POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Master's Degree Thesis

Modeling the behaviour of light electric vehicles in urban environments

Supervisors
Prof. Pangcheng David CEN CHENG
Prof. Anne Spalanzani

Candidate
Alessio Masi

April 2025 - Academic Year 2024-2025

Acknowledgements

I want to thank all of the Chroma team Grenoble for helping me during my stay, my supervisors Anne Spalanzani and Pangcheng David Cen Cheng for the help and the guidance they gave to me during the redaction of this thesis.
On top of this, I cannot forget to mention my dear friends who always supported me and always reminded me of my value and helped me navigate through hard moments with their strength Tommaso, Giorgio, Niccolo, Matteo, Duccio and Filippo. You welcomed me in your group in what was a very difficult moment for me and I never felt as a foreign body in the group bur rather accepted and involved in everything you have done and I am very glad that we cross our paths. Even if I never tell you I think very high of all of you and I would never trade your friendship for anything in this world, so I hope I will never lose you because it would feel as losing a family member.
Another important mention goes to the small group that remained of our high-school class, Francesco, Tommaso, Leonardo and Lorenzo (for the others I mentioned you already in the previous part, do not start to complain) with most of you I shared an important part of this journey and even though we took different journeys we stayed in touch and I have magnificent memories with all of you and I think we gave each other unforgettable moments. Then I have to mention what I can call the train group, Filippo, Niccolo and Thomas, since I think most of the funniest moments we shared where during the beautiful time we spent on the train to get to the university. I started my degree alone, after switching from another degree and I found you, almost by accident, but we formed immediately a very good group and even if we lost a bit each other on our way to our bachelor degree we still have incredible memories. I want to spend a few more words for you Thomas, since as I said before we found each other a bit randomly but in the end we were the survivors, and you were exactly what I needed. After that dark period COVID was, I found in you someone whom I could share everything with, we spent a lot of time together laughing, complaining and getting crazy about university but I think we formed a very good bond and I am pretty sure that I can defined you as a very good friend and a person that I will not forget in the future.
Also a mention to all the beautiful people that I met during my Erasmus and these 18 months. It is impossible to mention you all so I will just give you a general thanks for everything, you have been an important part of this journey and you made me grow a lot. But I want to thank in particular some of you which have been there for a lot of time, Felix, Michele, Loukas, Julia and Linnea, I learnt a lot from you, new cultures, way of thinking and I am sure that I will miss you a lot since you have been a fundamental part of my life for quite a long time. Finally, I cannot forget my flatmates, Angelo, Luis

and Arturo. Considering I was not even supposed to be there with you I think in the end I can define myself a very luck guy to have shared my days, months with you. I cannot forget the walks to the supermarket with the question about what would be the next surprise our magnificent neighbourhood had for us, the hikings we have done, the beers and the time we spent together. I am incredibily greatful to have met you and thanks to you even in those periods wher I was a bit more down due to the university load or personal things, I have always feel supported and light-hearted.
In the end, a huge thanks to all my family who together with my friends have been a pillar during all these academic years and have always believed in me and never let me down. Thanks, Mum for everything you have done for me, I know you have sacrificed a lot but if I am what I am today is also thanks to you and you should be proud of yourself, because I am, even if I do not show it too much. Dad, I know sometimes we have discussions or disagreemnts, but I think is important for a relationship to have those as well and even if you might think the opposite it shows how much I care about you, your opinion and your general well-being. Because even if you do not think so, you have pushed me a lot more in moments where I think I could not achieve what I have and you gave me fuel and rage to overcome obstacles and show to myself that I am capable of everything. Thanks to Nonno for always being there for me, you are very important to me and even if you don't know I learnt a lot of things from you and you have are an inspiration and a role model for me. Thanks to Nonna, because we shared some difficult moments, you are a strong woman who showed me how to react and behave so strongly in certain situations and your stregth will always be something I am going to look up for. Thanks to Zia Pia and Zio David, zia you have always believed in me no-matter what I wanted to do and tried to support me as you could, and you know this will always be appreciated. Thanks to Zia Rita and Zio Marco as well, because even if, we cannot hide it, the distance has prevented our relationship to develop as I wanted I can count on you no-matter what and I know you have always supported my choices and you would be there if I needed. Thanks to my beautiful cousins as well, Samantha and Gabriele (and the little Emily as well). For you goes the same as for your parents, the distance has prevented the relationship to develop as it could have, but so you know when I was younger (and now as well, do not worry) I was looking at you as role models, as people I thought really high of, so do not underestimate the roles you have had in my development.
Finally, the last thanks goes to all the people who will never read this thesis and hence will never know about this, but I would like to thank also a lot of streamers and youtubers, since they have been, in some moments, the only source of joy I had. I could count on you at any time to give me a smile or a laugh when I was feeling down, or fill the void I was feeling inside me, even if you do not know you have been critical for me.

Abstract

In recent years there has been a significant increase in the number of personal mobility vehicles (PMVs) of all kinds, used in urban scenarios. In this context, autonomous cars will encounter numerous problems due to their perception limitations and sometimes, unpredictable behaviours of PMV users. Therefore, it is particularly important to study the interaction between autonomous cars and those vehicles. Among PMVs, e-scooters have become increasingly popular. Despite this, very little attention has been given to studying their behaviour in shared spaces. This work aims to develop a behavioural model for e-scooters that can be integrated into the existing autonomous vehicles simulator SPACiSS. The behavioural model was created using the Social Force Model (SFM), adapting it to handle the different characteristics an e-scooter has with respect to classic SFM models. The updated model will be able to handle physical constraints such as the dynamic and kinematic due to the non-holonomicity of the vehicle as well as implementing a new defined force that models road elements such as lanes and sidewalks. The model was integrated into an ROS-based environment that handles the communication between the simulation and the visualisation process, as well as rendering the different scenarios and their agents. Due to the lack of real-world datasets, which are crucial to obtaining an accurate model, the calibration and the tests will be discussed only theoretically. Despite this, a crash analysis will be conducted. It has been possible because it was all on simulation, but it has proved fundamental to highlight the scenarios where the model has more difficulties and where it performs nicely. Despite the challenges encountered due to the lack of datasets, the model still proved to be effective and was able to simulate a large range of real-world situations. This work contributes to developing a behavioural model for an e-scooter that enriches the field of autonomous vehicle research. Future work should be focused on validating the model presented here with real-world data and expanding it to other personal mobility vehicles.

Contents

Chapter 1

Introduction

This chapter will present this thesis work, starting by understanding the context in which it nestles and going then to the goals this project proposes to achieve. We will then propose the solution we intend to implement to then conclude the introduction with a preamble for the chapters that have to come.

1.1 Context

The field of autonomous cars has been expanding continuously over the recent years, trying to go in the direction of more capable models. If we consider the starting point that was in 1980, there has been a huge leap forward, since now there are a few commercial cars capable of self-driving under certain conditions. At the same time, this is currently one of the biggest limitations, since what we are able to provide as of now, are cars that can drive without the intervention of the driver on highways or where the road condition is known as well as the weather condition.
If we look at what we would need in order to have a fully autonomous car, one of the hardest but at the same time most encountered scenarios, is driving in an urban context. This can lead to a various number of challenges, since in those cases the car has not only to navigate among cars but also will come across other road users such as pedestrians, bicycle riders, e-scooter riders, etc. These kinds of users are much more vulnerable than other cars, and that is why in this context they are also referred to as Vulnerable Road Users (VRU). When moving among VRUs, the car will encounter a large number of different behaviours which could also be unexpected, since it's not uncommon for pedestrians to cross the road not following the general road rules, but rather trying to find the shortest way to reach their goal. The same can be experienced with other vehicles such as Personal Mobility Vehicles (PMVs). Here we can also have unexpected behaviours such as crossing an intersection or a traffic light even if it is not their turn or ride on the road even if there are predefined paths for them. All these are complications when applied to the algorithms that control autonomous cars.
In particular, if we focus on a branch of the PMVs that is represented by the Powered PMVs, we can identify an increasing number of e-scooters in recent years. This has affected most large cities, thanks also to the sharing services that provide and offer this

type of vehicle on the streets. Furthermore, the easiness of use that these vehicles provide is also another strong point, together with the dimensions which are very close to those of a bike and guarantee facility in transportation and storing.
The upraise of e-scooters is a new challenge in the research spectrum of autonomous cars since it is a very limited explored branch. They can be easily compared to bicycles since they are both two-wheeled, but in fact there are some important differences that need to be accounted for such as the lower centre of mass and the riding position which are two important factors particularly when modelling the physical model of such a vehicle. From the behavioural side, the key point is the difference in acceleration between the two which can lead to different attitudes.
All that has been discussed until now are only some of the reasons why it is important to focus the research on finding new ways to model the behaviour of road users, particularly those which are unpredictable. Another important aspect to consider is how to test the navigation algorithms that are deployed. This is because conducting field tests other than being expensive, is also time and resources consuming, and might not always be possible to conduct such kinds of tests. This is why putting effort into researching a way to test the algorithms in other ways is fundamental. A good answer to this problem is given by the simulators, which are built to represent as realistically as possible the different scenarios that a car may encounter when navigating on the roads, despite the oxymoron that the association of these two worlds might represent.
That is why a research institute such as INRIA has decided to fund a project that goes exactly in this direction called ANNAPOLIS (AutoNomous Navigation Among Personal mObility devIceS) [5]. The project aims to increase the vehicle's perception capacity in terms of precision, measurement field of view and information semantics, through vehicle to intelligent infrastructure communication. The project will also seek new models or concepts to take into account unpredictable behaviours of the new means of individual electric transport, to interpret and analyze scenes under constant evolution, and finally to decide the best future and safe motion of the self-driving car even in highly dynamic environments with unexpected and dangerous events. The last point is where this work is positioned and it describes which problem it aims to solve.

1.2 Research Problem

As we have seen in the previous section, there are multiple challenges to be addressed to keep developing the field of autonomous car navigation. With this work, we decided to focus on the urban navigation problems and in particular those situations where the car has to move in crowded spaces or in shared spaces with other VRUs. This area itself has also a lot of different approaches that can be taken in order to tackle the challenge, and our direction was to study more the interaction of the car with e-scooters.
This is particularly interesting because as we have seen the number of users has continuously increased throughout the years and is expected to continue on this trend even in the years coming [6] [37]. Besides, the fact that this is a relatively new branch without a lot of research done, gives us freedom to explore while enriching an important part of the field.

Given this, we decided to investigate in particular the branch of behavioural models applied to e-scooters, in order to be able to put it in a simulator that could be used to test navigation algorithms in scenarios that include multiple VRUs. This specific subsection is interesting because there are only a few models proposed and none of them focus on the vehicle's interaction with a car in shared spaces that is exactly the gap that we want to fill with this work.
The objectives for this thesis can be summarized as follows:

- Design a behavioural model of an e-scooter
- Integrate it inside a simulator also comprised of other agents called SPACiSS
- Have the possibility to simulate different rider behaviours
- Enrich the scenarios available in the existing simulator SPACiSS by adding other road elements

1.3 Research Methodology

To conduct this work, we followed a standard research approach. Firstly, we dived into the literature to learn more about the field of simulators and behavioural models. Consequently, we tried to understand what could be the correct PMV candidate to be implemented in the simulator. After choosing the e-scooter as a vehicle, due to its spread use among people and the little work that we found in our literature review, we started digging looking for other studies that had focused on designing behavioural models for this kind of PMV.
To implement the e-scooter model we have proposed to use a modified version of the Social Force Model (SFM). This model is widely used in the literature to represent the interaction between different agents in a social context. We have defined a modified version of this, adapted to better suit and represent an e-scooter. This revised version of the social force is the main contribution of this thesis work, among the proposed formulation for lanes and sidewalks. This was essential to fulfil the goal of having more various scenarios and has also added a layer of complexity and possible analysis to the simulator.
Once the model was implemented we started with the calibration phase, so we decided which algorithm was meant to be used and how to validate the model after the calibration. Unfortunately, due to the lack of real-world datasets including e-scooters that could be used to calibrate and test the model, we decided to still design the test scenario to be used once the dataset becomes available. To find a way to test the model despite the lack of data, we performed a crash analysis in simulation that gave us a lot of insights into the model performances and areas that can be improved, even without a proper parameter calibration.

1.4 Thesis Overview

To conclude the introduction we will go through the next chapters summarizing their content and giving a brief peak into what we will see in the upcoming chapters.

1.4.1 Chapter 2 - State-of-the-art

The state-of-the-art chapter will present what is the current situation in the context of this work and will detail the rationale behind the choices we made for what regards the model and simulator. We will start by analyzing the existing simulators for autonomous vehicles, that are used to test navigation algorithms. We will focus on their strengths and weaknesses, and what are their main functionalities, while also giving a shallow introduction to their structure and their main purposes. After the discussion on simulators, the focus will shift to presenting the most common behavioural models that are used in modelling the interaction of agents such as pedestrians between each other. We will analyze three of the most used models, which are the Social Force Model, the Cellular Automata and those models based on Neural Networks. For each of these models, we will look at the principles behind them and how they work. We will understand what has been achieved in the literature using those models and which behaviours they manage to emulate. Finally, we are going to tighten the field of view, looking for applications of such models to e-scooters specifically. What we found analyzing the literature, is that as of now there have been proposed only models utilizing the Social Force Model and so we will focus on analyzing those and how they shaped the social forces to be applied to an e-scooter.
Eventually, we will explain the reasons behind our choices for what concerns the simulator and the model used in the production of this thesis. In the end, we decided to use SPACiSS as a simulator which was developed internally for the laboratory by Prédhumeau et al. [32], in order to enrich a simulator that had already filled the gaps for what concern pedestrians' interaction with an autonomous car in shared spaces, respect to the others in the literature such as CARLA or LGSVL. We also ended up using the Social Force Model because we wanted to give continuity to what was already implemented in the simulator and because it is a widely recognized and used model to represent this kind of interaction.

1.4.2 Chapter 3 - System modelling: E-scooter, road lanes, sidewalks and social force formulation

This chapter delves into the mathematical formulation of the model we have implemented in the simulator. It is fundamental to present the main contributions of this thesis and is the core of the project.
Following the research work done in revising the literature and the few approaches we found, to come up with the model we have used, we decided to use as a foundation the work of Liu et al. [25] which gave us the main elements to consider in the building of our social force formulation. The main modifications produced to the classical formula regard the inclusion of different shapes for the personal space of the e-scooter rider; considering a velocity-dependency when computing the distance between two neighbour agents and considering the kinematic and dynamic constraints that are present when dealing with a non-holonomic vehicle such as an electric scooter.
The personal space has been designed as an ellipse that will change its dimension dynamically based on the perceived density and the agent's velocity. The velocity-dependency

has become necessary since in the simulation there are agents going at different velocities and their relative velocity is not negligible. Thus it must be taken into consideration when computing the relative distance between agents. Finally, the kinematic constraints are fundamental to be implemented since the e-scooter cannot translate laterally but has limitations when turning, and so we present both a kinematic and dynamic bicycle model from where we derive the formulas to compute the updated position and orientation of the vehicle.
To conclude this chapter we present our model of the lanes and sidewalks, which are introduced within the simulator, fulfilling one of the goals of this project. The force that we designed for the sidewalk is capable of representing behaviours such as crossing a lane to avoid an obstacle or another agent, to successively come back to the correct lane after the avoidance manoeuvre.

1.4.3 Chapter 4 - Implementation

In the implementation we discuss the tools we used in this project and their characteristics. Here we will learn more about the environment we worked in and how the theoretical formulation translates into a practical form. The most important tools that have been used are the Robot Operating System (ROS), with all its internal tools, and the SPACiSS simulator.
ROS is a middleware that has been designed to be used in the robotic environment to handle the different parts of a robot in a distributed manner. Among all the tools that it is equipped with the most significant is rviz. This is a visualization tool that can display the robot and its sensor in a 3-dimensional space. In our work, we have used it to represent the different scenarios that we wanted to simulate including all the agents (pedestrians, AV, e-scooters) and the lanes or the obstacles.
SPACiSS is the simulator that we have decided to use as a base to implement the e-scooter model. It is built using ROS to handle the different parts of the system such as the simulation and the visualization and is built on top of the ROS package pedsim_ros [2], a famous package used to represent the interaction between pedestrians in a crowd using the Social Force Model. The base version can provide different kinds of scenarios with pedestrians interacting with an autonomous car. Among the behaviours this simulator can handle we have the group formation and the stopping behaviour when facing the car in order not to crash with it.
Our implementation is based on this simulator, which was also one of the objectives. We have tried to keep the structure of the simulator unaltered and attempted to integrate our model seamlessly within the simulator. We have modified a few packages in order to include the new formulation of the forces, including the added lane force, as well as the new scenarios we have designed to represent other situations rather than only an urban setting. Lastly, we have implemented a dynamic display of e-scooters that can handle an arbitrary number of e-scooters at runtime displaying all of them without any additional work from the final user when designing a new scenario to test.

1.4.4 Chapter 5 - Evaluation

The evaluation section is where we will critically evaluate and test the model we have designed and implemented. We will present the results of this experimentation and discuss them to understand the criticality and the strengths of this work.
When dealing with a model that uses the SFM one important part is to calibrate all the parameters. This is fundamental to obtain a model that can simulate correctly the behaviour of an agent based on the scenario we want to test and the individual behaviour of the agent we are considering. To do this, we have decided to use a calibration algorithm that uses a genetic algorithm to obtain the calibrated parameters based on an optimization function that will make a comparison between the real data and the simulated one. The biggest problem we have encountered in this phase is that since there is a lack of datasets including data on e-scooters that could be useful for this project, the goal was to create our own dataset designing the scenarios ourselves. This is also a goal of the global project ANNAPOLIS, but unfortunately, the engineers responsible for the data collection did not manage to collect the data on time for us to use them and so we did not have real data to use in our calibration process as well in the tests. This meant that we had to rely on previously calibrated models that we used as a reference for most of the parameters, while for those that were scenario/agent specific, we went for a trial-and-error procedure.
After calibrating the parameters and validating them, the first test we conducted was completely done in simulation, and we did a crash analysis. For the two scenarios that we implemented, we tested different cases and analysed the crashes that occurred to understand first if they were realistic or not and then the cause that could have provoked them.
Ultimately, we also designed a test scenario that was more complex than the other one we used in the calibration and the crash analysis, to test the model in a more complex context, but, as stated before, since we did not have real data to compare with, we could not test it and we just designed it.

1.4.5 Chapter 6 - Conclusion

In conclusion the work developed in this thesis managed to achieve all the goals that were established at the beginning. A behavioural model for an electric scooter has been developed and implemented within the existing simulator SPACiSS. The provided model has proved capable of emulating the normal behaviours of an e-scooter rider, as well as those of an irresponsible one while respecting the physical constraints imposed by the vehicle itself.
Despite the challenge provided by the lack of a dataset to calibrate the parameters, we still have managed to have a good model that behaves accordingly in all the different scenarios we have tested. On the other hand, the lack of data is also a big limitation since no accurate calibration has been conducted as well as more complex tests to verify the validity of the model to simulate even more difficult situations. The future work could be focused on firstly validating the model to then shift the focus to adapt the proposed Social Force Model also to other PMVs to enrich even more the SPACiSS simulator.

Chapter 2

State-of-the-art

In this chapter, we will analyze all the components that are required to understand the context of the thesis, to achieve the objectives. We will start from the actual state-of-the-art in the simulators field, and then focus on the different types of ways to implement the behavioural model of vehicles in a simulator, ending this chapter talking about the specific simulator we worked on and an overview of work related to e-scooter specifically.

2.1 Simulators

There are a lot of different simulators that are already existing but if we focus on those whose objective is testing autonomous vehicles the current state-of-the-art gives us a pair composed by CARLA and LGSVL [17], [35], [28].

2.1.1 SUMO

Simulation of Urban MObility (SUMO) [26] is one of the oldest simulators to test general traffic flow that is still updated and working. It is open-source software that allows the modelling of inter-modal traffic systems including road vehicles, public transport, and pedestrians. Due to this nature, it means that not only car movements are modelled but also trains and buses with the atomic element being the person. Thus, a person may take his car and go to the closest train station to take the train. Walking is not simulated but is modelled by estimating the time it takes to reach the destination.
Like the other simulators that we are going to present later, this also handles microscopic traffic flow simulation, which is one of our interests in this work. Figure 2.1 depicts the environment of SUMO simulator.
The Sumo network consists of nodes and a series of unidirectional edges representing the street, sidewalks and bike lanes. Pedestrians can walk and also ride vehicles and by performing these actions they will interact with other vehicles or pedestrians creating situations where they might impede the movement of the other agents.
Due to its limitations, it can be coupled with other simulators to extend its capabilities and provide a broader range of possibilities to the final user.

Figure 2.1: Images from SUMO simulator [26]

2.1.2 CARLA

Car Learning to Act (CARLA) [17] is an open-source simulator for autonomous driving research. It is built on top of Unreal Engine 4 (UE4) [18]. It simulates the world, as shown in 2.2, and provides a simple interface to interact with it in the form of server-client. The server is the one in charge of rendering the scenes, the simulation and the physics behind every vehicle. On the other side, the client sends commands to interact with the vehicle and the server itself (e.g. accelerating, turning, resetting the simulation).
The environment that it simulates is composed of static objects and dynamic objects. Among the other objects that are not directly controlled by the user, we can have multiple types of cars, pedestrians and other vehicles in the background that can detect and avoid each other, while following road rules, such as traffic lights, intersections etc.
The goals that it aims to accomplish are to guarantee an environment capable of supporting training, evaluation and prototyping of autonomous driving algorithms and solutions that include both perception and control of the vehicle. Based on this, CARLA prefix to give its users a platform to test so, which is composed of a multitude of different scenarios and weather conditions.
Due to its open-source nature, the users keep updating it and extending its functionalities and as of today, its database comprises multiple models of cars, motorcycles and a few bikes, while for what concerns the main topic of this thesis, the simulator still lacks the presence of vehicles such as e-scooters.

Figure 2.2: Images from CARLA simulator [27]

2.1.3 LGSVL

LGSVL [35] is a high-fidelity open-source simulator for autonomous driving. It is built using the Unity game engine [39] which handles the simulation. It is built such that there is a bridge that connects the autonomous driving (AD) stack and the simulation core. It supports the main open-source autonomous driving stack and their messages to communicate over the bridge.
The communication interface changes depending on the stack the user decides to use, in case it implements a custom runtime framework the user will also have to create a custom bridge interface.
The simulation engine's main job, on the other hand, is to take advantage of the powerful game engine that it uses to simulate the environment, the sensors and the vehicle's dynamics and controls. For what concerns the environment, the core can handle different weather conditions and also different times of the day. On top of that, the road has also an impactful meaning on the ego vehicle, which is the one the user is testing, which translates into it being able to follow traffic rules, such as traffic lights, lanes and stop signs; pedestrians also can follow annotated roads. The LGSVL simulation environment can be seen in Figure 2.3
Among its main applications we can find Software In the Loop (SIL) and Hardware In the Loop (HIL) testing of the AD stack, generation of synthetic data to train neural networks, and testing vehicle to everything communication (V2X).
Differently from CARLA, this project has been discontinued and their website is no longer accessible, despite the code still being available and downloadable.

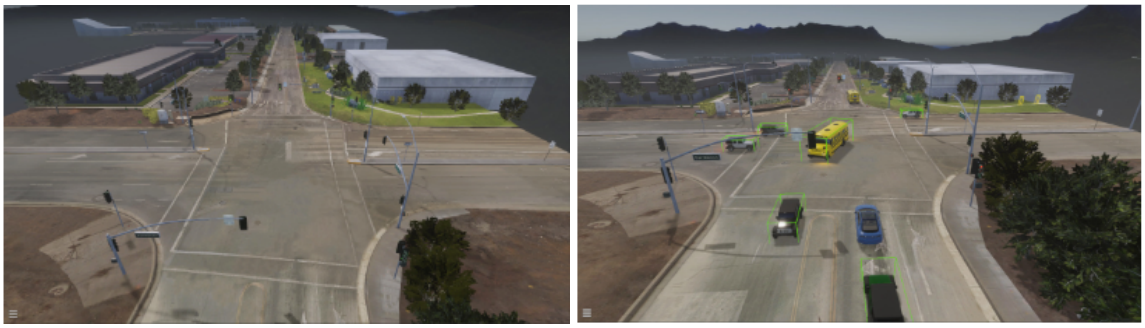


Figure 2.3: Images from LGSVL simulator [35]

2.1.4 SPACiSS

Simulator for Pedestrians and an Autonomous Car in Shared Spaces (SPACiSS) [33] [32] is an open-source simulator to test the interaction of pedestrians and an autonomous vehicle in shared spaces.
It is built on top of *pedsim_ros* [2] which is a package useful in robot navigation experiments with crowded scenes which are hard to acquire in practice. The autonomous vehicle (AV) movements in the simulation can be controlled with ROS [3] commands from an external algorithm, and simulated pedestrians dynamically react to the AV.

The objectives of this simulator and the reasons it has been built are strictly related and are based mainly on the need to have a simulator that integrates advanced pedestrians' behaviour and robotic functionalities for the AV's behaviour. Having these as principal goals what has been achieved is an agent-based model where each pedestrian is represented as an individual agent with its behaviour. There is also the possibility to have a group of pedestrians, all with the same goal and who move together as a whole. Each one of them is also represented as an agent but we also have a more general definition as a set of heterogeneous agents, as shown in Figure 2.4.
The model implemented in the simulator is based on a force model, in particular, called Social Force Model (SFM) [22] and that we are going to see in more detail in Section 2.2.3.
Among the drawbacks of this implementation, we have the lack of variety in the agents that surround the AV, that here is limited to pedestrians only, and also the low diversity of scenarios, which in this case are only represented as empty spaces without any presence of road or other road elements.

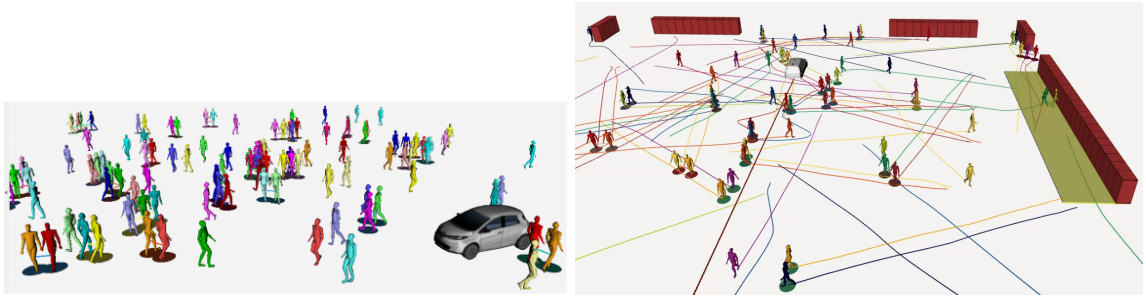


Figure 2.4: Images from SPACiSS simulator [33]

2.2 Behavioural models

To represent agents and model their behaviour inside simulators there are multiple approaches, where each one has its advantages and disadvantages. Among those we can identify three categories that are of particular relevance to the presented work: Cellular Automata [11], Neural Networks [20] and Social Force Model [22].

2.2.1 Cellular Automata

The idea of Cellular Automata (CA) was formalized by John von Neumann in the 1940s [14]. What he first designed had the concept of a grid composed of elements called cells. Each one of those has a value or a state and interfaces with the four cells surrounding it. The assembly of these 5 elements has been called *von Neumann neighbourhood*, as he was the first to use this example. The state of the cells changes at discrete time steps, and the new one is computed based on the old state and a set of predefined rules, which is why CA is in the category of rule-based algorithms. The CA with pedestrians is illustrated

in Figure 2.5. In his first description, each cell had 29 states to choose from.

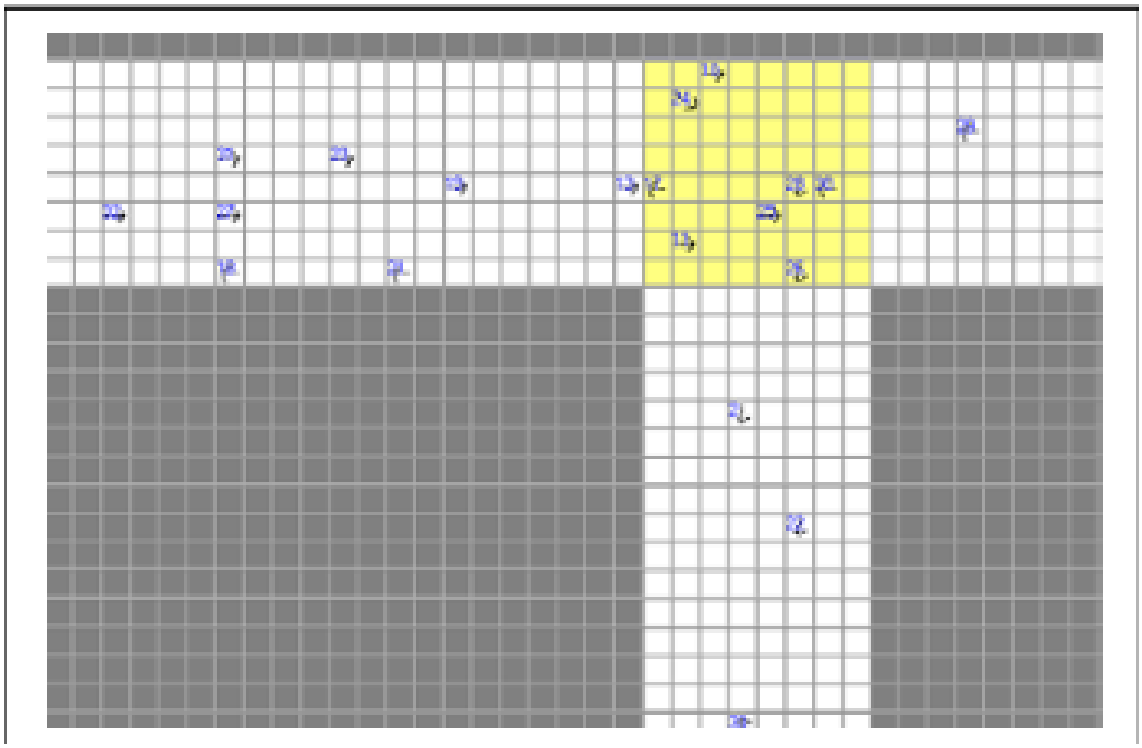


Figure 2.5: Grid describing a CA with pedestrians moving in it [16].

In the years this model has been refined and applied to a multitude of different cases, including also pedestrian and vehicle movements.
If we were to compare the first model to the new ones, we would notice that most of them have a lot in common in the basis since they all use a grid model where the person is placed and it can move in all the cells surrounding it according to rules. Where it comes the difference is in the refinement of the cells and the rules that dictate the movement of the agent. In [11], the size of the cell is $40x40cm^2$ and each pedestrian occupies a whole cell. The speed in case of non-interaction with other agents is of one cell per time step. The pedestrian has 9 options every time it decides to move, which are to stay in its cell or to move in one of the 8 that are surrounding it with a certain probability which is dependent on multiple factors including also the domain of application.
Other examples include a more refined cellular size, to have more accuracy and model in a more "continuous" way the movement of the agents. In [36], they use cells of $5x5cm^2$, and then map the body of the pedestrian to multiple cells, allowing also different types of persons and ages to occupy more or fewer cells. This refinement allows for smoother transition and a higher accuracy.
This model provides many advantages including the low computational complexity compared to other models and a higher speed. On the other side, this is all valid if we can deal with a discrete model, and if we keep the cells size big enough not to affect the space

complexity. For what concerns models that use CA for e-scooters the literature as it is now is still lacking, while the very few models we could find focus on using the social force model.

2.2.2 Neural Networks

The use of neural networks in the last years has increased exponentially and as a consequence their application to various fields. In [42], we can see one use case, to predict trajectories of pedestrians in crowd interaction. Another is to predict the motion of pedestrians at urban intersections [20]. In this context, we can divide the use of neural networks based on the architecture they use. Among those using deep neural networks, we have two clear examples that are Behavior Convolution Neural Networks (BCNN) [44] and Social Long-Short Term Memory (SLSTM) [7]. The first one is based on the walking paths of the pedestrians in the previous frames, which are then encoded into a displaced volume and fed to the Behavior CNN which predicts future displacements. The main problem with this approach is that, alone it is not sufficient to fully represent the interactions that happen at each moment between different people, and how the trajectory and the velocity of a pedestrian can influence those of another. To account for this problem, one possibility could be using an LSTM as the one presented in [7].
In the model they propose, their main objective is to use an LSTM to predict the trajectory of each agent but to solve the problem we had with BCNN, they also want to consider the neighbors' trajectory to refine the prediction. To do so, their idea is to exploit the hidden layers of the LSTM and use them as additional information when computing the new trajectory.

2.2.3 Social Force Model (SFM)

The last model we are going to analyze is the Social Force Model (SFM). Created in 1995 [22], the main objective was to introduce a formulation based on a continuous model that could describe the motion of pedestrians. The fundamental idea is that when walking, or in general moving, in a social environment we act as if we were subject to imaginary forces. The original formulation presented 4 main forces: a driving force towards the goal, a repulsive force given by the other pedestrians and obstacles, and finally a term modelling attractive terms, as displayed in Figure 2.6
This model has been so successful that throughout the years it has been continuously updated, finding new ways to design the forces leading to a more realistic behaviour. To give a more comprehensive view we will now go through some of the most impactful changes that have been made [12]. First of all, the different types of shapes that a pedestrian can be modelled with. In [38] [24], they designed the personal space of a person as an ellipse rather than a circle, with the idea that usually we tend to keep a greater distance in the direction of motion rather than on the other sides.
Another important aspect is group formation, which has been presented by Moussaïd et al. in [29]. Here they model a new term that can capture the forces that keep the group together such as wanting to keep eye contact with the other group members and not going too far from its center.

$$\frac{d\boldsymbol{v}_\alpha(t)}{dt} = \boldsymbol{f}_\alpha + \sum_\beta \boldsymbol{f}_{\alpha\beta} + \sum_b \boldsymbol{f}_{\alpha b} + \sum_\beta \boldsymbol{f}^{\text{att}}_{\alpha\beta} + \sum_i \boldsymbol{f}_{\alpha i} + \boldsymbol{\xi}$$

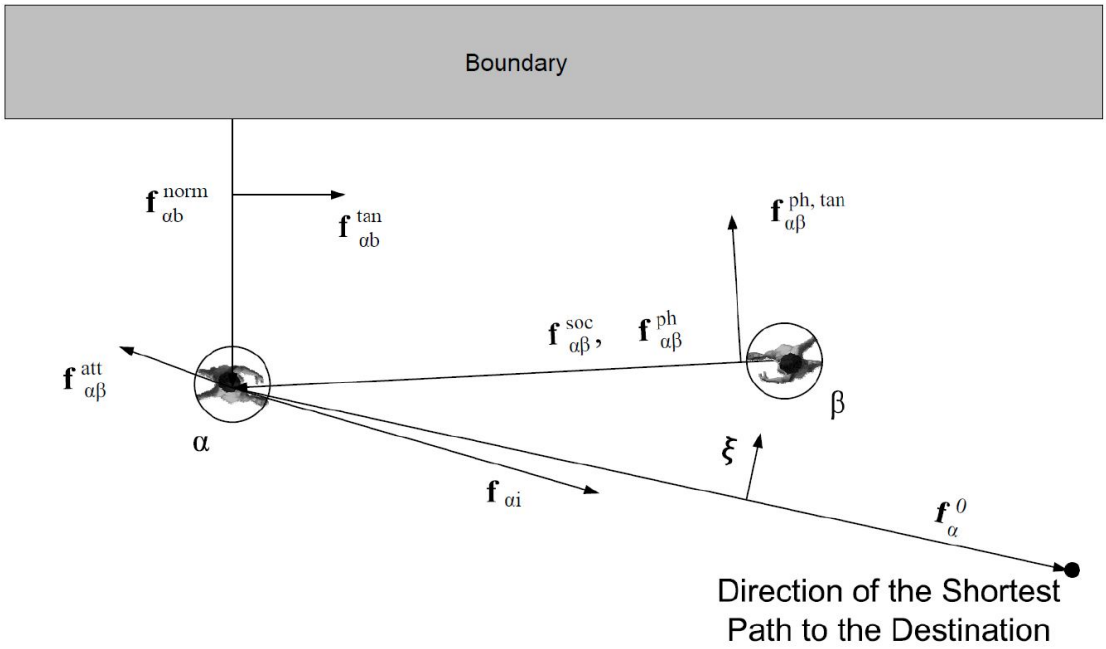


Figure 2.6: Elements composing the total social force for an agent [8].

For what concerns the movement itself, the first model could well handle the avoiding behaviour, and in the future model, the focus has shifted towards also adding other aspects such as overtaking [45] and self-stopping [30].
One last aspect to consider regarding the SFM is that it has a large number of parameters that represent different parts of the model and which need a very precise calibration to achieve the most realistic behaviour applied to the context of use.
Shifting now the focus to the application of the Social Force Model to e-scooters we can find two main contributions by Liu et al. [25] and Jafari et al. [23] that we are going to see in more details in the next section.

2.3 Behavioural models for e-scooters

Until now, we have seen the general ideas behind different behavioural models that are widely used in the literature. In this section, we are going to put the accent on these models applied directly to an e-scooter.
All the studies on this matter use an SFM to simulate the e-scooter interaction with other agents, while as of now no models have been proposed using the other techniques presented in the previous section. Among those, we can make two distinctions: those who use solely the forces and those who use also a decision process such as a state machine

on top of the force computation.
A clear example of the second category is given by He et al. [21], where the authors propose a simulation framework where they introduce an e-scooter mode to test mainly two scenarios: a lane change and crossing an intersection. To achieve the desired behaviour, they introduced a basic state transition model that comes into action whenever the scooter approaches a position to take such a decision. On the other hand, the social force model is used to regulate the lateral and longitudinal dynamics of the vehicle.
Under the category of the models using only the SFM, fall the majority of the ones produced. Here we also have different approaches to the problem, but we can distinguish two main contributions.
In [25], Liu et al. focus on the formulation of forces taking into account not only the effects of the other agents but also the geometry and kinematics of the e-scooter, defining a model capable of handling the non-holonomicity of the vehicle. This work is important because is the first one to design a specific formulation tailored to e-scooters and is called Scooter SFM (SSFM). Here they propose a different elliptical personal space geometry rather than the classic circular one. Furthermore, they account for the velocity difference that is present when dealing with different agents such as e-scooters and pedestrians. Lastly, they applied the kinematic bicycle model to the e-scooter in order to deal with the non-holonomic property. This model has shown good performances on the metrics analyzed. Despite this, a few limitations arose. The study focused only on the vehicle-pedestrian interaction and did not take into account different riding behaviours.
Following this work, another milestone was published by Jafari et al. [23]. This is an updated version of the previously discussed model trying to tackle the limitations of the previous one. In this work, they updated even more the personal space of the agents by introducing a velocity-dependent interacting space. It increases (or decreases) in size depending on the agent's speed. This led to a new formulation of the distance vector between the agents which in this new improved version is computed as the distance between the two closest points on the ellipse representing the interaction space. This is also the main drawback, since to compute the points it requires a minimization problem for each set of agents at each time step, which is very computationally expensive. The second contribution is regarding the calibration process. As mentioned, the calibration in the SFM is an important step and when having multiple agents of different kinds the runs required to calibrate all the parameters increase exponentially. This way they propose a weighted formulation using as weights a scale of importance depending on the agent type. The performances of this updated model have passed those of the SSFM on almost all the scenarios. The limitations of this work are the lack of different riding behaviours, the fact that is not designed to handle the interaction in a shared space with a car and finally also the high computational complexity that is required by this model.
On the other hand, there are models where, albeit not considering such aspects, they still obtain good results in testing, focusing on the exact calibration of the parameters of the model, in the scenarios they are studying [40].

2.4 Chosen approach

In the previous sections, we have presented some of the methods available to design e-scooters and represent them inside a simulator.
For what concerns the different platforms, all of them provide advantages and disadvantages, but the real need for a novel simulator that has been then designed and implemented by Prédhumeau et al. [33], is born mainly because all of the others lack when dealing about shared spaces and the interactions between AVs and pedestrians in such spaces.
For what concern this work, then the decision has been to continue the expansion of this simulator by adding the model of the e-scooter.
Regarding, the model used to implement the behaviour of such a vehicle, the final decision has been to use the SFM for two main reasons, the fact that using the SPACiSS simulator could give continuity given that the agents in this simulator move based on the force model and also being widely recognized as one of the main method when it comes to simulating behaviour of agents.

Chapter 3

System modelling: E-scooter, road lanes, sidewalks and social force formulation

In this chapter, we are going to present the mathematical formulation behind the work that has been done.
Firstly, we will see the e-scooter model that has been implemented, including the physical constraints and all the forces involved in its formulation.
Then in the second section, we will briefly discuss about the forces necessary to keep the agents in their respective lanes or within the boundaries of the sidewalk.

3.1 E-scooter model

In Section 2.3, we have seen the models available to represent an e-scooter. The models presented there, have very nice performances but have some limitations especially if we want to apply them to the goals fixed. That is the reason why, we have used them as inspiration to provide an even more tailored version of the e-scooter model that could accomplish all the objectives while defining a forward step in the field.
The main contributions that we will discuss in this section focus firstly on a revised version of the interaction space. It will become dynamic based not only on the agent velocity but also on the perceived density (Subsection 3.1.2). The second contribution is about the physical model applied to the e-scooter. We wanted to have a more complex and complete model that could account also for some of the other forces acting on the vehicle rather than the social force alone (Subsection 3.1.4). Last but not least, the design of two new forces representing the road lanes and the sidewalk will be discussed in section .
In this section we will see this, talking about all the main force components that form the final formulation we propose.

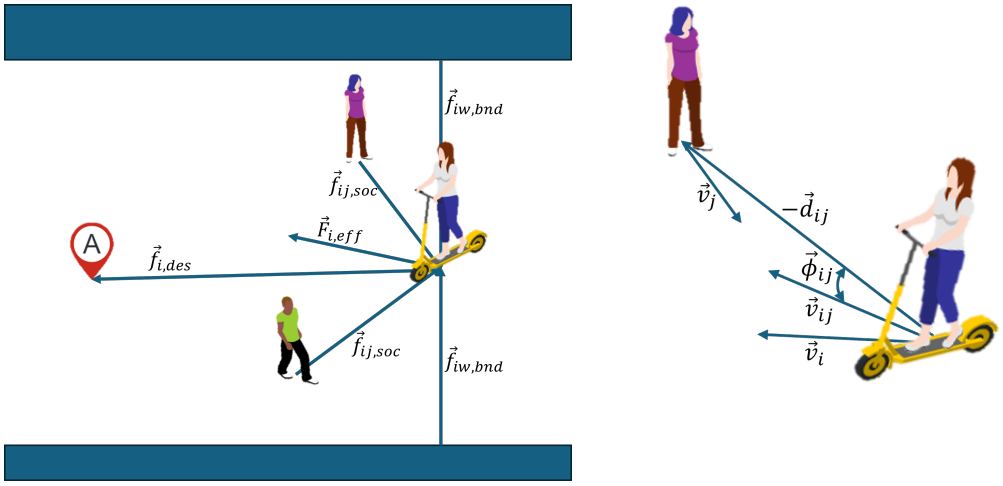


Figure 3.1: On the left is the definition of the SFM with all the forces involved acting on the agent i. On the right, the representation of the encounter angle is based on the relative velocity between the two agents.

3.1.1 SFM

As discussed in Chapter 2, the SFM is a useful technique when there is a need to represent continuously the behaviour of some agent.
The formulation utilised in this work is the following:

$$\vec{F}_{i,eff} = m_i \cdot \frac{\partial \vec{v}_i}{\partial t} = \vec{f}_{i,des} + \sum_{j \in N_i} \vec{f}_{ij,soc} + \sum_{w \in B_i} \vec{f}_{iw,bnd} + \vec{\xi}_i \tag{3.1}$$

Where $\frac{\partial \vec{v}_i}{\partial t}$ is the acceleration of the agent i with mass m_i and is equal to a summation of a term $\vec{\xi}_i$ which is responsible to deal with the fluctuations due to the randomness, plus 3 different forces: an attractive force $\vec{f}_{i,des}$, called desire force, which is the one that pushes the agent towards its goal; a repulsive force $\vec{f}_{ij,soc}$, called social force, which is the one exerted by the neighbours of the agent i on itself; and finally another repulsive force $\vec{f}_{iw,bnd}$, also called boundary force, that is the force acting on the agent i from the boundaries around it. N_i is the set of neighbours of the agent i and B_i is the set of boundaries perceived by the agent i.
The desired force, as mentioned before, is an attractive force and its formulation is the following

$$\vec{f}_{i,des} = m_i \cdot \frac{\vec{v}_{i,des} - \vec{v}_{i,act}}{\tau} \tag{3.2}$$

The formula is quite simple and compares the actual velocity of the $i-th$agent $\vec{v}_{i,act}$ with the desired one $\vec{v}_{i,des}$, which means the velocity it would like to keep in the case where it has no obstacles or other agents acting on it. Then we have a parameter, τ, which is the relaxation time and indicates the rate at which the agent would like to reach the

desired velocity. This is the parameter responsible for the aggressiveness of the agent in the simulation. While m_i is the mass of the agent i.
The boundary force, on the other hand, is a summation of all the forces that are acting on the agent from the boundary

$$\vec{f}_{iw,bnd} = \frac{\exp\left\{-\frac{\|\vec{d}_{iw}\|}{\epsilon}\right\}}{\sigma} \cdot -\vec{e}_{iw} \tag{3.3}$$

The use of an exponential in the formula is not random, but it is done to ensure that when the agent is getting closer to the obstacle the force increases, and the same happens if ϵ, the parameters capturing the obstacle force range, has a higher importance which means that the force weight in the exponential is higher. σ is also another parameter used to scale the force magnitude accordingly while, $\vec{e}_{iw}$ is the unit vector of $\vec{d}_{iw}$, and the minus in front is to ensure that the force is repulsive.
Finally, we have the social force itself

$$\vec{f}_{ij,soc} = \gamma(\phi_{ij}) \cdot \vec{g}_{ij}(\vec{r}_{ij}) \tag{3.4}$$

It is composed of only two factors: $\gamma(\phi)$ is the description of the anisotropic behaviour of the agent, which refers to the ability of an agent to act differently in different directions. ϕ is the encounter angle which is the angle formed between the distance vector $\vec{d}_{ij}$ and the relative velocity vector $\vec{v}_{ij}$ (Figure 3.1). $\vec{g}_{ij}$ is responsible for the concept of personal space of an agent with respect to the position of its neighbours. It is a function of $\vec{r}_{ij}$, which is the newly defined distance vector, that represent the distance between the centre of the agent j, and the agent i, as in (3.13).

$$\gamma(\phi_{ij}) = \lambda_i + (1 - \lambda_i) \cdot \frac{1 + \cos\phi_{ij}}{2} \tag{3.5}$$

$$\cos\phi_{ij} = \frac{\vec{v}_{ij}}{\|\vec{v}_{ij}\|} \cdot \frac{-\vec{d}_{ij}}{\left\|\vec{d}_{ij}\right\|} \tag{3.6}$$

In the first formula, λ is the parameter responsible for determining the isotropicity of the agent. Its value is bounded $0 \leq \lambda \leq 1$, where 0 means a complete anisotropic behaviour and 1 completely isotropic.

$$\vec{g}_{ij}(\vec{r}_{ij}) = \alpha_i \cdot \exp\left\{-\frac{n_{ij}}{\kappa_i}\right\} \cdot \left(\frac{\|\vec{r}_{ij}\| + \left\|\vec{r}_{ij} - \vec{y}_{ij}\right\|}{2 \cdot n_{ij}}\right) \cdot \vec{e}_{ij} \tag{3.7}$$

$$\vec{y_{ij}} = (\vec{v}_j - \vec{v}_i) \cdot \Delta T \tag{3.8}$$

$$\vec{e}_{ij} = \frac{1}{2} \cdot \left(\frac{\vec{r}_{ij}}{\|\vec{r}_{ij}\|} + \frac{\vec{r}_{ij} - \vec{y}_{ij}}{\left\|\vec{r}_{ij} - \vec{y}_{ij}\right\|}\right) \tag{3.9}$$

$$n_{ij} = \frac{1}{2} \cdot \sqrt{(\|\vec{r}_{ij}\| + \left\|\vec{r}_{ij} - \vec{y}_{ij}\right\|)^2 - \left\|\vec{y}_{ij}\right\|^2} \tag{3.10}$$

Looking at the way $\vec{g}_{ij}$ is defined, we can see the appearance of various parameters responsible for different aspects as we saw in precedence when talking about the boundary force. In particular, ΔT is a discrete-time step; κ is the force range constant; and α is responsible for the force magnitude.
In the following Subsection 3.1.2, we will present the geometry of the model where we will explain how the vector $\vec{r}_{ij}$ is computed and what it represents.

3.1.2 Geometry of the interaction between the agents

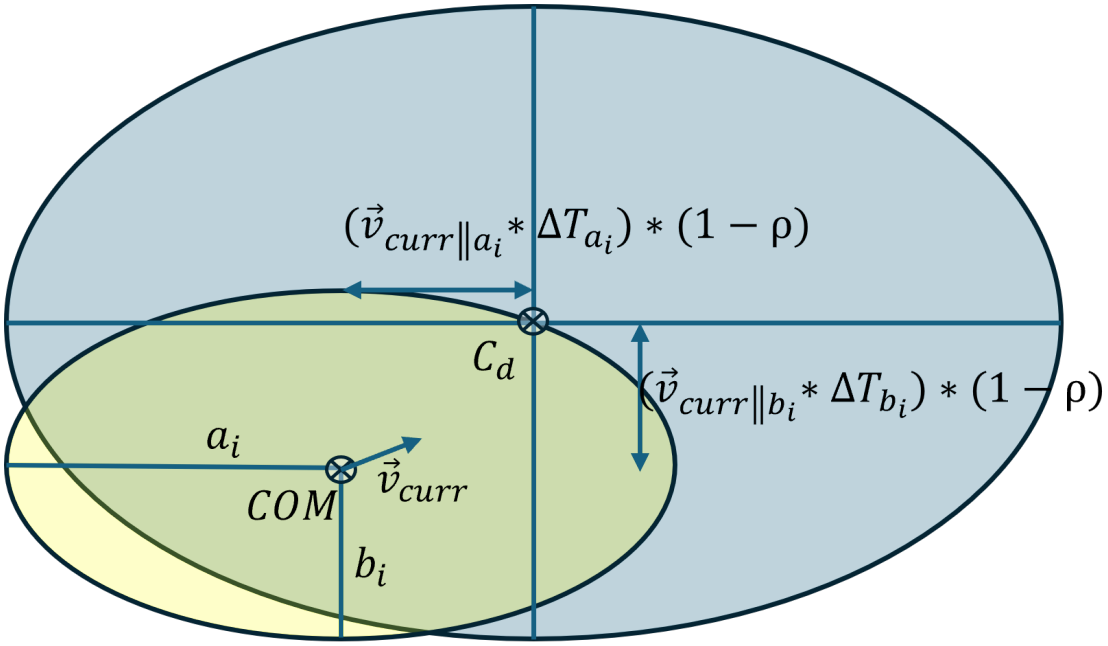


Figure 3.2: Representation of the two personal spaces of the agent. In yellow is the static personal space, while in blue is the dynamic interaction space. Is represented only in the case where $\rho \leq 1$. The other case is just the personal space(without the blue one).

The personal space of the agents until some years ago, has been always represented as a circle. Since then, many studies have been made on this topic and we now tend to think that the actual personal space of an agent is more of the shape of an ellipse [38]. In the model we have implemented we use this concept and we also make a distinction between the personal space and the interaction space. The latter is intended as a dynamic space that changes according to the density of the agents in the neighbourhood and the agent's speed and is the blue-coloured ellipse in Figure 3.2. On the other hand, the personal space is thought of, as a fixed element that varies from agent to agent but is constant in time, and usually, the agents never want to let others enter this space, because it would cause discomfort, and is the yellow, smaller space in Figure 3.2.
For what concerns the interaction space, as previously discussed, is dynamic so its value is dependent on the perceived density and its current velocity. The density value is based on the agent perception capabilities, and the area it can see is defined as follows:

$A_i = \frac{\alpha_i \cdot \pi \cdot r_i^2}{360}$, where α_i is the vision angle of the agent, in degrees. At the same time, r_i is its vision distance in meters. From here it derives that the density is $\rho = \frac{N}{A_i}$, where N is the number of neighbours of the agent.
We can so now define the updated values for the two axes which are:

$$a_d = \begin{cases} a_i + \left(\left\|\vec{v}_{curr\|_{a_i}}\right\| \cdot \Delta T_{a_i}\right) \cdot (1-\rho) & \text{, if } \rho \leq 1 \\ a_i & \text{, otherwise} \end{cases} \tag{3.11}$$

$$b_d = \begin{cases} b_i + \left(\left\|\vec{v}_{curr\|_{b_i}}\right\| \cdot \Delta T_{b_i}\right) \cdot (1-\rho) & \text{, if } \rho \leq 1 \\ b_i & \text{, otherwise} \end{cases} \tag{3.12}$$

The distinction between the two cases is necessary because in the case of a density greater than one, which implies that there is not sufficient space to navigate while keeping a gap from the others, the interaction space is null and the agent moves only trying to protect its personal space.

After defining the two spaces that each agent has, we can now focus on the distance

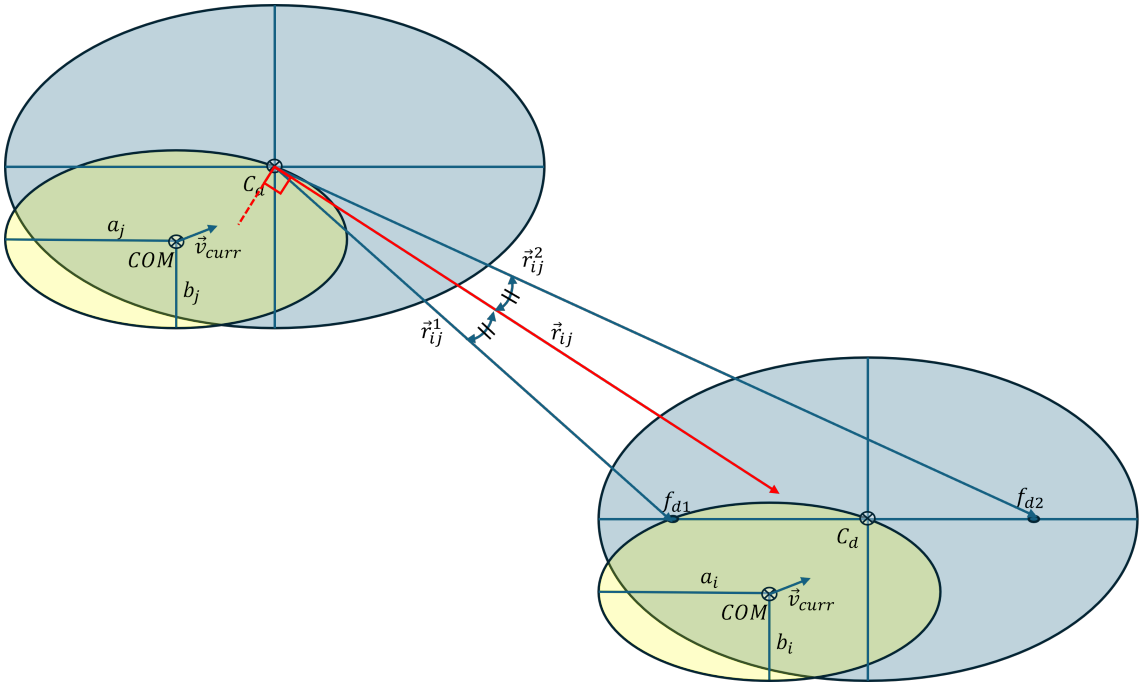


Figure 3.3: Graphical representation of the vector $\vec{r}_{ij}$

.

vector that rules the social interaction. This vector $\vec{r}_{ij}$ is the bisector of the two vectors $\vec{r}_{ij}^{\,1}$ and $\vec{r}_{ij}^{\,2}$ that connect the centre of the agent j to the foci of the larger ellipse of the agent i as displayed in Figure 3.3. The Mathematical definition of this vector is the one defined in (3.13).

$$\vec{r}_{ij} = \left(\frac{\left\|\vec{r}_{ij}^{\,1}\right\| + \left\|\vec{r}_{ij}^{\,2}\right\|}{2}\right) \cdot \left(\frac{\vec{r}_{ij}^{\,1}}{2 \cdot \left\|\vec{r}_{ij}^{\,1}\right\|} + \frac{\vec{r}_{ij}^{\,2}}{2 \cdot \left\|\vec{r}_{ij}^{\,2}\right\|}\right) \tag{3.13}$$

3.1.3 Velocity dependency

In the standard models, the distance between two agents is computed based on their current position, without taking into account their respective velocities. This can be somehow accurate in the case of two agents that are moving with a similar velocity, but when the gap between the two increases is no longer negligible. This happens because when two agents move at two different speeds the moment when we compute the distance, does not fully capture their relative position. Considering that in the simulator we are building we have different agents with different speeds, this is something that needs a deeper understanding.

As Jafari et al. presented in [23], a way to consider this is by projecting the original

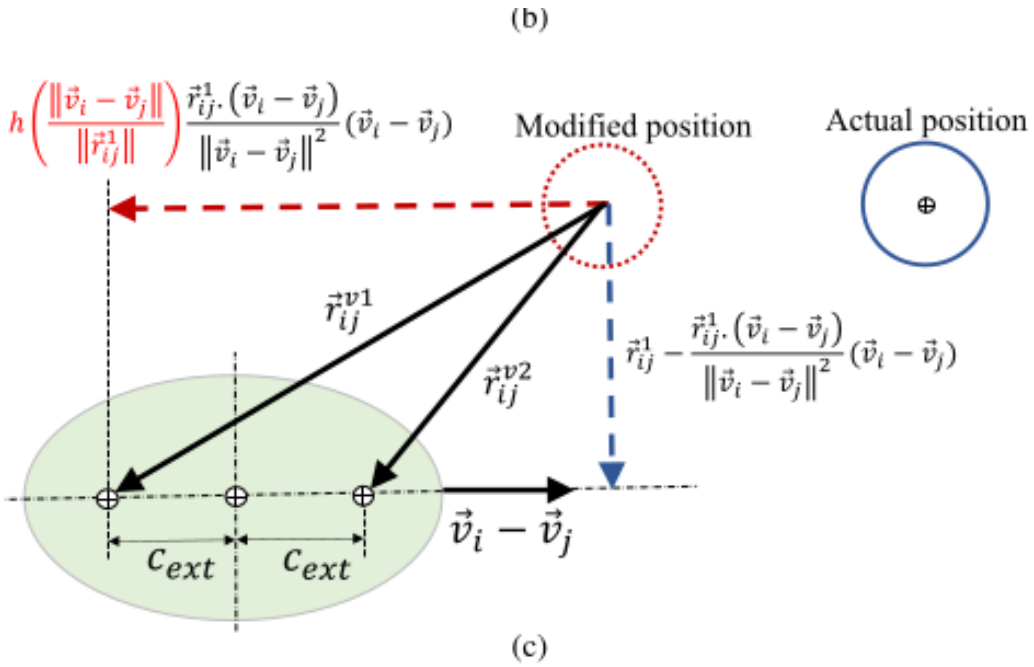


Figure 3.4: The projected vectors after applying the transformation to capture the difference in speed between the agents as presented in [25].

position vectors ($\vec{r}^1_{ij}$ and $\vec{r}^2_{ij}$), into the tangent and normal directions of the moving direction, $\vec{v}_i - \vec{v}_j$, as in Figure 3.4.

A scaling function $h(\cdot)$ is applied to the direction of the relative velocity vector, while the normal one is left unaltered. As for the scaling function, it has to satisfy a set of conditions:

$$h(0) = 1, h(\infty) = 0, 0 \leq h(x) \leq 1, \forall x \geq 0 \tag{3.14}$$

Typically, for this function, an exponential function to regulate the position vector is used, and the one we adopted is:

$$h(x) = \exp\left\{\frac{-|x|}{\mu}\right\} \tag{3.15}$$

μ is the velocity dependency parameter and needs calibration to get a proper value. Given this, the two velocity-dependent vectors that we are going to use to compute $\vec{r}_{ij}$ in (3.13) are:

$$\vec{r}_{ij}^{vk} = h\left(\frac{\|\vec{v}_i - \vec{v}_j\|}{\left\|\vec{r}_{ij}^{k}\right\|}\right)\cdot\frac{\vec{r}_{ij}^{k}\cdot(\vec{v}_i-\vec{v}_j)}{\|\vec{v}_i-\vec{v}_j\|^2}\cdot(\vec{v}_i-\vec{v}_j)+\vec{r}_{ij}^{k}-\frac{\vec{r}_{ij}^{k}\cdot(\vec{v}_i-\vec{v}_j)}{\|\vec{v}_i-\vec{v}_j\|^2}\cdot(\vec{v}_i-\vec{v}_j), k=1{,}2 \quad (3.16)$$

3.1.4 Kinematics and dynamics

The classic application of the SFM is to model pedestrian interaction, so after computing the total social force, $\vec{F}_{i,eff}$ for an agent, its integration gives us the velocity and the position of the agent at a determined moment.
When it comes to applying this, to a vehicle such as an e-scooter, we have to take into account the fact that it cannot translate sideways and its movements are limited by other physical forces that act on it. To get a realistic behaviour, not only concerning the rider's intention but also the e-scooter physical limitation we have to consider its non-holonomicity.
To do so, we will briefly introduce the kinematic bicycle model by Rajamani et al. [34]. This is the simplest way to represent a more complex vehicle such as a car, while still being accurate enough at low speeds.
The assumptions made in the formulation of this model are that the two wheels of both

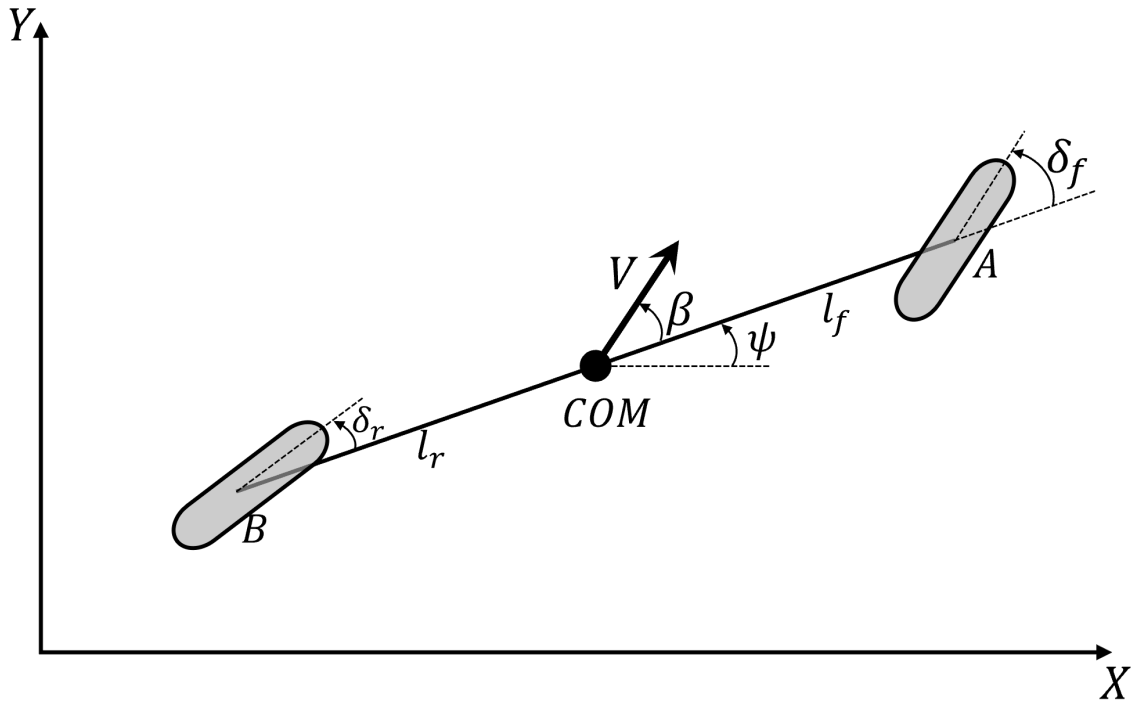


Figure 3.5: The kinematic bicycle model as represented by Rajamani et al. [34]

axles are collapsed into a single one (from here bicycle model) as presented in Figure

3.5, which implies that the two steering angles for the wheels on the same axle are the same; the sideslip angle is 0, thus the velocity vectors of the front and rear wheels are in the same direction as the corresponding wheels; the only inputs to the system are the steering angles (δ_f and δ_r) and the net velocity V. The formulas regulating this model are the following:

$$\dot{X}(t) = V \cdot \cos(\beta + \psi) \tag{3.17}$$

$$\dot{Y}(t) = V \cdot \sin(\beta + \psi) \tag{3.18}$$

$$\dot{\psi}(t) = \frac{V \cdot \cos\beta}{l_f + l_r} \cdot (\tan(\delta_f) - \tan(\delta_r)) \tag{3.19}$$

$$\beta = \tan^{-1}\left(\frac{l_f \cdot \tan(\delta_r) + l_r \cdot \tan(\delta_f)}{l_f + l_r}\right) \tag{3.20}$$

As mentioned before, this model is accurate enough, when considering low speeds or when we want to abstract to a higher level not considering the forces and torques that are acting on the vehicle, but when we are not in this situation we have to define a dynamic model. If we consider a 2 Degrees of Freedom (DoF) model, represented by its lateral position y and its yaw angle ψ. The lateral position is measured along the lateral axis of the vehicle that connects to the centre of rotation O. The vehicle yaw angle is the angle formed with the global X axis. V_x is the vehicle's longitudinal speed.

The lateral dynamics model that we will use is the one represented in Figure 3.6. This

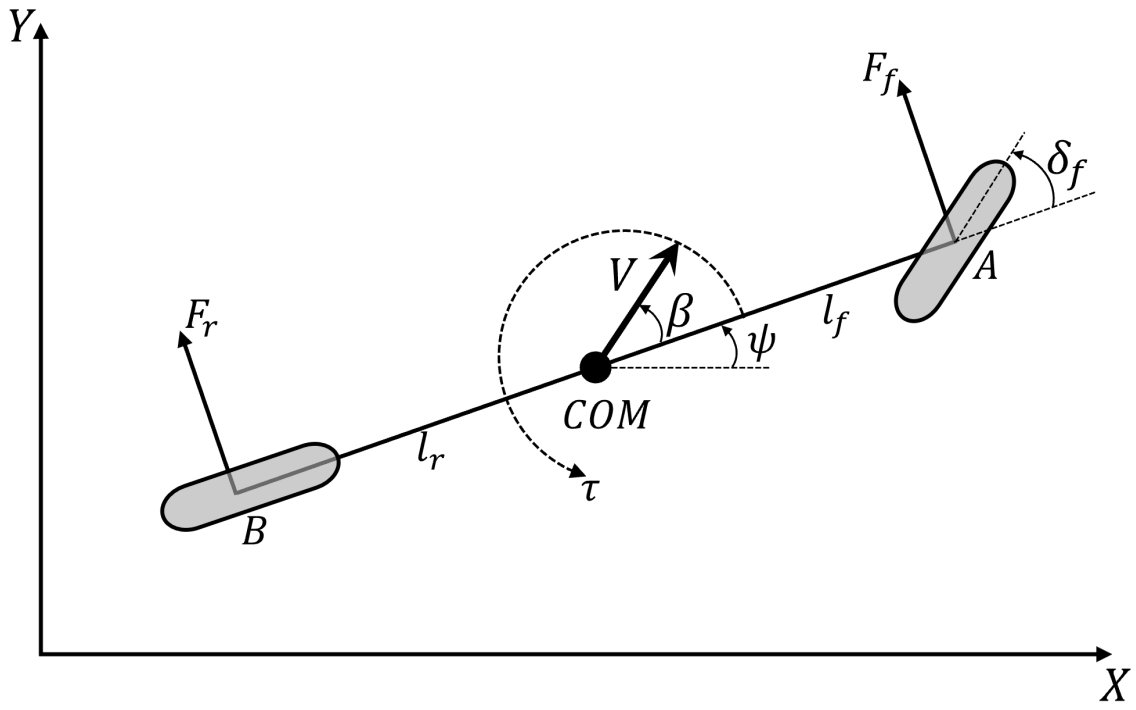


Figure 3.6: The dynamic bicycle model as represented by Rajamani et al. [34]

formulation can be described in terms of the vehicle sideslip angle β, defined as the angle

between the longitudinal axis of the vehicle and the orientation of the vehicle velocity vector, and r is the yaw rate of the vehicle body. The lateral dynamics of the vehicle are controlled by the front wheel steering angle δ. In the formula below, r represents the yaw rate and is equal to $\dot{\psi}$ ($r \equiv \dot{\psi}$), while I_z is the yaw moment of inertia.

$$\frac{\mathrm{d}\beta}{\mathrm{d}t} = -r + \frac{C_{\alpha f}}{m \cdot V_x} \cdot \left(\delta - \beta - \frac{l_f \cdot r}{V_x}\right) + \frac{l_r \cdot C_{\alpha r}}{m \cdot V_x} \cdot \left(-\beta + \frac{l_r \cdot r}{V_x}\right) + \frac{g \cdot \sin\phi}{V_x} \tag{3.21}$$

$$\frac{\mathrm{d}r}{\mathrm{d}t} = \frac{l_f \cdot C_{\alpha f}}{I_z} \cdot \left(\delta - \beta - \frac{l_f \cdot r}{V_x}\right) - \frac{l_r \cdot C_{\alpha r}}{I_z} \cdot \left(-\beta + \frac{l_r \cdot r}{V_x}\right) \tag{3.22}$$

The two models above are used when dealing with the lateral dynamics of the vehicle but when it comes to the longitudinal motion the main equation is regulated by the longitudinal forces acting on the vehicle [34] and are their point of application to the vehicle is presented in Figure 3.7. Those forces include the aerodynamic force F_{aero}, the force due to the rolling resistance of the tyres R_x, the longitudinal tyre force F_x and the force due to the gravity $F_{g\|x}$.

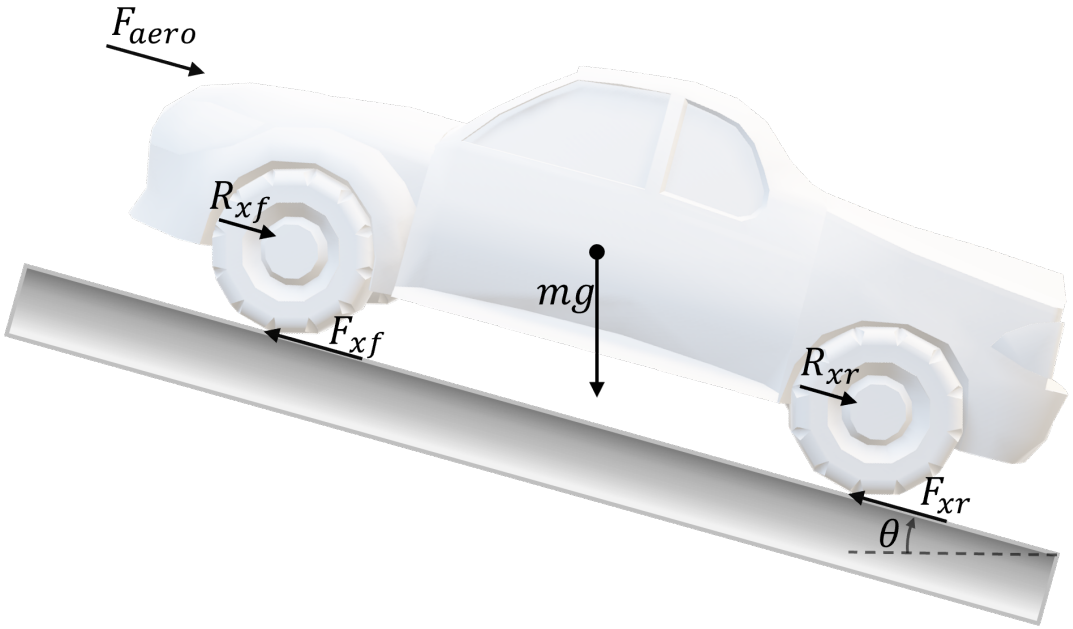


Figure 3.7: The longitudinal force model as represented by Rajamani et al. [34]

$$m \cdot \ddot{x} = F_{xf} + F_{xr} - F_{aero} - R_{xf} - R_{xr} - m \cdot g \cdot \sin\theta \tag{3.23}$$

What we have presented until now was a necessary preamble to correctly understand the model we have decided to choose when thinking about the e-scooter.
In our case, the only input that we have is the resultant force that we compute in (3.1)

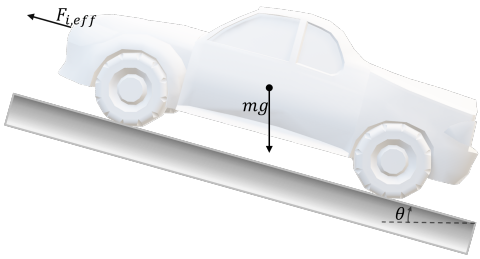


Figure 3.8: Longitudinal force model defined after the assumptions made

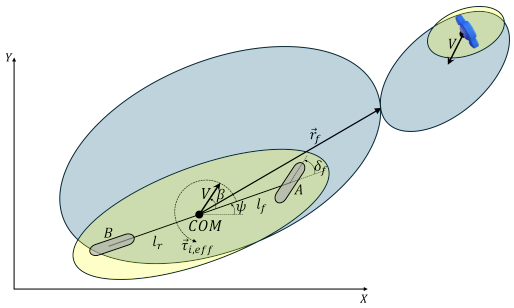


Figure 3.9: Dynamic force model defined after the assumptions made

and from this we have to derive all the other values.

For what concerns the longitudinal motion of the vehicle due to the nature of this work we have decided to neglect all the aerodynamic forces and rolling resistance forces, which translate to the only forces acting on the vehicle to make it move are the social force $\vec{F}_{i,eff}$ and the one due to gravity, as presented in Figure 3.8. We have made these assumptions since the main goal is to have a realistic behavioural model of an e-scooter rider, but despite being true that having a complex physical model could enhance the simulator, the idea is that it was not the first objective at this stage, and it can always be refined in the future. So the simplified longitudinal dynamics are:

$$m \cdot \ddot{x} = F_{i,eff} - m \cdot g \cdot \sin\theta \tag{3.24}$$

From (3.24) we can derive through integration the longitudinal speed: $V_x = \int \ddot{x}$.
Looking now at the lateral dynamics, since the effective force is not acting on the CoM it means that it generates a rotational torque, which is what makes the vehicle turn. As discussed before for the longitudinal dynamics, in this case as well we have made some simplifications to the model and we neglect all the other forces acting on the vehicle and consider the only force acting as the social force. Hence we can rewrite (3.22) as follows:

$$I_z \cdot \dot{r} = \vec{\tau}_{i,eff} = \vec{r_f} \times \vec{F}_{i,eff} \tag{3.25}$$

The vector $\vec{r_f}$ represents the rotational arm, the vector that connects the CoM to the point of application of the social force. This point is the one we can see in Figure 3.9 and represents the point of intersection between the two ellipses representing the interactive space. Thus, we now have a way to derive the yaw rate and so to compute the orientation of the vehicle ψ. The last value that we need to compute is the lateral speed, which following the assumptions we made of a negligible sideslip angle gives us the following set of equations.

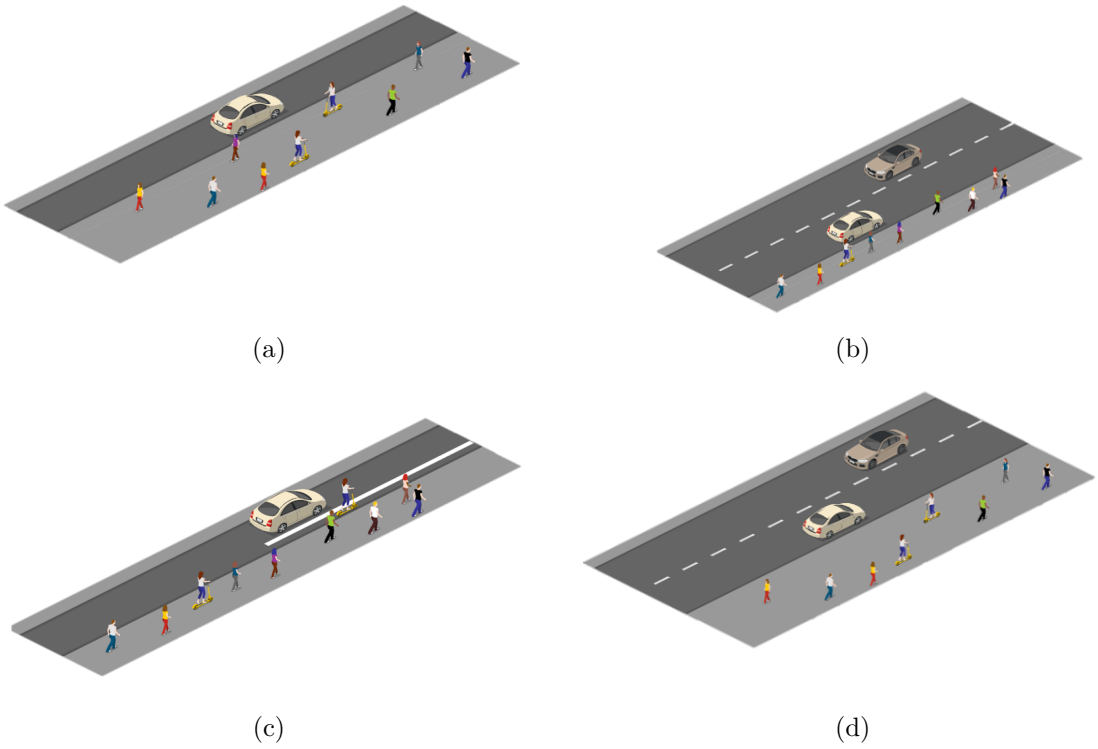
$$\dot{r} = \frac{\vec{r_f} \times \vec{F_{i,eff}}}{I_z} \tag{3.26}$$

$$V_y = V \cdot \sin\psi \tag{3.27}$$

The system composed by (3.24), (3.26) and (3.27) describes the equations that handle the physical model of the e-scooter.

3.2 Road lanes and sidewalk model

In this section, we explore how we have formulated the forces representing the road lanes and the sidewalk. There have been a few attempts to define forces acting to represent road elements such as the zebra crossing in [46], but we have not found extensive work done in the direction of road lanes and sidewalks in multiple configurations, that is what we will analyze here.



(a) (b)

(c) (d)

Figure 3.10: Examples of different road lanes and sidewalk configurations. a Single road lane, with a two-lane sidewalk, b Two-lane road, with a single-lane sidewalk, c Single road lane with lateral space for PMVs, with a single-lane sidewalk, d Two-lane road, with a two-way sidewalk.

3.2.1 Sidewalk

The sidewalks around the cities are of different kinds as we can see in Figure 3.10. Since the nature of this thesis is to explore the interactions between different agents in shared spaces we have decided to focus our work on the ones that are at least shared between faster vehicles, such as bicycles and e-scooters, and pedestrians going in the same direction. To correctly design a force capable of representing such elements, firstly is important to think about what is a sidewalk for its users. If we think about the sides of a sidewalk, they can be seen as a wall that typically users do not want to pass. However, a big difference is that, to avoid a collision with other users or to pass slower users when they

occupy the whole space on the sidewalk is possible to go outside the borders to then come back. This is valid for all types of sidewalks and must be taken into account. As for those with two lanes going in two different directions, they usually have a midline that agents do not want to pass as well, but they tend to pass it with more easiness than the outer lanes if necessary.
Given the nature of the lane delimiter that can be associated with that of a wall, we have decided to use as the main formula the same presented in the Subsection 3.1.1 for the boundary force.

$$F_{ext} = \beta_{ext} \cdot \exp\left\{ -\frac{\left\|\vec{d}_{i,el}\right\|}{\epsilon_{ext}} \right\} \cdot \vec{e}_{i,el} \tag{3.28}$$

$$F_{mid} = \beta_{mid} \cdot \exp\left\{ -\frac{\left\|\vec{d}_{i,ml}\right\|}{\epsilon_{mid}} \right\} \cdot \vec{e}_{i,ml} \tag{3.29}$$

What differentiates the two is that in this case, the weight will be different depending on which lane is representing and will assume a value such that, if there is the need the agents can go past the lane to then come back. To represent this, firstly each agent needs to know in which direction is going, to associate the external lanes to itself, which can be checked by performing a simple operation such as $atan2(V_y, V_x)$. Then each lane emits a force which in case of the absence of external forces makes the agents stay in the middle of the lane. This can be summarized by the following definition:

$$\begin{cases} \vec{f}_{ext} & \text{, if agent } a_i \text{ in lane} \\ -\vec{f}_{ext} & \text{, otherwise} \end{cases} \tag{3.30}$$

As we can see, when the agent goes outside the lane boundary, the force becomes attractive to bring back the agent.

3.2.2 Road Lane

Concerning the road lanes, the situation is less complex given how the simulator is built. On the car are not acting any other forces but the desired force to make it reach the goal and the boundary force to make it avoid hitting the walls, which means all it does is go straight to the goal. The outcome of this is that the car is not going to perform overtaking manoeuvres so the force that pulls back the car when going outside the boundaries of the lane and its logic are not necessary here. So to model this element we have used the same equation as in 3.3 and there is no difference between the external lane delimiter and the midline.

3.3 Final Formulation

After defining these new forces we can update the social force definition we presented before in (3.1), by adding the terms modelling the road lanes and the sidewalk. Therefore,

the force acting on an agent i at each moment can be presented as:

$$\vec{F}_{i,eff} = m_i \cdot \frac{\partial \vec{v}_i}{\partial t} = \vec{f}_{i,des} + \sum_{j \in N_i} \vec{f}_{ij,soc} + \sum_{w \in B_i} \vec{f}_{iw,bnd} + \sum_{l \in E_i} \vec{f}_{il,ext} + \vec{f}_{i,mid} + \vec{\xi}_i \quad (3.31)$$

Chapter 4

Implementation

This chapter will focus on how to translate the theoretical foundation we proposed in the previous chapter into a practical implementation. We will discover the tools used in the build of the simulator and the complete structure of the project, to then focus on the most interesting parts.

4.1 Tools

In the construction of this project, we have used different kinds of tools and software. In this section, we introduce those elements and try to understand their role in the project.

4.1.1 ROS

Robot Operating System (ROS), is an open-source robotic middleware suite [41] [3]. Here we will focus on explaining how ROS 1 Noetic works, since is the version used in the development of this project. ROS is not an operating system but a set of software, conceived for robot software development. Despite this, it still provides hardware abstractions, low-level control commands, message passing between processes and package management.
When you run a set of ROS-based processes they will be represented in a graph structure, where each process will be executed in a node, which may receive or publish sensor data, control, actuator and other messages. This graph structure can be seen in Figure 4.1.
ROS is not a real-time operating system, however, it is possible to integrate it with real-time code. When looking at the ROS ecosystem the software it contains can be divided into: ROS client libraries and packages containing application-related code which utilize the ROS client libraries.
What is fundamental to understand about this powerful middleware is its structure and its philosophy.
The processes are represented as nodes which are connected to each other by edges that are defined as topics. Each node can send messages to the others, provide a service or make service calls to other nodes and retrieve information from a common database called the parameter server. All this is possible thanks to a node called *Master node* which serves as the base node and registers all the others to itself while updating the parameter server.

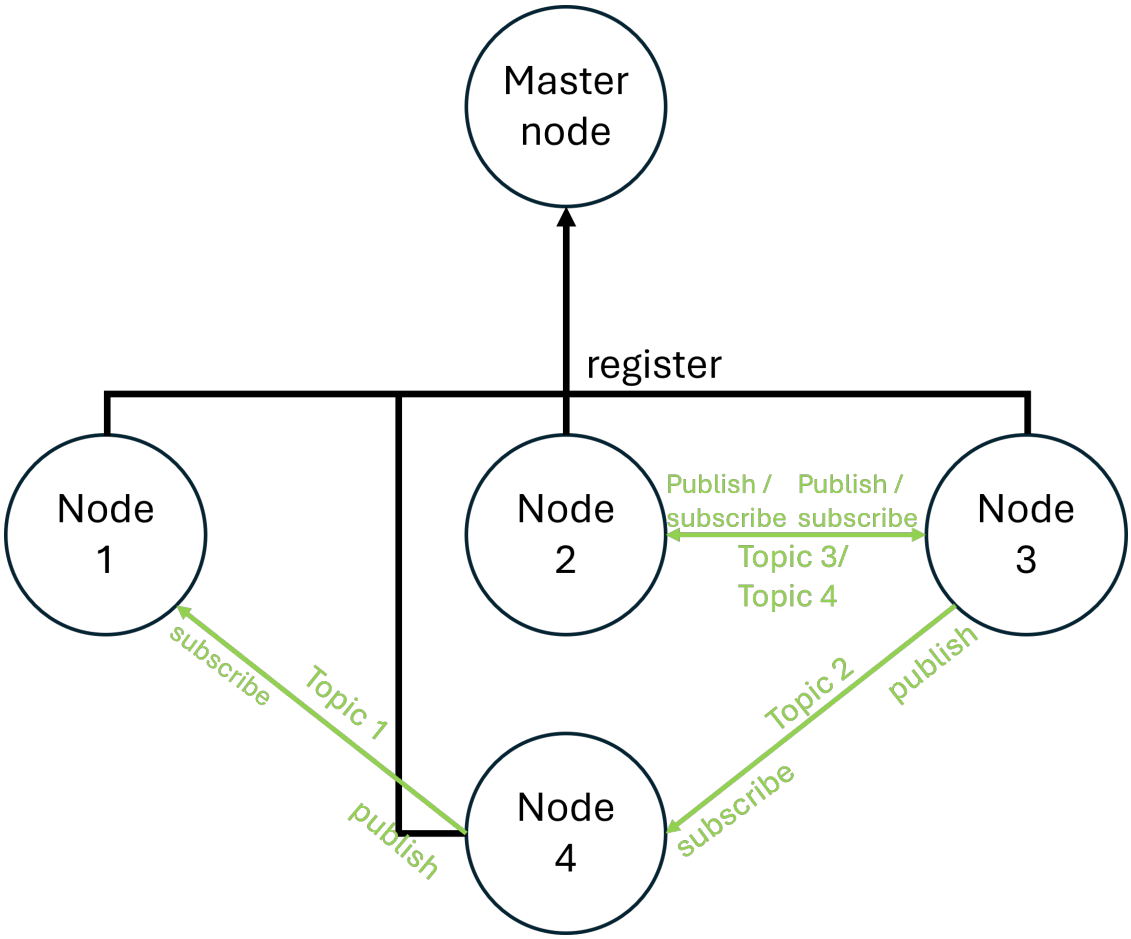


Figure 4.1: ROS structure highlighting the graph structure where each node must register to the master and can publish/subscribe to topics that represent the arcs.

The messages and service calls do not go through this node but it sets up peer-to-peer communication between the nodes decentralizing the architecture. This representation is well suited for a robot which is often represented as a union of different parts that may communicate with off-board computers to handle heavy-duty computations.
We will now go a bit more in detail on the most important concepts that we have mentioned until now such as nodes, topics, services, etc.

Nodes A node is a process running in the ROS graph structure. It registers to the master node with a name that must be unique within its namespace. Nodes with the same name can exist in different namespaces. This is the most important structure because most of the time the ROS client code a programmer writes is in the form of nodes that will do action based on the messages that they receive or send, or based on the services it asks or provides.

Topics A topic, often named also bus, is a structure over which a node can send or receive messages. It must have a unique name within its namespace as well. To send messages to a topic a node must publish to said topic, while to receive it has to subscribe. The publish/subscribe mechanism is anonymous because only the nodes that publish/-subscribe know the identity of the sender/receiver.

Services A service represents an action that a node can take which will produce a single result. This is suitable for those situations where we have a known start/end situation. Nodes can provide a service or request one from one another.

ROS parameter server The ROS parameter server is a database shared between the nodes, which contains static information. It usually contains parameters that do not change at all or do not change quite often and which will not be accessed frequently. An example could be a fixed URL to some file or the weight of the robot.

Packages Another important concept are the packages. These are the ways ROS represents the open-source implementation of algorithms or common robotics functionalities. They can be either provided as part of the ROS distribution or can be built by users to contain specific implementations.

rviz Among the multitude of tools that ROS provides, one of our interest is **rviz**.
This is a tool used to visualize robots, the environment they work in and their sensors in a 3-dimensional space, as in Figure 4.2. It is a highly configurable tool, which in our case has been used to represent the different scenarios and all the agents involved in them. The different robots are defined by a URDF file which represents the robot model and is interpreted and loaded by rviz. It is important to mention here another tool that makes possible all the transformations between the different frames of the robot parts and that of rviz which is a package called the **TransForm system** (tf) [19]. This package is important in a system such as a robot, which in our case is represented by the e-scooters and the car because each part of it has its own orientation and reference frame. To handle all the transformations that are necessary between the different parts and to represent the robot in rviz, using its base reference system, we would need to keep track of all this information, while the tf package can handle all this for us, given that all the namespaces are well set as well as the links in the model. It publishes the transformed coordinates in the requested reference frame.

4.1.2 SPACiSS

The next tool we are presenting is SPACiSS which is the platform we used as a base for our work. It is an open-source simulator built by Prédhumeau et al. [33] to have software to be used to test navigation algorithms for autonomous cars.
It is built using ROS as a middleware to handle the different areas involved in the simulation and it uses *pedsim_ros* [2] as a base to handle the pedestrians in the simulation,

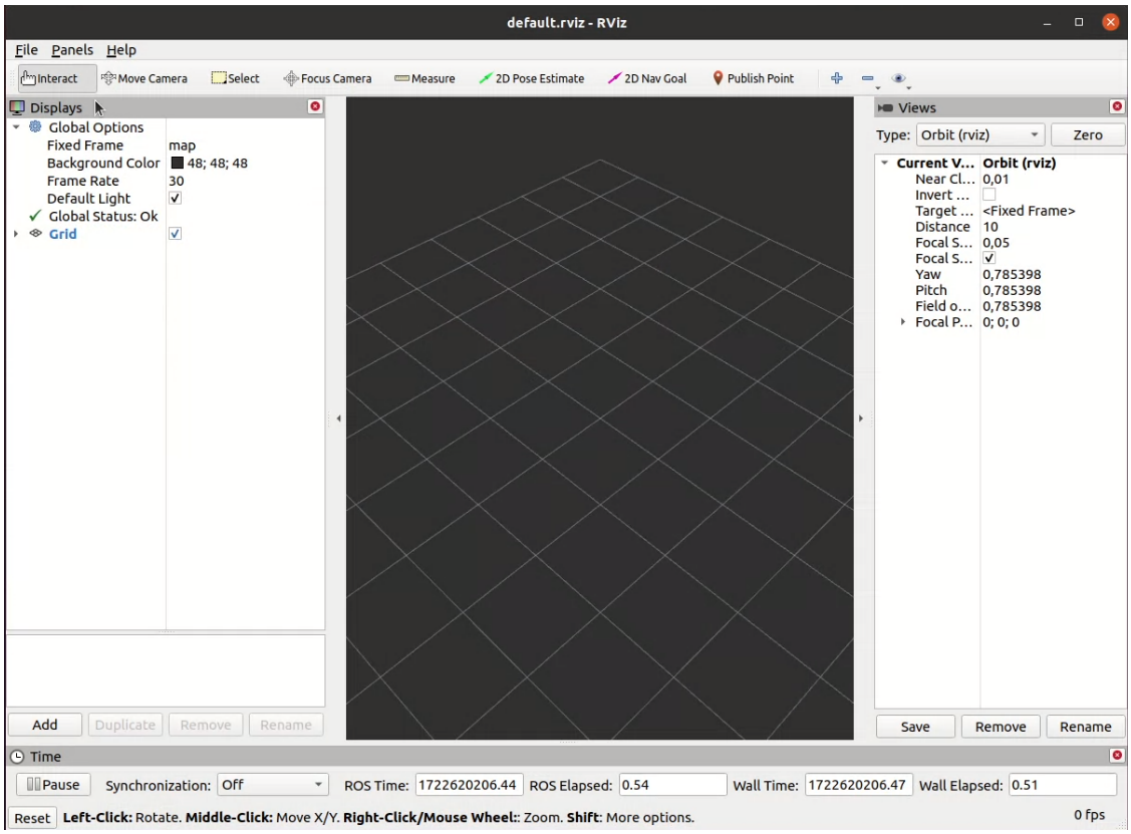


Figure 4.2: Rviz empty scene. On the left what is the topics or models to display and on the right the camera settings.

as shown in Figure 4.3. *pedsim_ros* is a package that simulates a crowd of pedestrians implemented using the SFM. It can simulate a large number of different situations like a group of people moving together.
With SPACiSS the goal was to enrich the different cases that pedsim alone could handle, and in the final version, there have been a lot of new adds, as we have discussed in Chapter 2.

There are two packages that are important to understand and discuss also for what we have implemented in this work.

Pedsim_simulator This, is the package liable for producing the simulation results. Its objective is to spawn all those nodes that are responsible for publishing the positions of all the agents and objects that are on the scene at each time step, after computing the updated data. Here happens all the process of perception, where each agent updates the perceived neighbours and obstacles; the decision where based on the perception process, the social force acting on each agent is computed; and finally the action part, where thanks to values computed in the previous step the agents can update accordingly their state. Finally, all these data are published on a series of topics that can be distinguished

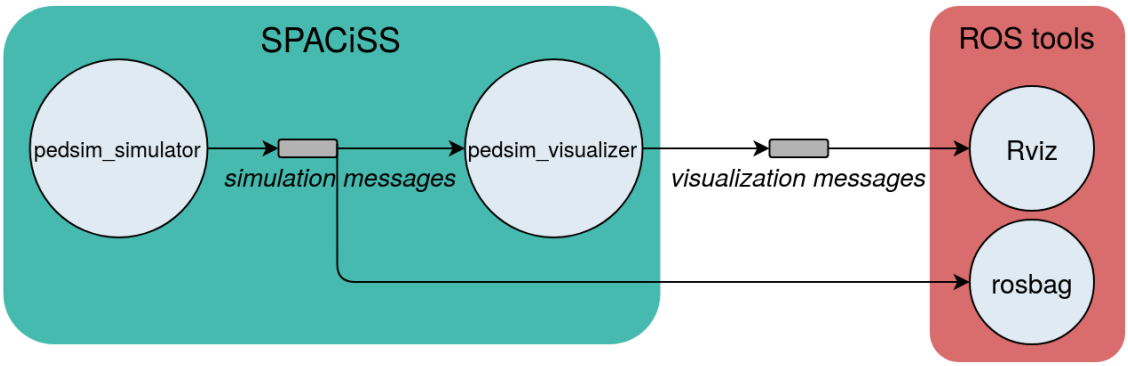


Figure 4.3: ROS nodes executed during a simulation with SPACiSS.

by the *simulation* tag.

Pedsim_visualizer Its goal is to subscribe to the simulation topics and to publish to rviz the topics so that can be displayed. The process here is much simpler, and what it does is just to get the positions provided by the simulation and put those in a form readable from the visualization software, which means defining appropriate messages.

Experimental_package It is the container for all the defined scenarios from which we can choose when running a simulation. Other than that it contains the URDF models for the car, which in this case is a Renault Zoe, which is the car the laboratory possesses and is used to run real-world experiments.
After defining briefly the most important parts of this simulator we can focus on the way it works.
There have been defined scenarios that are simulateable, including the city centre. There are two operative modes, one where the car that is shown is not controlled and the only force acting on it is the desired and the obstacle one, which is useful to test how the pedestrians react to an AV moving among them. The second mode is where you can control the car movements either with an algorithm or with terminal inputs. This is useful after having tested the pedestrian behaviour to use the simulator for what is built which means testing navigation algorithms for AVs in shared spaces.

4.2 Implementation

In this section, we will finally focus on our implementation of the model we have presented in Chapter 3. Firstly we are going to see the updated global view of the simulator to then delve into the parts that have been either modified or added.

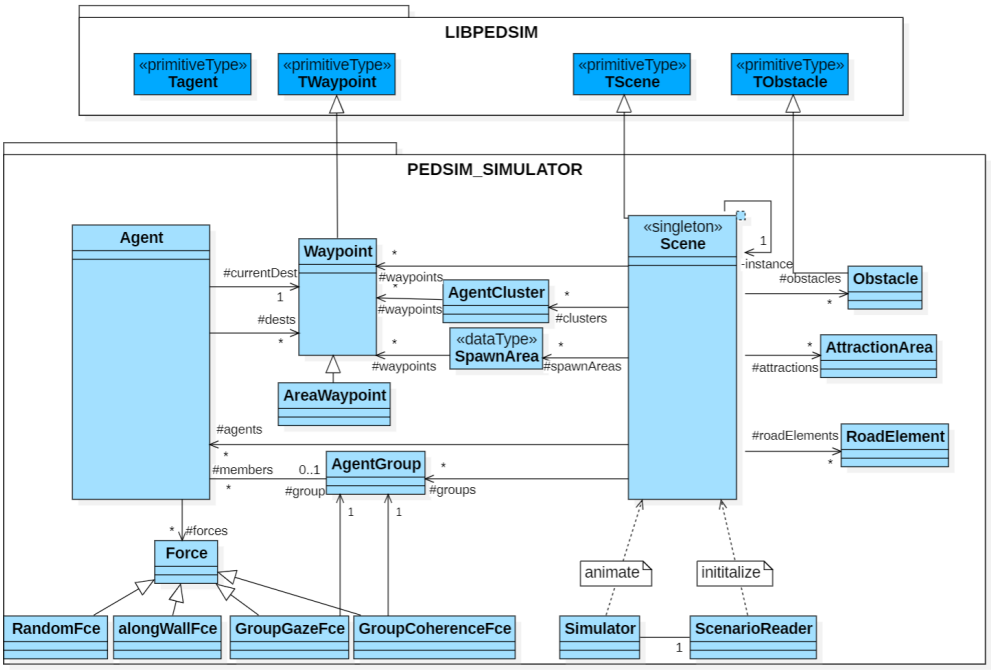


Figure 4.4: UML diagram of the package `pedsim_simulator` representing the dependencies in SPACiSS.

4.2.1 General view

As we can see from Figure 4.5, the structure stays unaltered, with most of the modifications that happened inside every single package, which means no additional package has been created. In the implementation of this model, one additional goal was to maintain, as much as possible, the global structure of the simulator the same, so that it would result in an addition to the existing version and not as something foreign to the simulator. This has been accomplished by copying the structure of the definition that was previously defined by existing classes or updating the definition of a previously created one rather than creating a new one that could be hardly understandable afterwards.

4.2.2 Pedsim_simulator

As discussed before, this is the core package of the project and here there have been the most changes. The base process has remained the same, so each agent goes through a perceive, decide, act course of actions. The definition of the agents has been updated to include a new type which is the e-scooter. This new type has a few characteristics in common with both the previously defined agents because it necessitates additional computations, due to its physical constraints, but at the same time, it can be assimilated to a pedestrian because on a scale of vulnerability compared to a car, is closer to the latter rather than the former.
The fundamental change has been the way the social force is computed for this new agent.

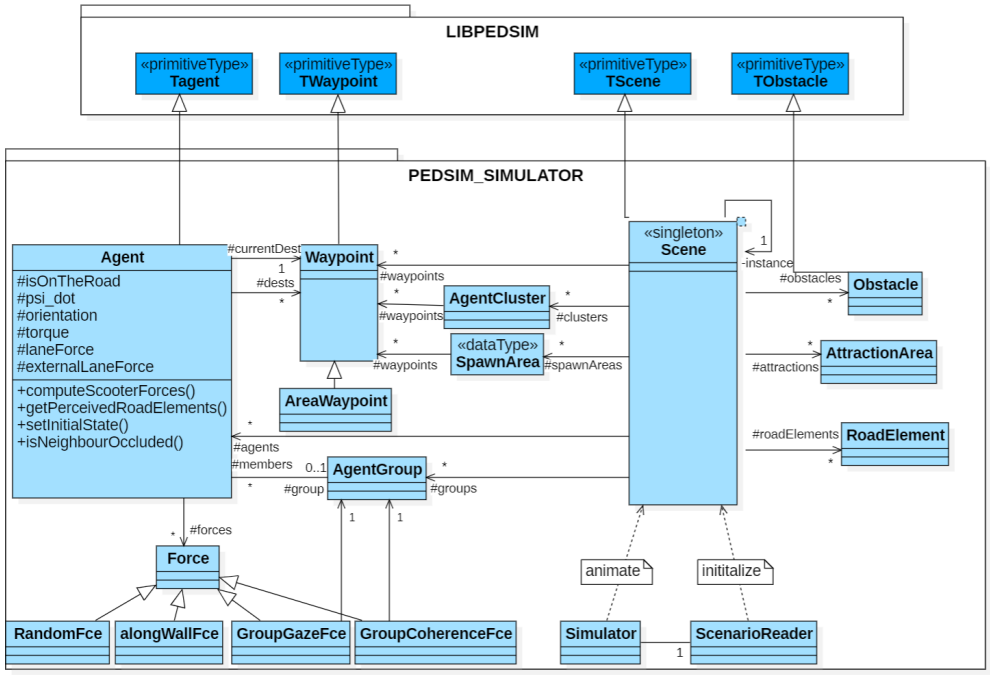


Figure 4.5: UML diagram of the package `pedsim_simulator`, with modifications made to this core package.

As we have seen in Chapter 3, there are different steps to be followed and all have been widely discussed there. What is interesting to see is that the e-scooter can perceive if in front has a group of pedestrians or a single one. In the first case, it does not go through the middle of the group, even if there could be an opening wide enough to let the vehicle pass without making contact with anyone, but rather goes around it on the side closer to its current position.

Another aspect to mention is how the lane and sidewalk forces are computed. This is important because, in the definition of these new classes, we tried to leave the final user with the least possible verbosity possible, when defining a scenario. This has a flip side, which means a slightly higher complexity when processing that information. When defining a lane (or a sidewalk) the user can decide among a set of predefined values, which represent if the road we want to represent is two lanes or only a single one and we specify only its position. At the same time when spawning an agent, we just define its position on the scene. This means that the e-scooter does not have the information of whether it is on the road or not since the beginning. We will check this by comparing the angle formed between its direction vector and the direction vector of the lane closer to it. This way we can update the state of the agent by setting in which lane is going and if it is on the road or not. This is important because when on the road the e-scooter will update its vision parameters increasing the perceived angle, since while on the road the riders are more focused also to what is happening behind them to avoid incidents.

Finally, the last topic to present from this package is how the kinematic constraints are

placed in action. After computing the updated social force there is a module that uses the information published by the simulation process, to compute the updated position and orientation to then feed them to a transform that will convert them in a fixed frame to be correctly visualized in rviz.

4.2.3 Pedsim_visualizer

In this package, there have been only a few modifications necessary to handle the newly defined road elements. The most important thing is that depending on the road element defined this package is responsible for processing it and displaying correctly the road, which means if there is a two-lane road it will display the discontinuous lane in the middle of it or not.

4.2.4 Experimental_package

The last package we will analyze is where the scenarios are defined. Here we have defined a new model that is necessary to display in rviz the e-scooter. We have used a Segway Ninebot Max G30 as a reference to build the URDF model.
We have defined three new scenarios which we have used to conduct the test we will present in Chapter 5.

The last important modification implemented is the addition of a dynamic launcher for the e-scooters. This was necessary since, to spawn multiple vehicles that are the same, as we have discussed in the previous section each one of them must have a different name. To do so, there are two ways, either to create a predefined number of vehicles that are spawnable and create each file manually, or give the possibility to the user to define a custom scenario with an arbitrary number of e-scooters that will be handled automatically by reading the launch file and assigning to each one of them a different namespace, which is why in the Figure 4.6 we have a much more complex tf graph structure.

Parameter	Original Value SPACiSS	New Value
Wall	Start position	-
	End position	-
Attraction zone	Position	-
	Dimension	-
	Attractive force	-
Destination	Position	-
	Dimension	-
	Behaviour (simple, source, sink)	-
Agents	Number of agents	-
	Distribution around the position	-
	Type of agent (pedestrian or AV)	Pedestrian, AV, or e-scooter
	Goal (work or pleasure)	-
	Destination	-
Spawn Area	Position	-
	Number of agents	-
	Distribution around the position	-
	Goal (work or pleasure)	-
	Destination	-
Road Element	-	Start position
	-	End position
	-	Type (Single road lane, two-lane road, single-lane sidewalk, two-lane sidewalk, single road lane with legal PMV riding)

Table 4.1: Comparison between the available parameters in a scenario between SPACiSS and our updated version. No changes are indicated as "-".

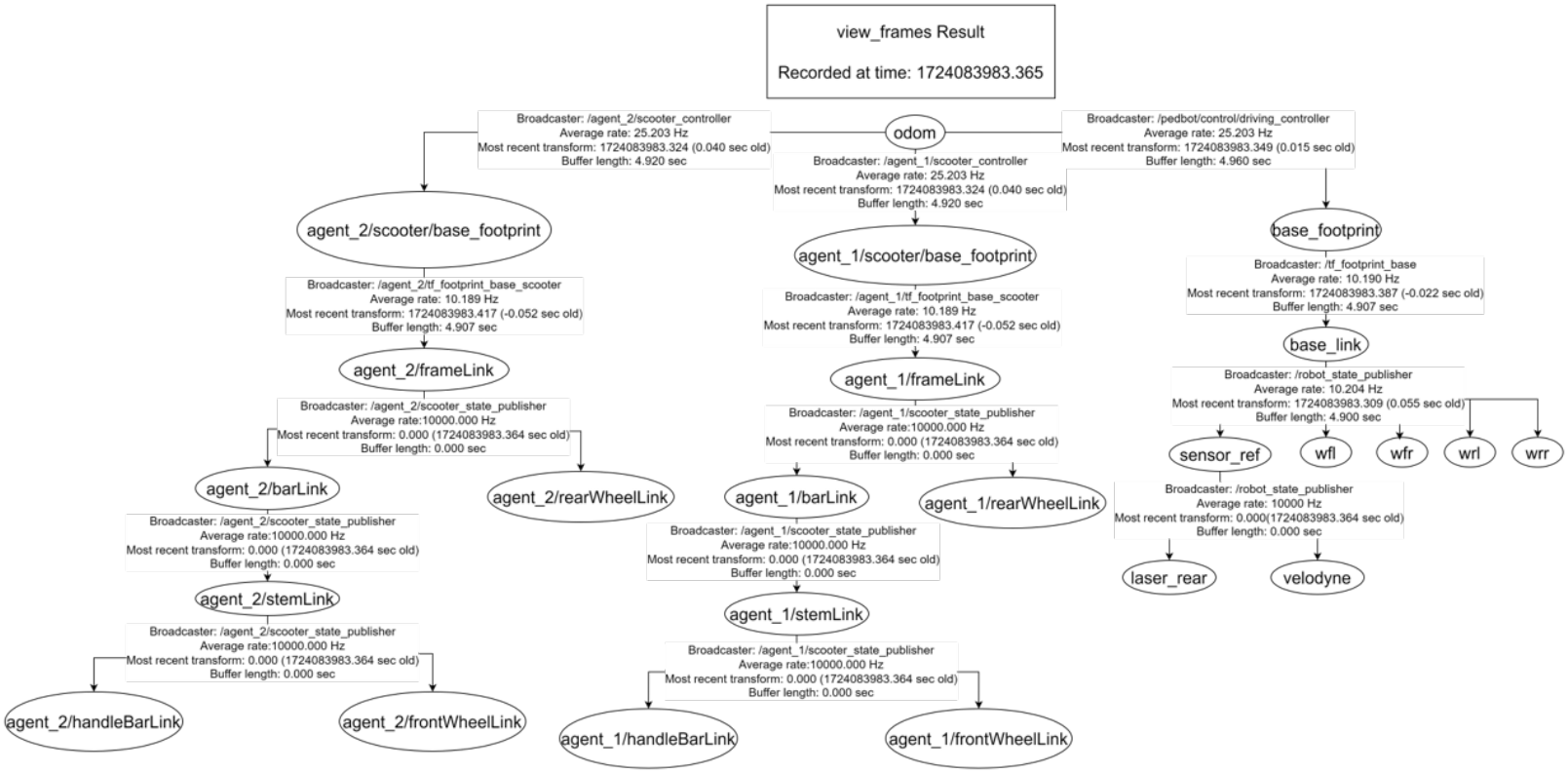


Figure 4.6: tf tree representing the transformation necessary to handle two e-scooters and a car correctly at the same time.

Chapter 5

Evaluation

This chapter covers the methodology used to conduct the experiment that we have decided to do on the model we have implemented. Since the SFM is a model which is strongly based on an accurate tuning of all the parameters involved in the formulas that we have seen in the Chapter 3, firstly we will present the metrics used to do so. Then it will be followed by the first experiments we conducted which are based solely on the simulation, and is a crash analysis to verify that in the eventuality of a crash, it would be a realistic situation and not something that in reality would be impossible. Finally, we will introduce the validation process that is conducted on real data.
One of the objectives of the evaluation phase was to build a personal dataset to test all that has been described until now, since in this field, at the moment, there is a gap in datasets including e-scooters in different scenarios. Unfortunately, this could not be possible, since the engineers who were responsible for the collection of the data in the various scenarios we proposed, could not manage to produce the dataset in time for us to conduct all the tests. Nonetheless, we still set up everything for the moment the dataset is going to be available and in the next sections we are going to present you the metrics that we want to use to perform the above-mentioned tests.

5.1 Calibration

The calibration process is a fundamental part of obtaining realistic behaviour of the implemented SFM model. As we have explored, there are a lot of parameters to tune (as can be seen in Table 5.1), and sometimes their value is also dependent on the scenario we are considering and the behaviour we want to simulate.
In the literature there are different ways to address this problem, both in the algorithms used for the optimization process and the definition of the function to be minimized. Some of them use techniques inspired by natural evolution (Evolutionary algorithms) like Yang et al. [43] who use a genetic algorithm to calibrate the parameters while Cui et al. [13] use a differential evolutionary algorithm; then there are also more mathematical approaches such as Bassoli et al. [9] who use a stochastic algorithm like a response surface methodology and Dias et al. [15] who use a cross-entropy method.

Figure 5.1: Scenario 1 representing a city centre

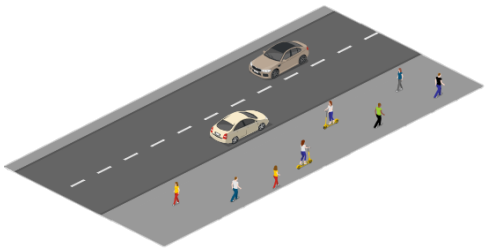


Figure 5.2: Scenario 2 representing a road and a sidewalk.

For our study, we decided to divide the calibration process into 3 parts, some of the parameters are fixed and derived from real-world data (length of the car, length of the scooter, vision distance etc.. [31] and are reported in Table 5.2, others are derived from the scenario 1, shown in Figure 5.1, where we have just walls acting as obstacle for all the agents on the scene and the last one from scenario 2, which is the one also including the road and the sidewalk.
For what concerns the agents, for the pedestrian-pedestrian interaction, we have used the parameters defined by Prédhumeau et al. [33], and so our focus shifted to e-scooter-x interactions and the calibration of those parameters.
In Scenario 1, we have an empty space delimited by walls that could represent a typical city centre where there is a shared space between pedestrians, two-wheeled vehicles and cars, but there are no lanes on the ground marking the limits for each agent. Scenario 2, as represented in Figure 5.2, is a different type of shared space, where we have a two-lane road with a sidewalk on its side. E-scooters can ride both on the sidewalk or on the road, depending on their need.
As for the calibration algorithm, we decided to use a Differential Evolution (DE) algorithm, which we think is the most suitable given the parameter space's complexity and its randomness that could help to come out of some local minima. The algorithm is a highly used and well-reviewed Matlab algorithm [10]. Its inputs are the parameters with their limits, the function to be minimized and the population size. The other parameters such as the scaling factor, the crossover rate, the mutation operation and the number of iterations can be modified but we decided to use the standard values. The final values are reported in the Table 5.1. As for the optimization function, we assume that what we want is to minimize the distance between the trajectory extracted from the real data and the simulated one, which can be translated as:

$$F = \sum_{t=0}^{T_{sim}} \sum_{i} \left\| \vec{p}_{i,t}^{real} - \vec{p}_{i,t}^{exp}(\Theta_{bnd}) \right\| \tag{5.1}$$

In this function, Θ_{bnd} refers to the set of all the parameters, in the case of scenario 1, but

the formula would be the same even with Θ_{lanes}.

$$\Theta_{bnd} = [v_{des}, \tau, \sigma, \epsilon, \lambda, \alpha, \kappa, \mu, a, b, \Delta T_a, \Delta T_b] \tag{5.2}$$

$$\Theta_{lanes} = [v_{des}, \tau, \lambda, \alpha, \kappa, \mu, a, b, \Delta T_a, \Delta T_b, \beta_{ext}, \epsilon_{ext}, \beta_{mid}, \epsilon_{mid}] \tag{5.3}$$

As mentioned before, some of those are subject to physical or legal restrictions. All the parameters need to be ≥ 0, but others have also more strict limitations such as:

$$0 \leq v_m \leq 6.94 m/s \tag{5.4}$$

$$0 \leq v_a \leq 13.89 m/s \tag{5.5}$$

$$0 \leq \tau \leq 1 \tag{5.6}$$

The desired speed changes depending on the agent behaviour that we want to emulate. v_m represents the maximum legal speed achievable in France [1] and represents a standard behaviour. On the other hand, v_a represents an aggressive agent behaviour that can go at a non-legal speed. This is important to have, because when we test navigation algorithms for an AV, this irresponsible and unpredictable behaviour is important to be simulated and available, and was also one of the objectives of this work. From this, we derive that we will have 4 sets of calibrated parameters. All the DE runs are repeated 30 times per scenario, with two different rider behaviours, for a total of 120 runs of the algorithm.

Parameter	**e-scooter**	**Pedestrian**
$\vec{v}_{des}$	$6.94/13.89 m/s$	$1.34 m/s$
τ	1.61	1.12
σ	0.2	0.2
ϵ	4	4
λ	0.03	0.08
α	13.5	3
κ	0.46	0.35
a_i	1.25	0.3
b_i	2.03	0.6
ΔT_{a_i}	1.05	0.70
ΔT_{b_i}	1	0.69
μ	5.91	-
β_{ext}	4.89	4.89
ϵ_{ext}	0.48	0.48
β_{mid}	4.19	4.19
ϵ_{mid}	0.28	0.28

Table 5.1: List of the SFM parameter values used in the simulator.

Parameter	e-scooter	pedestrian
length	$116.7cm$	-
weight	$19.9kg$	-
vision distance	$55m$	$10m$
vision angle	$220°$	$220°$
attention distance	$15.5m$	$5m$
attention angle	$100°$	$90°$

Table 5.2: Static parameters value. The vision/attention angle and distance are derived from Anton et al. [31]. The data of the e-scooter from the manufacturer's website [4].

5.1.1 Validation

To validate the model with the computed parameters, the objective is to test the scenarios we have presented before, and check if the simulated trajectory is consistent with the one we can observe in the real-world data.
As mentioned since we could not apply this process to our case, we have used the parameters value from [25] for the e-scooter, since it was an important basis to this study and was one of the few available which could provide a set of calibrated parameters. For those which were not included in that study, we decided to tweak them manually until obtaining the desired behaviour.

5.2 Crash Analysis

In this section, we will present the setup and the results obtained, analyzing the crashes between the agents in the scene, in two different scenarios. This kind of study is important because gives us those cases that are the most challenging.

5.2.1 Setup

To execute this test we have decided to use the two scenarios we presented in the previous section but we added more agents on the scene and considered different cases, depending on how the agents approach each other in the scene.

For scenario 1, we will analyze 3 different cases. The first one is represented in Figure 5.3 and is where the agents will go in a perpendicular direction to the motion of the AV so that they will approach it on the side. Another case is where the pedestrians and the e-scooter go in a parallel direction to the car and will be referred to as case 2 and its setting is displayed in Figure 5.4. This means half the agents will go after the AV, and the other half will be in front of it. Finally, an approach at 45° to the direction of motion, which is case 3 and is represented in Figure 5.5. In these 3 cases what we want to see is the behaviour of both agents, when approaching a moving vehicle that will not stop. What we can expect is that in the first case, a stopping behaviour, or slowing down to then go around the vehicle is the desired behaviour. In the second case, those who are behind the car will either try to pass it or not reach it, while those in front, when they

Figure 5.3: Case 1, where the agents approach the vehicle moving perpendicularly to it. The goal of the car is in front of it while that of the other agent is on the opposite side with respect to their starting point.

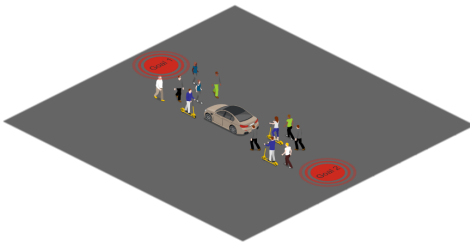


Figure 5.4: Case 2, where the agents approach the vehicle moving parallel to it. The goal of the car is in front of it while that of the other agent is on the opposite side with respect to their starting point.

Figure 5.5: Case 3, where the agents approach the vehicle moving obliquely at 45°. The goal of the car is in front of it while that of the other agent is on the opposite side with respect to their starting point.

notice the car, will move to the side of the road. Eventually, for the third case, also a stopping or avoidance behaviour.

For what concerns scenario 2, we will have 3 cases as well, but in this phase we want to focus more on the interaction between the e-scooter and the pedestrians, to understand how it reacts in those situations. Firstly, when one e-scooter is going in the same direction as the pedestrians and has to perform an overtaking manoeuvre is represented in Figure 5.6. Then when we have multiple e-scooters some of them going in the same direction as the pedestrians, others in the opposite as shown in Figure 5.7. Ultimately, we will analyze how the e-scooter model reacts when riding on the street alongside a car as in Figure 5.8, which is also a common scenario, and is also of interest to look at the reaction when approached by a car.

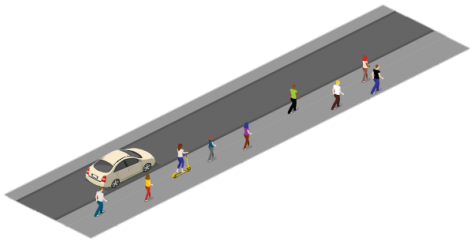


Figure 5.6: Case 1. Single-lane road, with a single-lane sidewalk on its side. All the agents move in the same direction.

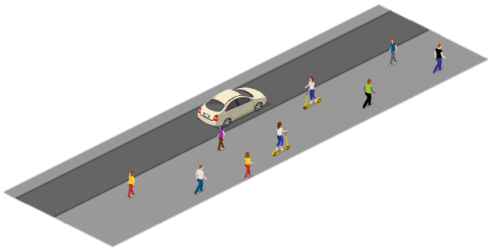


Figure 5.7: Case 2. Single-lane road, two-lane sidewalk. The agents move in all the two possible directions, trying to respect the mid-lane that separates them.

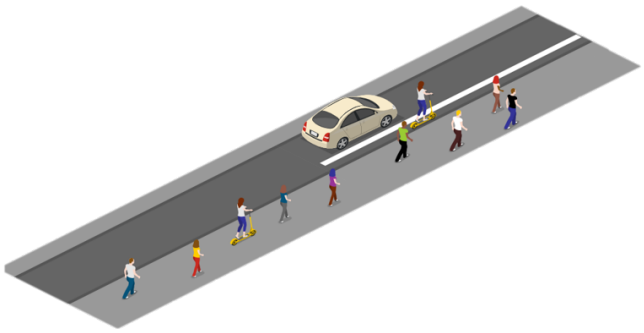


Figure 5.8: Case 3. Single-lane road with the possibility for the e-scooter to ride alongside the cars.

Each one of these tests will be repeated 30 times, with a total of 90 iterations per scenario. Each time the e-scooter rider aggressiveness will be chosen randomly with a 10% possibility of having an agent with aggressive behaviour.

5.2.2 Results Analysis

After performing the previously described tests, in this paragraph, we will discuss the results.

Starting with Scenario 1, we will focus only on the behaviour of the e-scooters since the pedestrians in such conditions have already been studied by Prédhumeau et al. [32]. In the first case, when approaching the car some of the scooters manage successfully to take evading actions and go around the car but some of them try to pass it by going in front of it. This leads to a front crash with the vehicle, which are rare in reality.

Scenario	Case tested	Aggressiveness	Number of crashes	Conditional probability of a crash
1	1	moderate	5	4.6%
1	1	high	3	25%
1	2	moderate	25	23.1%
1	2	high	4	33.3%
1	3	moderate	70	64.8%
1	3	high	8	66.7%
2	1	moderate	20	18.5%
2	1	high	0	0%
2	2	moderate	2	1.9%
2	2	high	0	0%
2	3	moderate	27	25%
2	3	high	6	50%

Table 5.3: Summary table with the results of the crash analysis, after 30 iterations per case. As we can see, from the conditional probability (last column), the level of aggressiveness influences the number of crashes that occur. The total number of agents on the scene was 25 per scene and case. In the total number of crashes, we included only those caused by e-scooters going against other agents.

In the second case throughout the iterations we have something strange happening. Even if the e-scooters that are in front of the vehicle move away from it once they see it, those who are behind it, if they are too close, and faster, sometimes rather than slowing down or going to the side crash into the back of it, which is a strange behaviour that in reality can happen but not as often as what we have experience here.
In the oblique motion (case 3), we experiment the harshest case. Here almost all the e-scooters report a crash with the AV, how we can see from the high percentages reported in Table 5.3. These crashes are due to the e-scooters that instead of waiting for the car to pass them, they try to outspeed it by going around it.
What we can get from these experiments, is that despite most of the iterations we obtain a realistic behaviour that does not lead to crashes or at least to realistic crashes, sometimes we obtain odd behaviours.
Moving to scenario 2, here the results are closer to what we could expect. In the iterations that were performed in the first case, we had a surprisingly outcome. The riders having a moderate aggressiveness, tended to remain stuck behind the pedestrians, if those were spread out occupying the majority of the space. On the other hand, those with an aggressive behaviour, thanks to their higher speed and the need to always achieve it, could manage to rapidly overtake all the pedestrians even when the space on the sidewalk was almost completely occupied. This weird behaviour we encountered was probably due to a lack of correct calibration of the parameters, particularly those representing the sidewalk boundaries.
On the other hand, for the second case, where there are multiple e-scooters and pedestrians, here in some iterations we could experience some crashes as notable from Table 5.3. These are mainly due to the overtaking at the same moment for two e-scooters going in different directions of a group of pedestrians. This lead sometimes to collisions between

the agents due to the way the force acts. What happens, most probably, is that the force on the e-scooter overtaking pushes it towards the opposite lane, as it does the one of the other e-scooters. The problem arises when the two vehicles are close to each other but have not completed the pass on the pedestrians yet. The force exerted by the e-scooter is stronger than the one of the pedestrians, given also a higher personal space. This leads the total force acting on the e-scooter to push it towards the pedestrian and therefore we have a collision. In the other case, when the collision is between the two motorized vehicles, the most common occurrence is when during the overtake the e-scooter moves to the side where there is the other vehicle and so they crash.
Lastly, the third case to consider is what happens if the e-scooter is on the road. Here if the e-scooter is behind the AV, nothing happens since it moves slower than the car. If it is in front, it tends to stay in the middle of the lane, so when the car approaches it has to move towards the border. Sometimes the contrast of the two forces (the repulsive force of the car and that of the lane border) creates the effect of a pendulum which makes the vehicle swerve toward the centre of the lane and therefore creates a collision with the car.
In conclusion what we have obtained from this experiments, can be considered satisfactory, since despite some crashes happening, we have to consider that what we have analyzed is without a proper parameter calibration and so we cannot have a full understanding of what could be the result in the other case. Other than that a diverse way of action of the forces in particular when the e-scooter is on the road could be helpful, without letting it stay in the middle of the lane but creating an imaginary lane thinner than the usual that makes it stays closer to the border of the lane so that when a car is approaching the sum of the two repulsive force could lead to a less strong force.
Despite this, such kind of study can still be interesting and prove that the model we implemented is capable of simulating behaviours such as overtaking and avoiding an obstacle or even braking and changing direction to avoid a crash.

5.3 Test

For what concerns the testing of the model implemented with the calibrated parameters, despite not having real data to use, we still designed a testing scenario to verify the capacity of the work done.

5.3.1 Setup

We designed a scenario, which is more complex than the one we have seen until now, both for the behavioural model and for a hypothetical AV. Our idea is a parking lot, as shown in Figure 5.9 which is a common scenario where agents such as pedestrians and e-scooters act with unpredictable behaviour. This could be a challenging situation for the SFM because, when moving in a parking lot, typically pedestrians and other 2-wheeled vehicles try to go straight to their goal, using also the spots that are not occupied by the cars, while avoiding those who are full. For the autonomous car could be an interesting scenario to test, because not only the agents can follow an unpredictable path to reach

Figure 5.9: Scenario representing a car park, where we have multiple e-scooters and pedestrians navigating through the parked cars.

their goals, but can also be hidden by the parked cars and thus not visible until you are very close to a collision.

Chapter 6

Conclusion

In this chapter we will go through what we have presented previously in this thesis, trying to find a correlation between the objective we were proposed to reach and the model we implemented. To conclude we will delve into the future by analyzing the limitations of this project and what could be the direction to expand the work done here.

6.1 Synthesis

In Chapter 1 we started by saying that in recent years there has been an increase of e-scooters on the road and that the riders do not always manage to follow common road rules creating unpredictable behaviours. We motivated the existence of this work by applying what we previously stated to the context of autonomous cars, in particular by implementing a model that could realistically represent an e-scooter in a simulator, to help those who develop navigation algorithms to have a testing platform.
The objectives of this work were to enrich a simulator by adding a new agent, and new meaningful scenarios and to be able to simulate unpredictable behaviours. This was enclosed in the broader objective of having a comprehensive simulator that could be helpful in the testing of navigation algorithms for autonomous cars.
During the development of this thesis, we have gone through all the different phases involved in developing such a work. Firstly we motivated our impression of a lack of work in the world of simulators focusing on crowded scenarios including different agents, by exploring the main existing simulators. Here it emerged that despite the presence of multiple state-of-the-art simulators in the field of AV navigation algorithms testing, none of them provide the possibility to test them in crowded scenarios and the opportunities provided by SPACiSS. We then presented our e-scooter model going into all the Mathematics and Physics details. Here we have introduced a model that would integrate seamlessly into the existing simulator and provide an updated version with respect to those presented in the literature. Other than that, we proposed a way of designing the limits of road elements such as lanes and sidewalks, that could be integrated with the SFM. Successively we talked about parameter calibration which is a very important aspect of having a well-preforming model when working with an SFM. Here we identified

the parameters that are mostly related to a different rider behaviour and how different people have a unique set of parameters that represent them. We also tested in simulation the possible scenarios where crashes could occur between e-scooters and a moving car and analyzed those cases to have a better understanding of the different scenarios.

6.2 Contributions

The work developed in this thesis has managed to accomplish the goals that had been proposed at the beginning, by providing a new model of an e-scooter obtaining a new implementation that could provide a realistic behaviour of such a vehicle. The same has been done for what concerns the road elements, since in the beginning the simulator we were using was composed just of a space that could resemble a city centre.
The contributions can be summarized as follows:

- Revised dynamic interaction space
- Updated physical model
- Design of a force representing the lanes
- Design of a force representing the sidewalk
- Implementation of the proposed model in a simulator comprising other agents like cars and pedestrians.

6.3 Limitations

As we have discussed in Chapter 5, the main limitation of this work is the lack of real data to be used in the evaluation phase of the model. This is a big letdown because, when proposing a new model is important to test it and evaluate its performance compared to real-world data, to ensure it is as close as possible to real situations that are fundamental in this field. On top of that, working with an SFM implies the need for a very large set of parameters that most of the time are even scenario-dependent and so a precise calibration of those parameters is a basic part when considering such models. This also affected the way we could have different rider behaviours since to obtain differences the only way we had was to tweak the parameters by hand based on the visual feedback we were getting when running the experiments. Unfortunately, this has been a huge limitation, but it was beyond our control and where we did not have any way to accelerate the process.

6.4 Future Work

Future work on this subject could be firstly focused on obtaining real-world data to calibrate the parameters and validate the model. After that, one could follow the same approach proposed here to add more functionalities and complexity to the simulator.

Firstly by focusing on new agents that could be added, such as bicycles both electric and classic and other PMVs that can be commonly found in shared spaces such as skateboards or monocycles. After that, another area that could be expanded regards the road elements, where could be introduced traffic lights or roundabouts, which are common zones where is important to test navigation algorithms and having a realistic behaviour of more fragile agents is important to consider, since those are scenarios where those kind of vehicles do not always follow road rules.

6.5 Final remarks

In conclusion, we can say we are satisfied with the work proposed in this thesis. Despite not having the possibility of validating the model we still defined a new approach in a field that as of now is not widely explored despite the continuous increase of e-scooters on the road.

Bibliography

[1] Circulation en trottinette électrique, rollers ou skateboard. https://www.service-public.fr/particuliers/vosdroits/F308, accessed August 12, 2024.

[2] "pedsim_ros". Accessed 29/07/2024, online at: https://github.com/srl-freiburg/pedsim_ros.

[3] Ros. Accessed 29/07/2024, online at: https://www.ros.org/.

[4] Segway scooter webpage. https://fr-fr.segway.com/products/ninebot-kickscooter-max-g30-1. [Online; accessed 21/08/2024].

[5] Annapolis. https://project.inria.fr/annapolis/, 2022. [Online; accessed 18/08/2024].

[6] The e-scooter trend - a popular mobility alternative. https://www.iaa-mobility.com/en/newsroom/news/urban-mobility/trend-t-scooter, 2024. [Online; accessed 18/08/2024].

[7] 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.

[8] Bani Anvari. Modelling crowd dynamics. Accessed 05/08/2024, online available at: https://bani-anvari.com/research/agent-based-modelling/modelling-crowd-dynamics/.

[9] Elisa Bassoli and Loris Vincenzi. Parameter calibration of a social force model for the crowd-induced vibrations of footbridges. *Frontiers in Built Environment*, 7:656799, 2021.

[10] Markus Buehren. Differential evolution, 2024. https://www.mathworks.com/matlabcentral/fileexchange/18593-differential-evolution, MATLAB Central File Exchange. Retrieved August 12, 2024.

[11] Carsten Burstedde, Kai Klauck, Andreas Schadschneider, and Johannes Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295(3-4):507–525, 2001.

[12] Xu Chen, Martin Treiber, Venkatesan Kanagaraj, and Haiying Li. Social force models for pedestrian traffic–state of the art. *Transport reviews*, 38(5):625–653, 2018.

[13] Geng Cui, Daichi Yanagisawa, and Nishinari Katsuhiro. A data driven approach to simulate pedestrian competitiveness using the social force model. *Collective Dynamics*, 6:1–15, 2021.

[14] Julia Damerow. John von neumann's cellular automata, June 2010. Accessed 30/07/2024, available online at https://embryo.asu.edu/pages/john-von-neumanns-cellular-automata.

[15] Charitha Dias, Miho Iryo-Asano, Hiroaki Nishiuchi, and Tomoyuki Todoroki. Calibrating a social force based model for simulating personal mobility vehicles and pedestrian mixed traffic. *Simulation Modelling Practice and Theory*, 87:395–411, 2018.

[16] Jan Dijkstra, AJ Jessurun, and Harry JP Timmermans. A multi-agent cellular automata model of pedestrian movement. In *Pedestrian and evacuation dynamics*, pages 173–181. Springer, 2001.

[17] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[18] Epic Games. Unreal engine.

[19] Tully Foote. tf: The transform library. In *Technologies for Practical Robot Applications (TePRA), 2013 IEEE International Conference on*, Open-Source Software workshop, pages 1–6, April 2013.

[20] Golnaz Habibi, Nikita Jaipuria, and Jonathan P How. Context-aware pedestrian motion prediction in urban intersections. *arXiv preprint arXiv:1806.09453*, 2018.

[21] Zhitong He and Lingxi Li. Simulation framework for vehicle and electric scooter interaction. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 4479–4484. IEEE, 2023.

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

[23] Alireza Jafari and Yen-Chen Liu. A heterogeneous social force model for personal mobility vehicles on futuristic sidewalks. *Simulation Modelling Practice and Theory*, 131:102879, 2024.

[24] Anders Johansson, Dirk Helbing, and Pradyumn K Shukla. Specification of the social force pedestrian model by evolutionary adjustment to video tracking data. *Advances in complex systems*, 10(supp02):271–288, 2007.

[25] Yen-Chen Liu, Alireza Jafari, Jae Kun Shim, and Derek A. Paley. Dynamic modeling and simulation of electric scooter interactions with a pedestrian crowd using a social force model. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):16448–16461, 2022.

[26] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo. In *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[27] Sebastien Loze. "carla democratizes autonomous vehicle r&d with free open-source simulator", October 2019. Online available at: https://www.unrealengine.com/en-US/spotlights/carla-democratizes-autonomous-vehicle-r-d-with-free-open-source-simulator.

[28] Sumbal Malik, Manzoor Ahmed Khan, and Hesham El-Sayed. Carla: Car learning to act. an inside out. *Procedia Computer Science*, 198:742–749, 2022. 12th

International Conference on Emerging Ubiquitous Systems and Pervasive Networks / 11th International Conference on Current and Future Trends of Information and Communication Technologies in Healthcare.

[29] 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):e10047, 2010.

[30] Daniel R Parisi, Marcelo Gilman, and Herman Moldovan. A modification of the social force model can reproduce experimental data of pedestrian flows in normal conditions. *Physica A: Statistical Mechanics and its Applications*, 388(17):3600–3608, 2009.

[31] Anton Pashkevich, Tomasz E. Burghardt, Sabina Pulawska-Obiedowska, and Matus Sucha. Visual attention and speeds of pedestrians, cyclists, and electric scooter riders when using shared road a field eye tracker experiment. *Case Studies on Transport Policy*, 10(1):549–558, 2022.

[32] Manon Prédhumeau. *Modelisation et simulation de comportements pietons realistes en espace partage avec un vehicule autonome*. PhD thesis, 12 2021.

[33] Manon Prédhumeau. Simulating Realistic Pedestrian Behaviors in the Context of Autonomous Vehicles in Shared Spaces. In *20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021)*, Online, France, May 2021. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS).

[34] Rajesh Rajamani. *Vehicle dynamics and control*. Springer Science & Business Media, 2011.

[35] Guodong Rong, Byung Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Martins Mozeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, Eugene Agafonov, Tae Hyung Kim, Eric Sterner, Keunhae Ushiroda, Michael Reyes, Dmitry Zelenkovsky, and Seonman Kim. Lgsvl simulator: A high fidelity simulator for autonomous driving. pages 1–6, 09 2020.

[36] Siamak Sarmady, Fazilah Haron, and Abdullah Zawawi Talib. Simulation of pedestrian movements using fine grid cellular automata model. *arXiv preprint arXiv:1406.3567*, 2014.

[37] Niketan Sharma. escooter statistics you need to know in 2024. https://www.nimbleappgenie.com/blogs/escooter-statistics/, April 2024. [Online; accessed 18/08/2024].

[38] Pradyumn Kumar Shukla. *Modeling and simulation of pedestrians*. PhD thesis, Diploma Thesis, 2005.

[39] Unity Technologies. Unity.

[40] Yeltsin Valero, Adrien Antonelli, Zoi Christoforou, Nadir Farhi, Bachar Kabalan, Christos Gioldasis, and Nicolas Foissaud. Adaptation and calibration of a social force based model to study interactions between electric scooters and pedestrians. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–7. IEEE, 2020.

[41] Wikipedia contributors. Robot operating system — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Robot_Operating_

System&oldid=1238209286, 2024. [Online; accessed 17-August-2024].

[42] Yanyu Xu, Zhixin Piao, and Shenghua Gao. Encoding crowd interaction with deep neural network for pedestrian trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[43] Dongfang Yang, Ümit Özgüner, and Keith Redmill. A social force based pedestrian motion model considering multi-pedestrian interaction with a vehicle. *ACM Transactions on Spatial Algorithms and Systems (TSAS)*, 6(2):1–27, 2020.

[44] Shuai Yi, Hongsheng Li, and Xiaogang Wang. Pedestrian behavior understanding and prediction with deep neural networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 263–279. Springer, 2016.

[45] JKK Yuen and EWM Lee. The effect of overtaking behavior on unidirectional pedestrian flow. *Safety Science*, 50(8):1704–1714, 2012.

[46] Weiliang Zeng, Peng Chen, Hideki Nakamura, and Miho Iryo-Asano. Application of social force model to pedestrian behavior analysis at signalized crosswalk. *Transportation Research Part C: Emerging Technologies*, 40:143–159, 2014.