.1: Euclidean distance, discrete Fréchet distance, and undirected Hausdorff distance. There were no changes to the structure of the behaviour trees used by the evaluation pedestrians in the previous intersection data set. However, different parameter values may be applied to the behaviour trees' maneuver and condition nodes as necessary.

After generating all the valid evaluation scenarios from the single crosswalk data set, each scenario is executed and the generated trajectory of the model pedestrian is recorded. Each generated trajectory is compared to its corresponding empirical trajectory from the data set using our three trajectory-matching metrics. Table 5.5 displays the results of this process.

	# of Scenarios	ED (m)	Max. ED (m)	FD (m)	HD (m)
Full	1017	1.36	2.01	1.71	1.71

Table 5.5: Trajectory-matching metrics results for all evaluation scenarios at the single crosswalk data set location.

We see an improvement across all three metrics from the intersection data set location. While this is expected with generally shorter pedestrian journeys (average scenario duration of 12.32s), given the same evaluation criteria as our original data set, we can reasonably conclude that our model extends well to this secondary road structure in terms of replicating pedestrian trajectories. We further justify this claim by investigating individual evaluation scenarios by visually comparing the generated and prerecorded trajectories.

Figure 5.8 shows the trajectory traces of three example evaluation scenarios with different paths across the environment. Due to the physical structure of this location, there are limited unique paths through the scene to present.

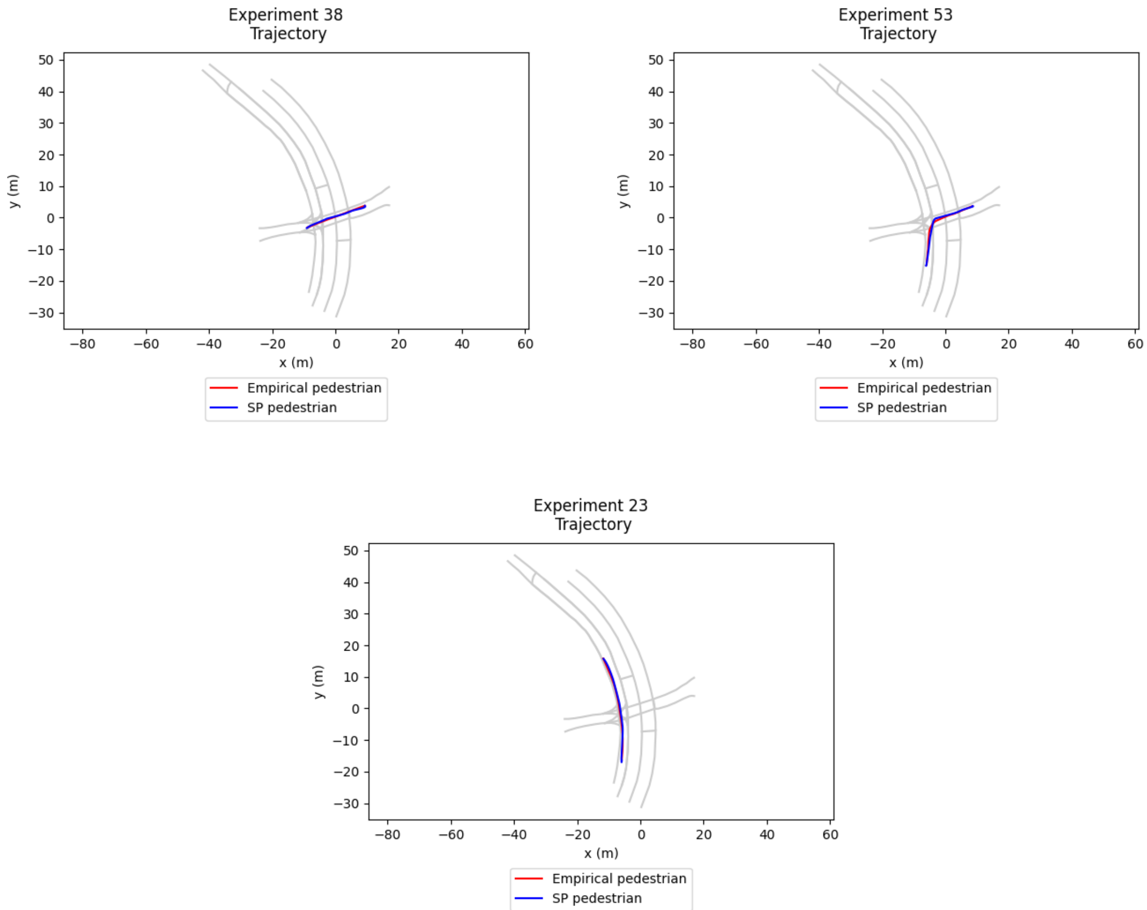


Figure 5.8: Trajectory traces of three example evaluation scenarios with varying paths from the single crosswalk data set.

Chapter 6

Limitations

In this chapter, we identify limitations of our work in terms of the research scope, model, and evaluation. With the limitations identified here, we aim to motivate future work and interesting research avenues.

6.1 Research Scope Limitations

In our work, we primarily focus on producing pedestrian behaviours and movements based on the two traffic data sets. During this process, we do not use our model to generate new scenarios. The next step of this process is to apply and refine our model in the context of scenario-based testing of autonomous vehicles. Our model caters to this step by facilitating the injection of behaviours into a traffic scenario with the intention of creating rare and critical scenarios.

6.2 Model Limitations

We also identify the notable technical limitations of our model. Since the catalog of maneuvers and conditions provided with our model was derived directly from the two data sets, we limit the coverage of pedestrian behaviours to those present in the available data. Extending the maneuvers and decision conditions would increase the possible behaviour tree configurations and therefore benefit the model's behaviour coverage.

In our model, we do not implement group dynamics that emerge when multiple pedestrians travel together. There are extensions of the Social Force model [52] [30] that have been developed to address group and subgroup behaviours. Alternatively, group dynamics can be handled by the higher-level behaviour trees with group-specific maneuvers and conditions. Additionally, our point-mass model of pedestrians limits the representation of pedestrians. With the current model, we abstract static and dynamic pedestrian body poses to a single point with mass. The addition of such poses has the advantage of being able to represent different pedestrian states that cannot be differentiated with a point-mass model, such as crouching versus standing up. New maneuvers may also be created based on body pose.

Furthermore, our model does not differentiate between different types of pedestrians, such as adults, elderly people, children, or people with disabilities. In doing so, we limit our pedestrian representation further as different types of pedestrians exhibit different behaviours. As an example, a group of children waiting for a school bus may pace and fidget in a small area with no intention of leaving that space. Exposure to these specific, yet common, situations is beneficial to the testing and development of AVs and can reduce misinterpretation of pedestrian actions.

6.3 Evaluation Limitations

Lastly, we discuss the limitations of our evaluation process. As mentioned in the model limitations, the amount of available data bounds the maneuvers and behaviours developed during evaluation. Between the two traffic data sets, there are a total of 1417 pedestrians and 3853 motorized road users. Moreover, the two data set locations are both within Ontario, Canada. Applying our model to data sets located in different countries would introduce new behaviours and would promote a more diverse set of maneuvers and conditions.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis, we presented a novel hierarchical pedestrian behaviour model that is capable of producing realistic trajectories through different traffic environments while following the rules of the road and making rational real-time decisions. A multi-layer approach was applied to this problem that incorporated a high-level Behaviour layer that determined an appropriate maneuver to be processed by the Maneuver layer and finally translated into low-level trajectory movement by the Motion Planner layer. The conjunctive use of behaviour trees with an adapted Social Force Model ensures the model's agents are an accurate representation of real-world pedestrians. They were shown to make the same decisions when faced with multiple options and also display natural movements and interactions with other pedestrians and vehicles in the scene.

We evaluated our model in terms of its ability to produce realistic low-level movements in two environments with different road structures, replicate high-level decisions made by real-world pedestrians observed in a naturalistic data set, and be extended to different road structures and geometries with varying rules for right-of-way without significant changes to the model code.

We revisit our design objectives and discuss how the model satisfies each one. G1 is concerned with the facilitation of test scenario creation. The GeoScenario language provides a clear representation of the full scenario from the underlying Lanelet2 map to the dynamic pedestrian and vehicle agents. Additionally, the JOSM scenario editing tool allows testers to easily create and adapt scenarios represented in GeoScenario. G2 requires our model to provide customizable pedestrian behaviours. The model allows designers to set a wide range of tunable parameters to adjust pedestrian behaviours, including but not limited to, level of aggressiveness, desired speed, and parameters to control a pedestrian's response to a yellow crossing signal. The use of behaviour trees to explicitly represent decision processes lends itself well to creating scenarios with highly intentional actions, possible injecting unusual behaviours to force rare events. The full scope of the requirement to have dynamic interactions between agents (G3) was partially covered by our evaluation process. Although our model effectively handled the interactions within our evaluation scenarios, since we evaluated on existing data, we did not show the full range of possible interactions between road users. This is left to future work for the test scenario design process. Finally, G4 required the model to display realistic human movements and decisions. This requirement is evaluated and satisfied and is empirically proven through the results of the high level decision metric and low level trajectory matching metrics.

7.2 Future Work

7.2.1 Improvements to Model

Throughout this thesis, we make note of possible improvements to our presented model to increase the level of intelligence of its pedestrian agents. In its current development iteration, the model pedestrians do not take into account the vehicle's traffic lights when selecting the optimal crosswalk to take. In the real-world, this information is available to human pedestrians and it could allow pedestrian agents to make a more informed decision in this situation. Additionally, we employ a fairly simplified version of the Social Force Model since we often allow higher-lever behaviour trees to handle interactions. However, there are on-going improvements made to the SFM, such as incorporating group forces and subgroup forces [30], any of which could be integrated into our model to elevate the Motion Planner layer. In terms of scenario design, another useful addition would be the ability to explicitly define a sequence of waypoints to the pedestrian's destination. This would allow testers to force a certain path across a roadway (that may not include a crosswalk) to better account for jaywalking and other unusual behaviours. Finally, our last suggested improvement involves a systematic search for optimal parameters of the model. For functionality such as corner cutting behaviour and control when a pedestrian enters the crosswalk with a yellow state, a manual search was performed for sufficiently optimal parameter values. However, it could be valuable to devise a process that automates the search for these parameter values and saves the time of future researchers and test engineers.

7.2.2 Future Research Directions

Aside from direct improvements to the model, our work motivates various different research avenues. Given an environment where pedestrians do not have right-of-way (e.g. popular jaywalking section), a completely different set of conditions and maneuvers are required to model pedestrians' expected behaviours. This raises the interesting research question of which environmental factors are influential in a pedestrian's crossing decision in these situations and how do these factors vary across different pedestrians? On a different topic, our model generates sequences of maneuvers and conditions throughout the journey of a model pedestrian. Identifying trends or common sequences could lend itself to prediction tasks of estimating the likelihood of an observed pedestrian's future maneuvers or actions.

Our work opens up opportunities to advance the testing of AVs and automated driving systems (ADS). In particular, our approach provides flexibility in creating extremely specific test scenarios involving pedestrians. In these scenarios, pedestrians may temporarily cause a sudden, but necessary, variation in the ego vehicle's behaviour. In this case, the AV should still be expected to achieve their scenario goal despite the unexpected disturbance. It is also crucial to target and test AVs' responses to critical scenarios. The National Highway Traffic Safety Administration (NHTSA) provides extensive crash reports which are an invaluable source for critical situations resulting in an accident. The data of the crash reports could be leveraged with the NHTSA pre-crash scenarios [53] involving pedestrians to create valuable test scenarios on which to test AVs' critical responses. We leave the task of applying our model to the process of setting up critical scenarios and ensuring their repeatability to future researchers.

Additionally, in order to ensure efficient testing of autonomous vehicles, it is important to consider what aspects of pedestrian behaviours are relevant to the AV. While a model may be able to represent a wide range of complex behaviours, if these behaviours are not relevant to the decision-making processes of AVs, incorporating them into scenario-based testing may impede the progress of AV development.

We encourage future research on these topics and others not listed since human behaviour is far from fully understood and the need for its study is expedited with the rapid on-going development of automated driving systems and autonomous vehicles.

References

[1] DataFromSky. https://datafromsky.com.

[2] Java Open Street Map Editor. https://josm.openstreetmap.de.

[3] Ramiro A. Agis, Sebastian Gottifredi, and Alejandro J. García. An event-driven behavior trees extension to facilitate non-player multi-agent coordination in video games. *Expert Systems with Applications*, 155:113457, 2020.

[4] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[5] Anelia Angelova, Alex Krizhevsky, Vincent Vanhoucke, Abhijit Ogale, and Dave Ferguson. Real-time pedestrian detection with deep network cascades. In *Proceedings of BMVC 2015*, 2015.

[6] André Antakli, Igor Vozniak, Nils Lipp, Matthias Klusch, and Christian Müller. HAIL: Modular agent-based pedestrian imitation learning. In Frank Dignum, Juan Manuel Corchado, and Fernando De La Prieta, editors, *Advances in Practical Applications of Agents, Multi-Agent Systems, and Social Good. The PAAMS Collection*, pages 27–39, Cham, 2021. Springer International Publishing.

[7] Bani Anvari, Michael G. H. Bell, Panagiotis Angeloudis, and Washington Y. Ochieng. Calibration and validation of a shared space model: Case study. *Transportation Research Record*, 2588(1):43–52, 2016.

[8] Bani Anvari, Michael G.H. Bell, Aruna Sivakumar, and Washington Y. Ochieng. Modelling shared space users via rule-based social force model. *Transportation Research Part C: Emerging Technologies*, 51:83–103, 2015.

[9] Shakir Belle, Curtis Gittens, and T.C. Nicholas Graham. Programming with affect: How behaviour trees and a lightweight cognitive architecture enable the development of non-player characters with emotions. In *2019 IEEE Games, Entertainment, Media Conference (GEM)*, pages 1–8, 2019.

[10] Andrew Best, Sahil Narang, Daniel Barber, and Dinesh Manocha. AutonoVi: Autonomous vehicle planning with dynamic maneuvers and traffic constraints. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2629–2636, 2017.

[11] Frédéric Bouchard. Expert system and a rule set development method for urban behaviour planning. Master's thesis, University of Waterloo, 2020.

[12] S. Bouzat and M. N. Kuperman. Game theory in models of pedestrian room evacuation. *Phys. Rev. E*, 89:032806, Mar 2014.

[13] C Burstedde, K Klauck, A Schadschneider, and J Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295(3):507–525, 2001.

[14] F. Camara, O. Giles, R. Madigan, M. Rothmüller, P. Holm Rasmussen, S. A. Vendelbo-Larsen, G. Markkula, Y. M. Lee, Laura Garach, N. Merat, and C. W. Fox. Predicting pedestrian road-crossing assertiveness for autonomous vehicle control. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2098–2103, 2018.

[15] F Camara, R Romano, G Markkula, R Madigan, N Merat, and C Fox. Empirical game theory of pedestrian interaction for autonomous vehicles. In R Grant, T Allen, A Spink, and M Sullivan, editors, *Measuring Behavior 2018: 11th International Conference on Methods and Techniques in Behavioral Research*, pages 238–244. Manchester Metropolitan University, June 2018. This is an author produced version of a paper published in Proceedings of Measuring Behavior 2018: 11th International Conference on Methods and Techniques in Behavioral Research.

[16] Fanta Camara, Serhan Cosar, Nicola Bellotto, Natasha Merat, Charles Fox, et al. Continuous game theory pedestrian modelling method for autonomous vehicles. 2020.

[17] Vicent Ortiz Castelló, Omar del Tejo Catalá, Ismael Salvador Igual, and Juan-Carlos Perez-Cortes. Real-time on-board pedestrian detection using generic single-stage algorithms and on-road databases. *International Journal of Advanced Robotic Systems*, 17(5):1729881420929175, 2020.

[18] Chen Chai, Yiik Diew Wong, Meng Joo Er, and Evan Tat Meng Gwee. Fuzzy cellular automata models for crowd movement dynamics at signalized pedestrian crossings. *Transportation Research Record*, 2490(1):21–31, 2015.

[19] Qianwen Chao, Zhigang Deng, and Xiaogang Jin. Vehicle–pedestrian interaction for mixed traffic simulation. *Computer Animation and Virtual Worlds*, 26(3-4):405–412, 2015.

[20] Thomas Eiter and Heikki Mannila. Computing discrete Fréchet distance. Technical report, Citeseer, 1994.

[21] Epic Games. Unreal Engine.

[22] Zhijie Fang, David Vázquez, and Antonio M. López. On-board detection of pedestrian intentions. *Sensors*, 17(10), 2017.

[23] Shumin Feng, Ning Ding, Tao Chen, and Hui Zhang. Simulation of pedestrian flow based on cellular automata: A case of pedestrian crossing street at section in China. *Physica A: Statistical Mechanics and its Applications*, 392(13):2847–2859, 2013.

[24] David Gerónimo, Angel D. Sappa, Daniel Ponsa, and Antonio M. López. 2D–3D-based on-board pedestrian detection system. *Computer Vision and Image Understanding*, 114(5):583–595, 2010. Special issue on Intelligent Vision Systems.

[25] Michael Goldhammer, Sebastian Köhler, Stefan Zernetsch, Konrad Doll, Bernhard Sick, and Klaus Dietmayer. Intentions of vulnerable road users—detection and forecasting by means of machine learning. *IEEE Transactions on Intelligent Transportation Systems*, 21(7):3035–3045, 2020.

[26] Junbiao Guan, Kaihua Wang, and Fangyue Chen. A cellular automaton model for evacuation flow using game theory. *Physica A: Statistical Mechanics and its Applications*, 461:655–661, 2016.

[27] Ren-Yong Guo. New insights into discretization effects in cellular automata models for pedestrian evacuation. *Physica A: Statistical Mechanics and its Applications*, 400:1–11, 2014.

[28] Dirk Helbing, Illés Farkas, and Tamás Vicsek. Simulating dynamical features of escape panic. *Nature*, 407(6803):487–490, Sep 2000.

[29] Dirk Helbing and Péter Molnár. Social force model for pedestrian dynamics. *Phys. Rev. E*, 51:4282–4286, May 1995.

[30] Lin Huang, Jianhua Gong, Wenhang Li, Tao Xu, Shen Shen, Jianming Liang, Quanlong Feng, Dong Zhang, and Jun Sun. Social force model-based group behavior simulation in virtual geographic environments. *ISPRS International Journal of Geo-Information*, 7(2), 2018.

[31] Yingfan Huang, Huikun Bi, Zhaoxin Li, Tianlu Mao, and Zhaoqi Wang. STGAT: Modeling spatial-temporal interactions for human trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[32] Damian Isla. Building a better battle. In *Game Developers Conference, San Francisco*, volume 32, 2008.

[33] Fatema T. Johora, Dongfang Yang, Jörg P. Müller, and Ümit Özgüner. On the generalizability of motion models for road users in heterogeneous shared traffic spaces, 2021.

[34] Taras I. Lakoba, D. J. Kaup, and Neal M. Finkelstein. Modifications of the Helbing-Molnár-Farkas-Vicsek social force model for pedestrian evolution. *SIMULATION*, 81(5):339–352, 2005.

[35] Maziyar Layegh, Babak Mirbaha, and Amir Abbas Rassafi. Modeling the pedestrian behavior at conflicts with vehicles in multi-lane roundabouts (a cellular automata approach). *Physica A: Statistical Mechanics and its Applications*, 556:124843, 2020.

[36] Jaedong Lee, Jungdam Won, and Jehee Lee. Crowd simulation by deep reinforcement learning. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games*, MIG '18, New York, NY, USA, 2018. Association for Computing Machinery.

[37] Jooyong Lee, Taewan Kim, Jin-Hyuk Chung, and Jinho Kim. Modeling lane formation in pedestrian counter flow and its effect on capacity. *KSCE Journal of Civil Engineering*, 20(3):1099–1108, 2016.

[38] Chuanyao Li, Shikun Liu, and Xuekai Cen. Safety and efficiency impact of pedestrian–vehicle conflicts at non signalized midblock crosswalks based on fuzzy cellular automata. *Physica A: Statistical Mechanics and its Applications*, 572:125871, 2021.

[39] Dewei Li and Baoming Han. Behavioral effect on pedestrian evacuation simulation using cellular automata. *Safety Science*, 80:41–55, 2015.

[40] Kunming Li, Stuart Eiffert, Mao Shan, Francisco Gomez-Donoso, Stewart Worrall, and Eduardo Nebot. Attentional-GCNN: Adaptive pedestrian trajectory prediction towards generic autonomous vehicle use cases. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 14241–14247, 2021.

[41] Kunming Li, Mao Shan, Karan Narula, Stewart Worrall, and Eduardo Nebot. Socially aware crowd navigation with multimodal pedestrian trajectory prediction for autonomous vehicles. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8, 2020.

[42] Xiang Li and Jian-Qiao Sun. Studies of vehicle lane-changing to avoid pedestrians with cellular automata. *Physica A: Statistical Mechanics and its Applications*, 438:251–271, 2015.

[43] Xiaomeng Li, Xuedong Yan, Xingang Li, and Jiangfeng Wang. Using cellular automata to investigate pedestrian conflicts with vehicles in crosswalk at signalized intersection. *Discrete Dynamics in Nature and Society*, 2012:287502, Nov 2012.

[44] Chong-U Lim, Robin Baumgarten, and Simon Colton. Evolving behaviour trees for the commercial game DEFCON. In Cecilia Di Chio, Stefano Cagnoni, Carlos Cotta, Marc Ebner, Anikó Ekárt, Anna I. Esparcia-Alcazar, Chi-Keong Goh, Juan J. Merelo, Ferrante Neri, Mike Preuß, Julian Togelius, and Georgios N. Yannakakis, editors, *Applications of Evolutionary Computation*, pages 100–110, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[45] Guan-Wen Lin and Sai-Keung Wong. Evacuation simulation with consideration of obstacle removal and using game theory. *Phys. Rev. E*, 97:062303, Jun 2018.

[46] Manxia Liu, Weiliang Zeng, Peng Chen, and Xuyi Wu. A microscopic simulation model for pedestrian-pedestrian and pedestrian-vehicle interactions at crosswalks. *PLOS ONE*, 12(7):1–23, 07 2017.

[47] Yangzexi Liu, Jingqiu Guo, John Taplin, and Yibing Wang. Characteristic analysis of mixed traffic flow of regular and autonomous vehicles using cellular automata. *Journal of Advanced Transportation*, 2017:8142074, Oct 2017.

[48] S.M. Lo, H.C. Huang, P. Wang, and K.K. Yuen. A game theory based exit selection model for evacuation. *Fire Safety Journal*, 41(5):364–369, 2006.

[49] Yuexin Ma, Xinge Zhu, Sibo Zhang, Ruigang Yang, Wenping Wang, and Dinesh Manocha. TrafficPredict: Trajectory prediction for heterogeneous traffic-agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):6120–6127, Jul. 2019.

[50] Umberto Michieli and Leonardo Badia. Game theoretic analysis of road user safety scenarios involving autonomous vehicles. In *2018 IEEE 29th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, pages 1377–1381, 2018.

[51] Adam Millard-Ball. Pedestrians, autonomous vehicles, and cities. *Journal of Planning Education and Research*, 38(1):6–12, 2018.

[52] Mehdi Moussaïd, Niriaska Perozo, Simon Garnier, Dirk Helbing, and Guy Theraulaz. The walking behaviour of pedestrian social groups and its impact on crowd dynamics. *PLOS ONE*, 5(4):1–7, 04 2010.

[53] W. G. Najm, John D. Smith, and Mikio Yanagisawa. Pre-Crash Scenario Topology for Crash Avoidance Research. Technical report, U.S. Department of Transportation, NHTSA, April 2007.

[54] Mirco Nanni and Dino Pedreschi. Time-focused clustering of trajectories of moving objects. *J. Intell. Inf. Syst.*, 27:267–289, 11 2006.

[55] Payam Nasernejad, Tarek Sayed, and Rushdi Alsaleh. Modeling pedestrian behavior in pedestrian-vehicle near misses: A continuous gaussian process inverse reinforcement learning (GP-IRL) approach. *Accident Analysis & Prevention*, 161:106355, 2021.

[56] Ishaan Paranjape, Abdul Jawad, Yanwen Xu, Asiiah Song, and Jim Whitehead. A modular architecture for procedural generation of towns, intersections and scenarios for testing autonomous vehicles. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 162–168, 2020.

[57] L.A. Pereira, D. Burgarelli, L.H. Duczmal, and F.R.B. Cruz. Emergency evacuation models based on cellular automata with route changes and group fields. *Physica A: Statistical Mechanics and its Applications*, 473:97–110, 2017.

[58] Benedetto Piccoli and Andrea Tosin. Time-evolving measures and macroscopic modeling of pedestrian flow. *Archive for Rational Mechanics and Analysis*, 199(3):707–738, 2011.

[59] Fabian Poggenhans, Jan-Hendrik Pauls, Johannes Janosovits, Stefan Orf, Maximilian Naumann, Florian Kuhnt, and Matthias Mayr. Lanelet2: A high-definition map framework for the future of automated driving. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1672–1679, 2018.

[60] Yong-Sheng Qian, Xiao Feng, and Jun-Wei Zeng. A cellular automata traffic flow model for three-phase theory. *Physica A: Statistical Mechanics and its Applications*, 479:509–526, 2017.

[61] Rodrigo Queiroz. 2022.

[62] Rodrigo Queiroz, Thorsten Berger, and Krzysztof Czarnecki. GeoScenario: An open DSL for autonomous driving scenario representation. In *IEEE Intelligent Vehicles Symposium (IV)*, Paris, 2019. IEEE, IEEE.

[63] Yan qun Jiang, Peng Zhang, S.C. Wong, and Ru xun Liu. A higher-order macroscopic model for pedestrian flows. *Physica A: Statistical Mechanics and its Applications*, 389(21):4623–4635, 2010.

[64] N. Rinke, C. Schiermeyer, F. Pascucci, V. Berkhahn, and B. Friedrich. A multi-layer social force approach to model interactions in shared spaces using collision prediction. *Transportation Research Procedia*, 25:1249–1267, 2017. World Conference on Transport Research - WCTR 2016 Shanghai. 10-15 July 2016.

[65] Ismael Sagredo-Olivenza, Pedro Pablo Gómez-Martín, Marco Antonio Gómez-Martín, and Pedro Antonio González-Calero. Trained behavior trees: Programming by demonstration to support AI game designers. *IEEE Transactions on Games*, 11(1):5–14, 2019.

[66] Dongmei Shi, Wenyao Zhang, and Binghong Wang. Modeling pedestrian evacuation by means of game theory. *Journal of Statistical Mechanics: Theory and Experiment*, 2017(4):043407, apr 2017.

[67] Xiaodan Shi, Xiaowei Shao, Zhiling Guo, Guangming Wu, Haoran Zhang, and Ryosuke Shibasaki. Pedestrian trajectory prediction in extremely crowded scenarios. *Sensors*, 19(5), 2019.

[68] Wonho Suh, Dwayne Henclewood, Aaron Greenwood, Angshuman Guin, Randall Guensler, Michael P Hunter, and Richard Fujimoto. Modeling pedestrian crossing activities in an urban environment using microscopic traffic simulation. *SIMULATION*, 89(2):213–224, 2013.

[69] Li Sun, Zhi Yan, Sergi Molina Mellado, Marc Hanheide, and Tom Duckett. 3DOF pedestrian trajectory prediction learned from long-term autonomous mobile robot deployment data. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5942–5948, 2018.

[70] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10400–10409, June 2021.

[71] Elizabeth Swanson, Mikio Yanagisawa, Wassim Najm, Frank Foderaro, and Philip Azeredo. Crash avoidance needs and countermeasure profiles for safety applications based on light-vehicle-to-pedestrian communications. August 2016.

[72] Jun Tanimoto, Aya Hagishima, and Yasukaka Tanaka. Study of bottleneck effect at an emergency evacuation exit using cellular automata model, mean field approximation analysis, and game theory. *Physica A: Statistical Mechanics and its Applications*, 389(24):5611–5618, 2010.

[73] Kardi Teknomo. Microscopic pedestrian flow characteristics: Development of an image processing data collection and simulation model. *CoRR*, abs/1610.00029, 2016.

[74] Emmett Tomai, Rosendo Salazar, and Roberto Flores. Simulating aggregate player behavior with learning behavior trees. In *Proceedings of the 22nd Annual Conference on Behavior Representation in Modeling & Simulation*, 2013.

[75] Paul Torrens, Xun Li, and William A. Griffin. Building agent-based walking models by machine-learning on diverse databases of space-time trajectory samples. *Transactions in GIS*, 15(s1):67–94, 2011.

[76] Brooke R Ullman, Marcus A Brewer, Kay Fitzpatrick, and Gerald L Ullman. Investigating pedestrian components in temporary traffic control. 2007.

[77] Qian Wang, Beshah Ayalew, and Thomas Weiskircher. Predictive maneuver planning for an autonomous vehicle in public highway traffic. *IEEE Transactions on Intelligent Transportation Systems*, 20(4):1303–1315, 2019.

[78] WenJing Wu, RunChao Chen, Hongfei Jia, Yongxing Li, and ZhiKang Liang. Game theory modeling for vehicle–pedestrian interactions and simulation based on cellular automata. *International Journal of Modern Physics C*, 30(04):1950025, 2019.

[79] Yinhua Xia, S. C. Wong, and Chi-Wang Shu. Dynamic continuum pedestrian flow model with memory effect. *Phys. Rev. E*, 79:066113, Jun 2009.

[80] Hao Xue, Du Q. Huynh, and Mark Reynolds. SS-LSTM: A hierarchical LSTM model for pedestrian trajectory prediction. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1186–1194, 2018.

[81] Mikio Yanagisawa, Elizabeth Swanson, Philip Azeredo, and Wassim Najm. Estimation of potential safety benefits for pedestrian crash avoidance/mitigation systems. April 2017.

[82] Dongfang Yang, Keith Redmill, and Umit Ozguner. A multi-state social force based framework for vehicle-pedestrian interaction in uncontrolled pedestrian crossing scenarios, 2020.

[83] Xiao Yang, Binxu Wang, and Zheng Qin. Floor field model based on cellular automata for simulating indoor pedestrian evacuation. *Mathematical Problems in Engineering*, 2015:820306, Oct 2015.

[84] Weifeng Yuan and Kang Hai Tan. An evacuation model using cellular automata. *Physica A: Statistical Mechanics and its Applications*, 384(2):549–566, 2007.

[85] Weiliang Zeng, Hideki Nakamura, and Peng Chen. A modified social force model for pedestrian behavior simulation at signalized crosswalks. *Procedia - Social and Behavioral Sciences*, 138:521–530, 2014. The 9th International Conference on Traffic and Transportation Studies (ICTTS 2014).

[86] Han-Tao Zhao, Xin-Ru Liu, Xiao-Xu Chen, and Jian-Cheng Lu. Cellular automata model for traffic flow at intersections in internet of vehicles. *Physica A: Statistical Mechanics and its Applications*, 494:40–51, 2018.

[87] Han-Tao Zhao, Shuo Yang, and Xiao-Xu Chen. Cellular automata model for urban road traffic flow considering pedestrian crossing street. *Physica A: Statistical Mechanics and its Applications*, 462:1301–1313, 2016.

[88] Zhuping Zhou, Yang Zhou, Ziyuan Pu, and Yongneng Xu. Simulation of pedestrian behavior during the flashing green signal using a modified social force model. *Transportmetrica A: Transport Science*, 15(2):1019–1040, 2019.

[89] Cheng-hua ZHU, Miao CHEN, Duan-feng HAN, and Yue QIAO. Research on multiple exits evacuate based on game theory and social force. *Computer Simulation*, page 11, 2014.