Politecnico di Milano

School of Industrial and Information Engineering
Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB)
Master of Science in Telecommunication Engineering

Vehicular Mobility Modeling and Map-Assisted Vehicular Trajectory Prediction System in Realistic Simulated Traffic Scenario

Supervisor:
Chiar.mo Prof. Umberto SPAGNOLINI

Co-Supervisor:
Dott. Ing. Marouan MIZMIZI

Master Thesis Candidate:
Claudio ANTONUZZO
Student ID:
893887

Academic Year 2020-2021

A mio Papà e a mia Mamma,
che mi sono stati accanto
nel momento più difficile
della mia vita.

Acknowledgements

Vorrei sfruttare questo spazio a disposizione all'interno del mio lavoro di Tesi per ringraziare alcune persone che, per una ragione o per un'altra, sono state molto importanti, in misura diretta o indiretta, per il contributo e il sostegno che mi hanno dato, durante questo lungo viaggio, per raggiungere un traguardo così importante per la mia vita come la Laurea Magistrale.

Per prima cosa, ringrazio il relatore della mia tesi, Professore Umberto Spagnolini, per avermi dato l'importante opportunità, unica nel suo genere, di lavorare, fare esperienza e confrontarmi all'interno di un team di ricerca costituito da persone preparate, competenti e precise nel loro lavoro, su attività e progetti di ricerca importanti, interessanti ed entusiasmanti che sicuramente, negli anni a venire, troveranno numerosi campi di applicazione pratici nella vita di tutti i giorni.
Ringrazio anche il correlatore della mia tesi, Ing. Marouan Mizmizi, che mi ha seguito per tutta la durata del progetto di ricerca e del mio lavoro, dandomi suggerimenti utili che mi hanno aiutato a comprendere quali fossero le giuste direzioni da seguire per la stesura del lavoro.

Ringrazio la mia Famiglia per tutto il supporto, il conforto, i suggerimenti, i consigli e l'affetto che non mi hanno mai fatto mancare in tutti questi mesi, nonostante il periodo difficile dovuto al covid e al distanziamento sociale.
Ringrazio quindi mio Papà e mia Mamma, i quali mi hanno incitato sempre a non mollare mai, anche e soprattutto nei momenti più difficili e bui, nei quali avrei voluto mollare, facendomi comprendere che nella vita posso raggiungere qualsiasi obiettivo e posso fare qualsiasi cosa, se lo voglio e se mi dò la possibilità di farlo.
Senza di loro, se fossi stato solo, probabilmente non ce l'avrei mai fatta.
Ringrazio mio fratello Andrea, con il quale mi confido praticamente su tutto, confrontandoci e supportandoci su tutti i fronti nonostante il poco tempo e la distanza, per avermi aiutato a comprendere l'importanza della dedizione, della cultura del lavoro, del sacrificio, della fatica e della determinazione se si vuole raggiungere un determinato obiettivo, usando sempre le parole giuste al momento giusto, dimostrandomi di essere una persona sulla quale potrò sempre contare nella mia vita.
Ringrazio, infine, mia Zia e mio Zio per la loro discrezione, in quanto, nonostante ci sentiamo poco e nonostante la distanza che ci separa, il loro affetto e sostegno nei miei confronti rimane immutato, come se ci sentissimo e vedessimo tutti i giorni, accontentandosi del troppo poco tempo che dedico a loro. Vi voglio bene.

Ringrazio i miei colleghi di corso, che mi hanno permesso di vivere il periodo universi-

tario con più leggerezza e spensieratezza, che mi hanno accompagnato durante tutto il corso di Laurea Magistrale e che mi hanno aiutato, sin dall'inizio del mio percorso, ad integrarmi in un ambiente completamente nuovo.

Dedico un pensiero anche ai miei coinquilini, con cui ho passato praticamente la maggior parte del periodo universitario, condividendo tempi, spazi e sani momenti di leggerezza, e dai quali ho potuto imparare molte cose nuove, tra cui l'importanza dell'integrazione tra lingue e culture diverse, il che mi ha fatto crescere molto sotto il profilo umano, dell'esperienza, dell'apertura mentale e della capacità di essere più pazienti e riflessivi.

Per concludere, ringrazio i miei Amici, senza i quali non avrei mai potuto affrontare questo viaggio lungo e tortuoso. Mi riferisco soprattutto a quelli che ho conosciuto poco prima che scoppiasse la pandemia di covid, con i quali ci siamo supportati e confortati a vicenda, nonostante la distanza per lunghi tratti, in maniera decisiva per affrontare quello che è destinato a rimanere uno dei periodi più difficili e incerti delle nostre vite. Non so come si sarebbe sviluppato il rapporto con loro se non ci fosse stata la pandemia, ma so per certo che quel frangente ci ha resi più uniti, solidali ed empatici tra di noi, il che ha consolidato ancor di più il nostro legame.

Contents

List of Figures

List of Equations

List of Tables

Acronyms

5GAA	5G Automotive Association
6G	6-th Generation Mobile Radio Network
AoI	Area of Interest
BEV	Bird's Eye View
C-ITS	Cooperative Intelligent Transportation Systems
CA	Constant Acceleration
CAS	Cooperative Awareness Sensing
CAV	Connected and Automated Vehicles
CFM	Car-Following Models
CNN	Convolutional Neural Network
CONTRAM	CONtinuous TRaffic Assignment Model
CSI	Channel State Information
CTM	Cell Transmission Model
CTRA	Constant Turn Rate and Acceleration
CTRV	Constant Turn Rate and Velocity
CT	Coordinated Turn
CV	Constant Velocity
DBN	Dynamic Bayesian Networks
DL	Deep Learning
eMBB	enhanced Mobile Broadband
FC	Fully-Connected
FF	Feed-Forward
FR1	Frequency Range 1
FR2	Frequency Range 2
GNN	Graph Neural Networks
GP	Gaussian Process
GRU	Gated Recurrent Unit
HMM	Hidden Markov Model

IDM	Intelligent Driver Model
IMM	Interactive Multiple Model
JSD	Jensen-Shannon Divergence
LoA	Levels of Automation
LR	Logistic regression
LSTM	Long Short-Term Memory
LWR	Lighthill-Whitham-Richards
MAP	Maximum A-Posteriori
ML	Machine Learning
MM	Multi-model
NN	Neural Network
NR	New Radio
NV	Non Effective Vehicles
OBU	On-Board Unit
OSM	OpenStreetMap
pdf	Probability Density Function
PoI	Point of Interest
QoS	Quality of Service
ReLU	Rectified Linear Unit
RMSE	Root Mean Squared Error
RNN	Recurrent Neural Network
RVM	Relevance Vector Machines
SNR	Signal-to-Noise Ratio
SoA	State of the Art
SUMO	Simulation of Urban MObility
SV	Surrounding Vehicles
SVM	Support Vector Machines
TV	Target Vehicle
UTM	Universal Transverse of Mercator
URLLC	Ultra-Reliable Low-Latency Communications
V2X	Vehicle-to-Anything
VANETs	Vehicular Ad-Hoc-Networks
VRU	Vulnerable Road Users
VTP	Vehicular Trajectory Prediction

Abstract

English Version

The impact of the 6G cellular standard will revolutionize the future of the automotive field. The development of Cooperative Intelligent Transportation Systems is crucial to improve safety, efficiency, and comfort of mobility, reduce costs, traffic congestion, and pollution, lead to higher Levels of Automation of the vehicles, and lower human-controlled driving.
To reach these goals, vehicles should autonomously establish reliable V2X communication links in the long term. For future V2X communications, the mm-Wave is a promising solution to guarantee high data rate and low latency links.
However, in a vehicular context, V2X systems in mm-Wave bands encounter many challenges due to the physical characteristics of the communication channel and mobility. Indeed, the path loss in these frequencies leads to beamforming techniques to reduce the outage probability.
Unfortunately, this needful choice introduces a higher sensitivity to blockage due to the difficulty of predicting mobility. If not adequately handled, this aspect may lead to highly unreliable and inefficient V2X communication, with detrimental consequences for traffic safety and the quality of service of the V2X use cases.
Thus, mobility and trajectory prediction of dynamic agents, i.e., vehicles, pedestrians, is essential to predict the evolution of the physical propagation channel and the position of static and dynamic blockers. This knowledge allows us to predict link blockage and channel state, which are crucial to proactively set up more robust V2X communication links that guarantee higher reliability.
This master thesis goes in this direction, aiming to study vehicular mobility modeling and proposing a map-assisted vehicular trajectory prediction system to support future 6G mm-Wave V2X communications.
The system exploits vehicles' state information and digital maps to perform a manoeuvre-based trajectory prediction, exploiting also ML techniques and Bézier curves.
Results obtained from simulations performed in realistic intersection scenario show that the system, in case of turnings, provides mean and RMSE prediction errors lower than 10 m for a prediction horizon up to 5 s.

Italian Version

Lo standard di reti mobili 6G avrà un forte impatto nel settore automotive. Lo sviluppo dei sistemi di trasporto intelligente è cruciale per la sicurezza e il comfort della mobilità, per ridurre costi, traffico, inquinamento, e per raggiungere livelli di guida autonoma sempre più alti.
Per ottenere questi risultati, i veicoli dovrebbero configurare, in autonomia, canali di comunicazione affidabili nel lungo periodo. In questo ambito, le onde millimetriche sono una soluzione valida per garantire alto ritmo di trasmissione e bassa latenza.
Tuttavia, nelle bande di frequenza delle onde millimetriche, i sistemi di comunicazione veicolare sono soggetti a numerosi problemi dovuti alle caratteristiche fisiche del canale e alla mobilità. Le attenuazioni in tali bande, infatti, costringono all'uso di tecniche di beamforming per ridurre la probabilità di interruzione del servizio.
Purtroppo, tale scelta comporta una maggiore sensibilità del sistema alle ostruzioni del canale, dovute alla difficoltà nel prevedere la mobilità, con la conseguente riduzione di affidabilità dei collegamenti.
Pertanto, la predizione della mobilità di veicoli e pedoni è essenziale per la predizione delle ostruzioni e per l'evoluzione del canale. Questa conoscenza, infatti, permette ai veicoli di configurare, in tempo utile, canali di comunicazione più robusti e affidabili.
La seguente tesi si pone l'obiettivo di studiare la mobilità veicolare e propone un sistema di predizione di traiettorie veicolari, assistito da mappe digitali, con l'obiettivo di supportare i futuri canali di comunicazione veicolare in ambito 6G.
Il sistema sfrutta informazioni sulla dinamica dei veicoli e le mappe digitali per predire la mobilità veicolare, sfruttando anche tecniche ML e le curve di Bézier.
Risultati provenienti da simulazioni condotte in un incrocio realistico mostrano che il sistema, per i veicoli che svoltano, raggiunge valor medi e deviazioni standard dell'errore di predizione più bassi di 10 m per un orizzonte temporale fino a 5 s.

Chapter 1

Introduction

The development of the 6-th Generation Mobile Radio Network (6G) cellular standard will benefit the economy and the quality of our lives in the future [1].
Indeed, with 6G, it will be possible to provide to the community a large number of new services and applications we could not even imagine by exploiting previous wireless technologies, thanks to the flexibility of the new physical layer architecture.
Without any doubt, the impact of 6G will revolutionize the future of the automotive field. The development of Cooperative Intelligent Transportation Systems (C-ITS) is crucial to improve safety, efficiency, and comfort of mobility, reduce costs, traffic congestion, and pollution, lead to higher Levels of Automation (LoA) of the vehicles and lower human-controlled driving [2].
To reach these goals, it is crucial that vehicles can autonomously establish reliable Vehicle-to-Anything (V2X) communication links in the long term. In particular, for future V2X communications, Frequency Range 2 (FR2) at 26-28 GHz, also called Millimeter Wave - mm-Wave, is a promising solution to guarantee high data rate and low latency links, meeting the stringent requirements needed to support those V2X uses cases belonging to the area between Ultra-Reliable Low-Latency Communications (URLLC) and enhanced Mobile Broadband (eMBB) services [3][4][5], i.e., extended sensing, cooperative manoeuvre, platooning.
So, mm-Wave communication systems can provide high data rate links, suitable for the Quality of Service (QoS) requirements of the aforementioned V2X use cases. However, in vehicular context, V2X systems in mm-Wave bands encounter many challenges due to the physical characteristics of the communication channel and mobility.
Indeed, the path loss in these frequencies, and the increased amount of thermal noise and interference due to larger bandwidths, lead to beamforming techniques to compensate for these detrimental effects and reduce the outage probability. Unfortunately, this needful choice introduces a higher sensitivity to blockage due to the difficulty of predicting mobility.
If not adequately handled, this aspect may lead to highly unreliable and inefficient V2X communication, with detrimental consequences for traffic safety and QoS of the V2X services.
Thus, mobility and trajectory prediction of dynamic agents, i.e., non-Connected and Automated Vehicles (CAV) or Vulnerable Road Users (VRU) like bikers, pedestrians, riders, is essential to predict the evolution of the physical propagation channel and

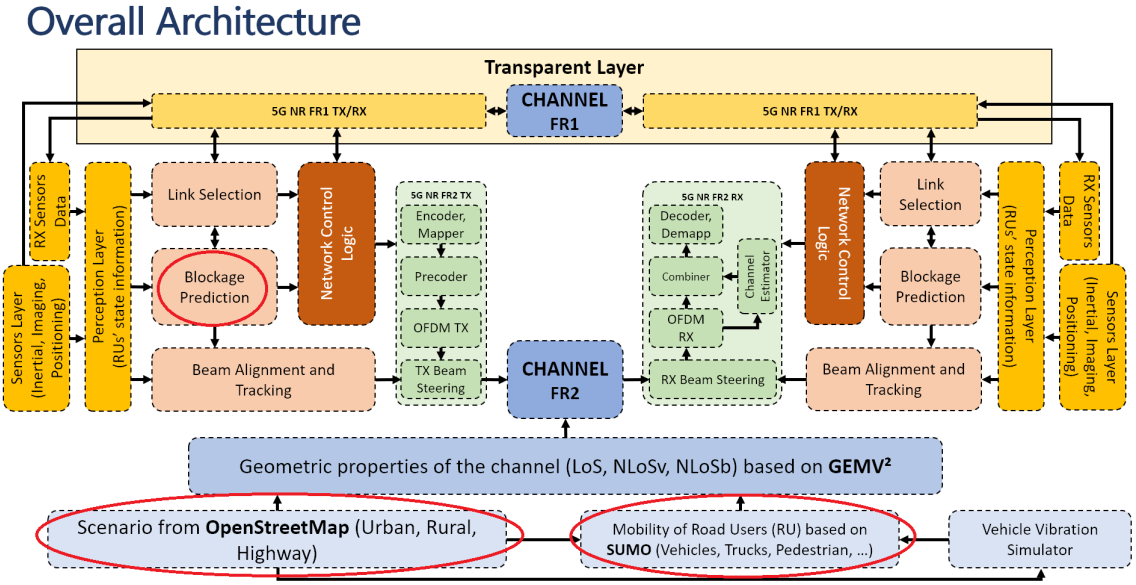


Figure 1.1. Overall architecture of 5G-NR-FR1/FR2-V2X Project. My thesis is focused mainly to support the red circled building blocks.

the position of static and dynamic blockers in the vehicular scenario. This knowledge allows us to accurately predict link blockage and other relevant network metrics, such as Channel State Information (CSI), Signal-to-Noise Ratio (SNR), and link availability, which are crucial to proactively set up more robust V2X communication links that guarantee higher reliability.
This master thesis goes in this direction, aiming to study vehicular mobility modeling and proposing a map-assisted Vehicular Trajectory Prediction (VTP) system to support future 6G mm-Wave V2X communications.
The work takes place in 5G-New Radio (NR)-FR1/FR2-V2X Project in the context of "Huawei-Polimi" Joint Lab, whose goal, as briefly shown in Figure 1.1, is to simulate a vehicular network on top of realistic traffic scenarios (specifically highway, urban and rural) and implement predictive solutions to set up reliable mm-Wave V2X communications.
The proposed strategy is expected to mitigate the detrimental effects of link blockage, guaranteeing the setup of V2X communication links and meeting the stringent URLLC and eMBB requirements in the long term.
The rest of the work is organized as follows:

- Chapter 2 provides a State of the Art review on vehicular mobility modeling and vehicular/pedestrian trajectory prediction.
  In particular:
  - Section 2.1 gives a description and comparison of main mobility models available in the literature, outlining the models implemented in Simulation of Urban MObility (SUMO) [6][7], the software used to generate realistic traffic simulations, and proposes a methodological approach that we should follow in order to build and/or adequately choose a mobility model to simulate a realistic vehicular scenario;

  - Sections 2.2 and 2.3 propose a general review and a detailed investigation of the main vehicular and human motion and trajectory prediction models available to the research community.
    This review is critical in order to point out the advantages and drawbacks of using a specific model in place of other models in order to find the best strategy for the implementation of the map-assisted VTP system.

- Chapter 3 describes the strategy we follow for the implementation of the map-assisted VTP system.
  In particular:

  - Section 3.1 outlines the proposed strategy, with a detailed description of all the passages involved for trajectory prediction.
    To briefly introduce the topic, we initially assume that at a specific time instant t_0, we have some CAVs and non-CAVs in a traffic scenario. The prediction system knows all vehicles' current and past state (position and vehicular dynamics), simulating that CAVs have already estimated and exchanged information about their current state, together with their accuracies, through a Cooperative Awareness Sensing (CAS) procedure.
    Given these assumptions, the system inputs vehicles' state information and the digital maps provided in the simulation by OpenStreetMap (OSM) [8], and performs a manoeuvre-based trajectory prediction of the vehicles involved in the scenario.
    Specifically, the system first predicts the future vehicle manoeuvre. Then, according to the predicted manoeuvre, it provides the output predicted trajectory;
  - Section 3.2 outlines the proposed methods for manoeuvre prediction. Specifically, two strategies are implemented and compared: a Model-based approach and a Machine Learning (ML) based approach;
  - Section 3.3 describes the proposed method for trajectory prediction. In particular, the strategy involves the combination of information from digital maps and Bézier curves to predict future vehicle trajectory [9][10].

- Chapter 4 illustrates some results about implementing the map-assisted VTP system on manoeuvre and trajectory prediction, obtained from a simulation performed in Milan. Specifically, we illustrate the results on manoeuvre and trajectory prediction in Sections 4.1 and 4.2, respectively.

- Chapter 5 concludes this master thesis, summarizing what has been done so far and giving some suggestions and proposals for future research.

Chapter 2

Mobility Modeling and trajectory prediction

This chapter provides an overview of the State of the Art (SoA) on vehicular mobility modeling and vehicular/Pedestrian trajectory prediction.
In particular:

- Section 2.1 gives an outline of vehicular mobility modeling by investigating the main classification methodologies, describing and comparing the mobility models commonly implemented in simulated scenarios and finally giving some insights about SUMO and its mobility description;
- Sections 2.2 and 2.3 respectively propose a general review and then a more detailed investigation on the main vehicular and human motion and trajectory prediction models available to the research community, with the final goal of finding the best approach for VTP system implementation.

2.1 Vehicular Mobility Modeling

A study on mobility models is very important for V2X communications. Indeed, to obtain reliable results in V2X research activities it is crucial to test and evaluate technologies and protocol implementations in real testbed environments, but logistic difficulties, economic issues and technology limitations encountered here make simulations the unique mean of choice in the validation of networking protocols for Vehicular Ad-Hoc-Networks (VANETs), and a widely adopted first step in development of real world technologies [11][12].
So, it is important to study and define vehicular mobility models providing an accurate and realistic mobility description, at both microscopic and macroscopic levels, in order to guarantee realistic simulated scenarios that reflect real behavior of vehicular traffic.
In literature, a common classification is based on the level of detail of motion representation, distinguishing between macroscopic, mesoscopic and microscopic levels of analysis.
Accordingly, mobility models can be separated into the following categories [12][13] (Fig. 2.1):

1. *Macroscopic models*: vehicular traffic is represented as a hydrodynamic phenomenon, where flows of cars move along roads similarly to fluids within tubes. Traffic is regarded as continuous flow, and gross quantities of interest are modeled, such as the density or the mean velocity of cars, often using formulations borrowed from fluid dynamics theory, not providing information about individual vehicles, but just an aggregate overview.
   Some examples are Lighthill-Whitham-Richards (LWR) model, Cell Transmission Model (CTM), Link-Node CTM;

2. *Mesoscopic models*: It is a middle layer between macroscopic and microscopic models, where individual mobile entities are modeled at an aggregate level, exploiting gas-kinetic and queuing theory results or macroscopic-scale metrics, such as velocity/density relationships, to determine the motion of vehicles.
   Some examples are CONtinuous TRaffic Assignment Model (CONTRAM), DynaMIT model, Semi-Poisson Buckley Model;

3. *Microscopic models*: Each vehicle is independent, an autonomous entity, together with its movement which is represented in great detail, and its dynamics is treated independently from those of other cars, except for those near enough to have a direct impact on driver's behavior.
   Microscopic models are able to reproduce fine-grained real-world situations, such as front-to-rear car interaction, lane changing, flows merging at ramps, intersections, with complex acceleration and overtaking behavior description, which results in different speeds by cars travelling within the same road segment.
   The most famous microscopic models are mainly Car-Following Models (CFM) such as Krauss model (the most important and used one) [14], Nagel-Schreckenberg model [15], Wiedemann Psycho-Physical Model [16], Intelligent Driver Model (IDM) [17], Gipps model [18]. They describe the behavior of each driver in relation to its neighboring vehicles. As they regard each car as an independent entity, they fall into the category of interaction-aware microscopic-level descriptions.
   The first CFM models date back to the late 1950s, and since then they have been the most popular methods to analytically represent vehicular traffic dynamics. Many of them determine the motion of a vehicle as a function of the state of a single neighboring car, typically the one in front. For this reason, they are also referred to as Follow-the-Leader models. In such descriptions, the speed or acceleration depend on factors such as the distance from the front car and the absolute and relative speed or acceleration of both vehicles.

The three classes of models above have advantages and disadvantages.
On one hand, macroscopic models only provide an aggregate and high-level view of the system, but are mathematically tractable and can be simulated at minimal computational cost.
On the other hand, microscopic models can be extremely detailed, but they also require significant processing power to be run at large scales. Clearly, mesoscopic models fall in between the other two classes.
While macroscopic and mesoscopic descriptions are typically employed to capture the dynamics of large-scale vehicular systems, microscopic ones are usually applied to

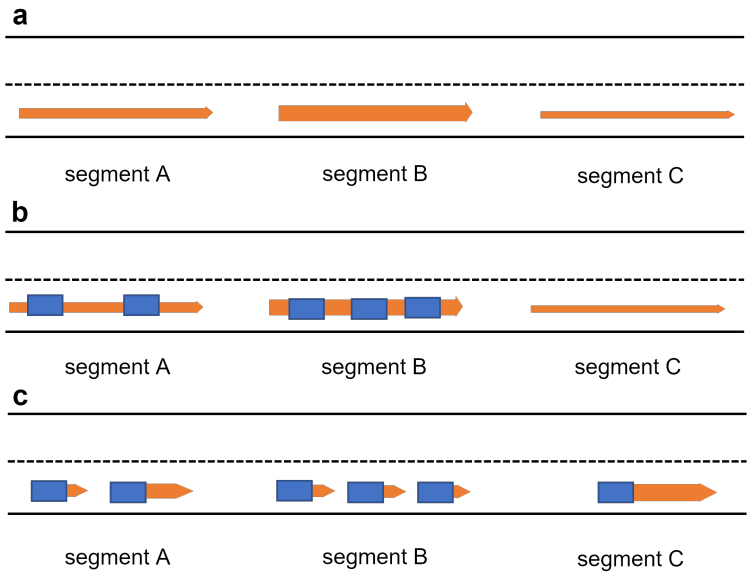


Figure 2.1. Classification of vehicular mobility models according to their level of detail: (a) Macroscopic. (b) Mesoscopic. (c) Microscopic [13].

reproduction of traffic in smaller areas, such as single highways or urban areas.
However, in the case of VANETs simulation the choice to be done for mobility description is mandated by the fact that vehicle-to-vehicle communications have ranges in the order of hundreds of meters. This means that the precision in representing the position of each vehicle needs to be in the order of the meter or less; otherwise, inaccuracies in the mobility representation risk to bias the communication performance.
In other words, vehicular network simulations often require a high level of detail in terms of car motion representation because of the reduced spatial scale of short- and middle-range communication techniques that must be implemented. The necessity of precision in the order of sub-meters in the definition of vehicles' absolute and relative positions bounds the mobility descriptions to be used for network simulation to the microscopic domain.
As a result, the only sensible choice is that of microscopic models, that consider individual vehicles and can thus output their actual location rather than an approximate one [12].
For further details on the main characteristics and peculiarities of the vehicular mobility models we described, there is some available material that offers exhaustive description [13][19][20][21].
In the context of microscopic modeling, several components should be taken into account in order to obtain a more realistic representation of road traffic:

1. We need a faithful description of the road infrastructure. This description should include not only the road layout and the network topology, but also, speed limits, one-way constraints, traffic lights at intersections and their temporization, stop and yield signs, roundabouts, overpasses, highways ramps, traffic rules, etc.

2. The microscopic behavior of each driver must be modeled accurately. The acceleration and speed of each vehicle must be the result of its interactions with

surrounding cars, road infrastructure and road signalization.

So, the traditional branching of models into macroscopic, mesoscopic, and microscopic should be integrated in a broader context when we need to provide a realistic representation of road traffic.
The work in [11] tries to go in this direction since it proposes a more comprehensive framework that outlines the main aspects we should consider to better differentiate, classify, and evaluate the degree of realism of the different mobility models, identifying the so-called **Functional blocks**, whose main actors are **Motion Constraints** and **Traffic Generator**:

1. On one hand, **Motion constraints** describe the relative degree of freedom of each vehicle. Macroscopically, motion constraints are streets or buildings, but microscopically, constraints are modeled by neighboring cars, pedestrians, or by modelization's diversities either due to the type of car or to the driver's habits.

2. On the other hand, **Traffic generator** defines different kinds of cars and deals with their interactions according to the environment under study. Macroscopically, it models traffic densities, speeds and flows, while microscopically it deals with properties like the inter-distance between cars, acceleration, braking, overtaking.

Another important aspect of realistic motion modeling is time, which can be seen as the third functional block that describes different mobility configurations for a specific time of the day or day of the week.
Finally, we also have a fourth fundamental block, the External Influence, modeling the impact of a communication protocol or any other source of information on the motion patterns.
According to the concept map in Fig.2.2 [11], mobility models intended to describe

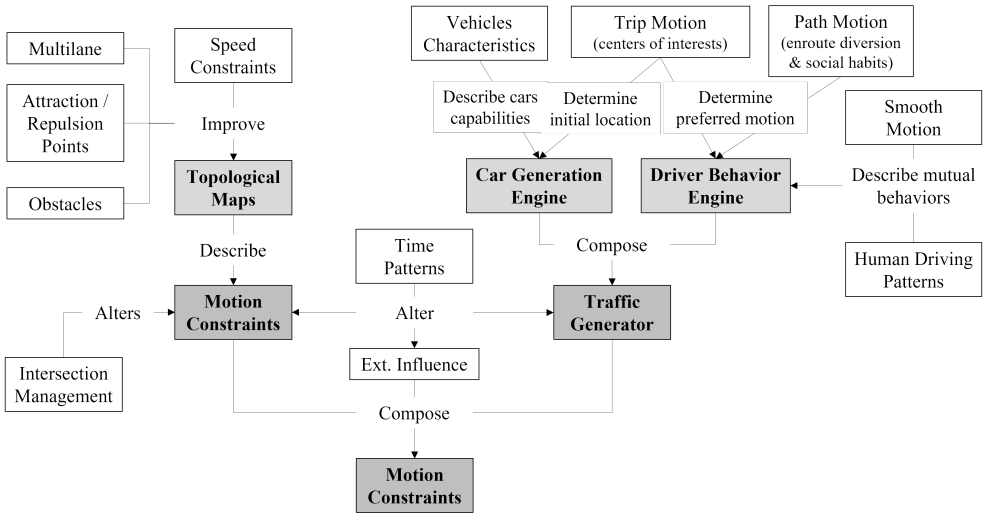


Figure 2.2. Concept map proposed in [11] for the generation, classification and/or proper choice of realistic vehicular mobility models.

realistic vehicular motion patterns should account for the following building blocks:

- Accurate and realistic topological maps: street topologies should manage different densities of intersections, contain multiple lanes, different categories of streets and their associated speed limitations.

- Obstacles: obstacles should be considered as constraints to cars mobility.

- Attraction/repulsion points: initial and final destinations of road trips are not random. Most of the time, drivers are moving to similar final destinations, called attraction points (e.g. office), or from similar initial locations, called repulsion points (e.g. home), a feature that creates bottlenecks.

- vehicles characteristics: each category of vehicle has its own characteristics, which has an impact on a set of traffic parameters. For example, macroscopically speaking, some urban streets and highways are forbidden to trucks depending on the time of the day. Microscopically speaking, acceleration, deceleration and speed capabilities of a car or a truck are different. Accounting for these characteristics alters the traffic generator engine when modeling realistic vehicular motions.

- Trip motion: a trip is macroscopically seen as a set of source and destination points in the urban area. Different drivers may have diverse interests, which affect its trip selection.

- Path motion: a path is macroscopically seen as the set of road segments taken by a car on its trip between a source and a destination point. As it may also be observed in real life, drivers do not randomly choose the next heading when reaching an intersection, as it is currently the case in most vehicular networking traffic simulations. Instead, they choose their paths according to a set of constraints such as speed limitations, time of the day, road congestion, distance, and even drivers' personal habits.

- Smooth deceleration and acceleration: vehicles do not abruptly break and accelerate. Models for decelerations and accelerations should consequently be considered.

- Human driving patterns: drivers interact with their environments, not only with respect to static obstacles, but also to dynamic ones, such as neighboring cars and pedestrians. Accordingly, the mobility model should control the mutual interactions between vehicles, such as overtaking, traffic jam, preferred paths. This is the category which is nearest to the mobility description previously mentioned and has the most significant impact on the realism of vehicular mobility. Indeed, vehicles are involved in a complex interaction and they are controlled by human beings. This interaction is often referred to as micro-mobility, as it refers to the control of acceleration, deceleration levels and reaction time in order to maintain a safe inter-distance between consecutive cars. The most widely used vehicular micro-mobility models are the CFM ones previously mentioned. This is the major class implementing mobility patterns. The CFM adapts a following car's mobility according to a set of rules in order to avoid any contact with the leading vehicle. A general scheme that compactly

represents how CFM models work is illustrated in Fig. 2.3 [11].
There are a lot of Car Following Models available in literature already mentioned before, for which we refer to [11][12][19][21][22] and some relative sub-references.

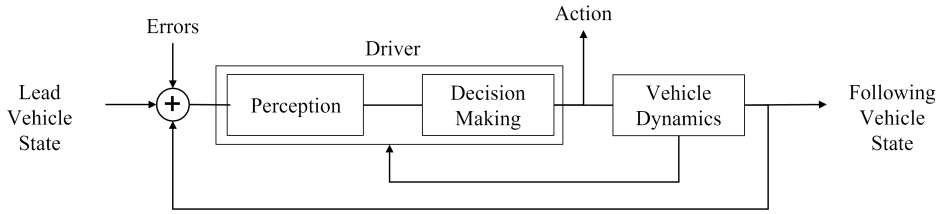


Figure 2.3. General scheme for CFM models.

- Intersection Management: It corresponds to the process of controlling an intersection and may either be modeled as a static obstacle (stop signs), a conditional obstacle (yield sign), or a time-dependent obstacle (traffic lights).

Summing up what we considered so far, SUMO [6][7][23] is the most convenient solution to generate realistic traffic simulations, because it provides multi modal mobility modeling with accurate microscopic description. It is considered as the most complete and reliable open source traffic simulator, providing realistic vehicular behavior and simulations based on realistic maps, and its mobility description reflects the reasoning that the work in [11] suggest to generate and build realistic mobility models, more than in other simulators.
SUMO is an open source, highly portable, microscopic and continuous traffic simulation package designed to handle large networks. It allows for intermodal simulation, including pedestrians and public transport, and comes with a large set of tools for realistic scenario creation.
The main features of SUMO are [23]:

- Discrete-time, continuous-space vehicle movement.
- Multi-modal mobility: a trip can be composed of different portions, with different transportation means (e.g., bus, train, and on foot) associated with each portion.
- Multi-lane roads with rules for lane changing and lane merging.
- Rules for unregulated (e.g., stop signs) and regulated (traffic light or general traffic rules) intersections.
- Driver behavior modeling.
- Personalized output generation.

To conclude Section 2.1 it is worth to understand how SUMO manages vehicles'interaction considering road infrastructure and the set of traffic rules a realistic traffic scenario should include.
Basically, SUMO uses microscopic models, with proper slight modifications to deal with road infrastructure and traffic rules. Mainly, the models implemented are CFM

Models, and an exhaustive overview of the possibilities is in [24]. The default model is a modified version of Krauss model, whose detailed description can be found in [25], and it is worth to mention that:
"The implemented model follows the same idea as that of Krauß, namely: Let vehicles drive as fast as possibly while maintaining perfect safety (always being able to avoid a collision if the leader starts braking within leader and follower maximum acceleration bounds). The implemented model has the following differences:

- *Different deceleration capabilities among the vehicles are handled without violating safety (the original model allowed for collisions in this case).*
- *The formula for safe velocity was adapted to maintain safety when using the Ballistic-position update rule. This was done by discretizing some of the continuous terms. The original model was defined for the Euler-position updated rule and would produce collisions when using Ballistic."*

In addition to this, SUMO manages in a particular way vehicles'interactions in case of lane changing/overtaking and intersections:

- In the former case, a more detailed description can be found in [26][27]. In place of the default model, the so-called "Sub-lane Model" is used to simulate continuous (not instantly) lane changing [27] with a set of parameters useful to model lane change properly [28]. The main reasoning behind this model is to further divide the lanes within the roads (called edges) in sub-lanes, and the more are the sub-lanes, the better is the approximation of lane changing and overtaking.
- In the latter case, intersection rules are governed by the "Intersection model" [29][30] which manages vehicles' interactions in approaching all the phases for a correct and safe crossing/turning in an intersection, with a proper evaluation of vehicle positions, speed, acceleration, taking into account road infrastructure where vehicles are involved and traffic rules as well.

To conclude, SUMO allows also the simulation of pedestrians using configurable pedestrian models. This can be used to model the itineraries of persons which alternate between walking and riding in vehicles. It can also be used to model the interactions between vehicles and pedestrians whenever their paths intersect in the road network. SUMO defines an abstract interface to describe the interaction between pedestrians and other simulation objects. This interface is currently implemented by three pedestrian models. More details in [31].

2.2 Vehicular Trajectory Prediction

Vehicular trajectory prediction is crucial in the context of future mm-Wave V2X communications. Indeed, V2X systems in mm-Wave bands encounter many challenges due to the physical characteristics of the communication channel and mobility.
As a matter of fact, the path loss in these frequencies lead to beamforming techniques to compensate for these detrimental effects and reduce the outage probability. Unfortunately, this introduces a higher sensitivity to blockage due to the difficulty of

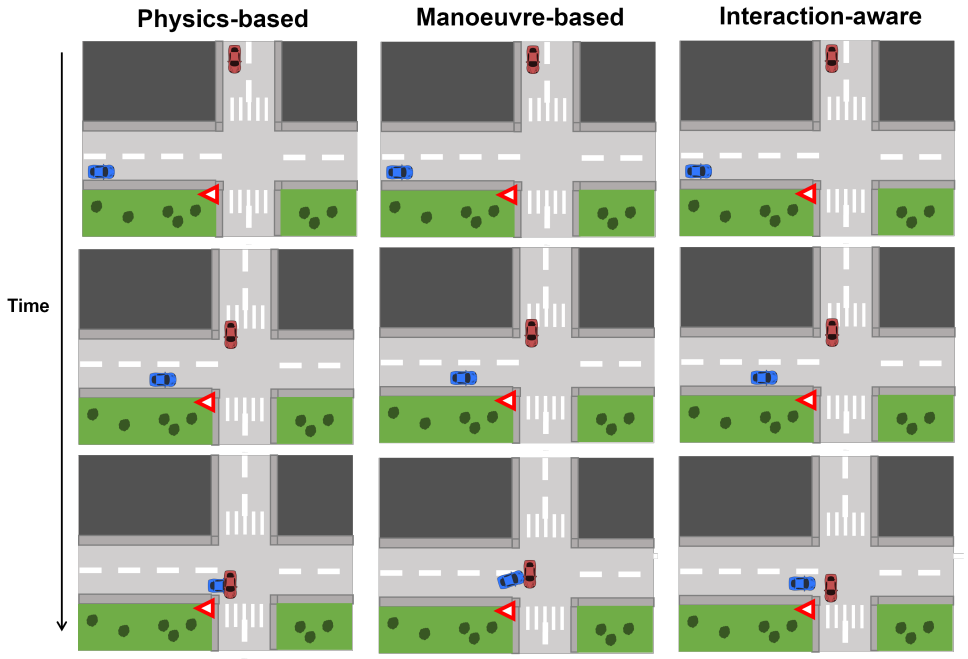
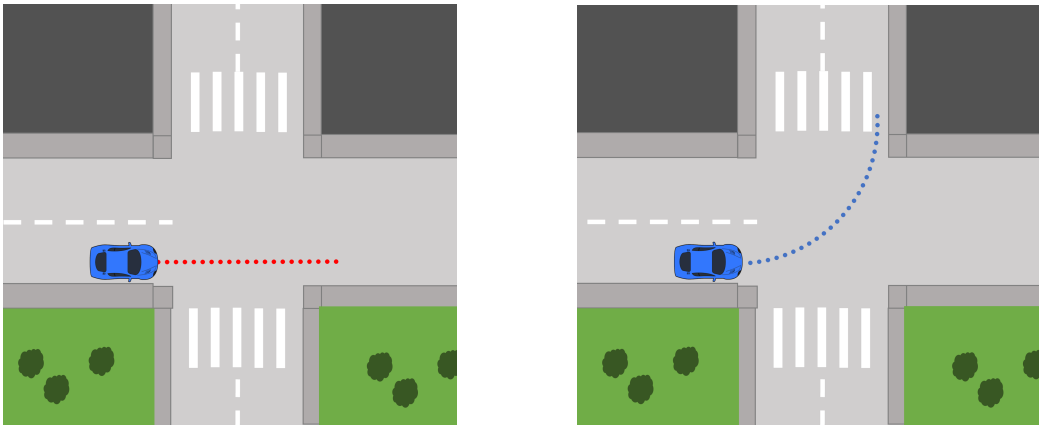


Figure 2.4. Examples of motion prediction with the different types of motion models.

predicting vehicular mobility.
So, prediction of the position vehicles will have in the future is essential to predict the evolution of the physical propagation channel and the position of static and dynamic blockers in the vehicular scenario. This knowledge allows to accurately predict link blockage and provides the CAVs the optimal conditions to proactively set up robust V2X communication links that guarantee high reliability and low latency.
The work in [32] proposes a classification of vehicular trajectory prediction methods which consists of three levels, with an increasing degree of abstraction:

- ***Physics-based*** motion models: they are the simplest models, they consider that the motion of vehicles only depends on the laws of the physics;
- ***Manoeuvre-based*** motion models: they are more advanced as they consider that the future motion of a vehicle also depends on the manoeuvre that the driver intends to perform;
- ***Interaction-aware*** motion models: these models take into account the inter-dependencies between vehicles' manoeuvres.

Fig. 2.4 illustrates the main differences between the three families of motion models with some examples. Here, the physics-based motion model assumes a constant speed and orientation for the cars, the manoeuvre-based motion model assumes that the red car goes straight and the blue car turns left, the interaction-aware motion model assumes that the red car goes straight, that the blue car turns left and that the joint motion of the cars is constrained by the traffic rules.
Physics-based motion models represent vehicles as dynamic entities governed by the laws of physics. Future motion is predicted using dynamic and kinematic models linking some control inputs (e.g. steering, acceleration), car properties (e.g. weight)



(a) CV model (b) CTRV model

Figure 2.5. CV and CTRV examples

and external conditions (e.g. friction coefficient of the road surface) to the evolution of the state of the vehicle (e.g. position, heading, speed). Extensive work has been done on such models for vehicles [33][34][35], and they remain the most commonly used for short-term prediction time window. Their complexity depends on how fine the representation of the dynamics and kinematics of a vehicle is, how uncertainties are handled, whether or not the geometry of the road is taken into account, and so for. We can distinguish between two different subgroups:

- ***Dynamic*** models: describe motion based on Lagrange's equations, taking into account the different forces that affect the motion of a vehicle, such as the longitudinal and lateral tire forces, or the road banking angle. Car-like vehicles are governed by complex physics (effect of driver actions on the engine, transmission, wheels etc.), therefore dynamic models can get extremely large and involve many internal parameters of the vehicle. Such complex models are relevant for control-oriented applications, but for applications such as trajectory prediction simpler models are preferred.

- ***Kinematic*** models: describe vehicle's motion based on the mathematical relationship between the parameters of the movement (e.g. position, velocity, acceleration), without considering the forces that affect the motion. For their simplicity they are far more popular and used than dynamic models for trajectory prediction.
  The simplest kinematic models are:

  - Constant Velocity (CV) and Constant Acceleration (CA) models, which both assume straight motion for vehicles (2.5);
  - Constant Turn Rate and Velocity (CTRV) and Constant Turn Rate and Acceleration (CTRA) models, which take into account the variation around the z-axis by introducing the yaw angle and yaw rate variables in the vehicle state vector.

The most used *Physics-based* models are the *Kinematic* ones for their simplicity. Typically, you apply one specific model to the current state of a vehicle, assuming that

this state is perfectly known and that the evolution model is a perfect representation of the motion. The advantage is its computational efficiency, which makes it suitable for applications with strong real time constraints.
Since they only rely on the low-level properties of motion, *Physics-based* motion models are limited to short-term motion prediction, in general 1 s [32]. Typically, they are unable to anticipate any change in the motion of the car caused by the execution of a particular manoeuvre or changes caused by external factors. In addition to this, they don't take into consideration the environment vehicle is involved, leading to poor prediction accuracy in the long-term.
Manoeuvre-based motion models assume that vehicle motion corresponds to a series of sequential actions executed independently from the other vehicles according to the environment itself, and we can express these actions in compact form under the term "manoeuvre". trajectory prediction with manoeuvre-based approach is based on the early recognition of the manoeuvre that drivers intend to perform. trajectories predicted through this kind of strategy are more relevant and reliable than Physics-based ones in case of long term prediction.
Following the work in [32], we can organize *manoeuvre-based* motion models available to the research community in two sets:

- ***Prototype trajectories***: the idea is that the trajectories of vehicles can be grouped into a finite set of clusters, each cluster corresponding to a typical motion pattern [36]. Motion patterns are represented using prototype trajectories which are learned from data during a training phase. Subsequently, prediction can be performed online given a partial trajectory by finding the most likely motion pattern(s) and using the prototype trajectories as a model for future motion.
  Many possibilities exist for representing a motion pattern based on the sample trajectories. One solution is to have several prototypes for each class, e.g. a subset of the training samples. However, Gaussian Process (GP) seems to be better suited for motion patterns representation. They model a process as a Gaussian distribution over a function. When applied in the context of vehicle trajectories, the assumption is that the trajectories in the learning dataset are sample functions from a Gaussian Process. Therefore, the learning consists in fitting a Gaussian distribution over these functions. The main advantages of GPs are their robustness to noise in the observed trajectories and their ability to represent the variations in the execution of a motion pattern in a consistent and probabilistic manner. 2.6 Main limitations of prototype trajectories are:

  - Need of large number of prototypes to model the motion pattern;
  - Difficulty in adapting them to different road layouts, in particular when applied to road intersections. Because each motion model is trained for a specific intersection geometry and topology, they only can be reused at intersections with a similar layout;
  - High computational cost and lack of the ability to consider physical limitations of a vehicle for GPs, leading to unrealistic sample trajectories in many cases.

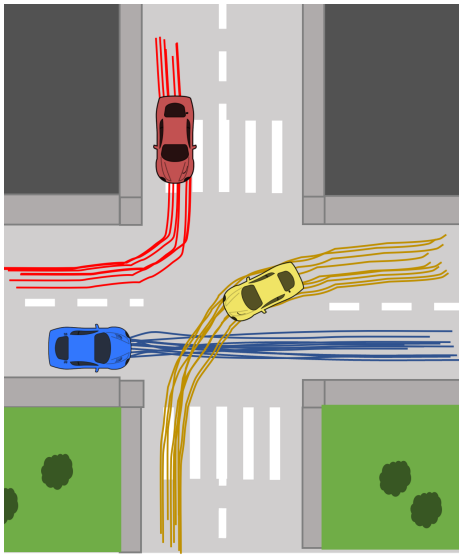


Figure 2.6. Clustered trajectories. Each cluster is represented by a color and corresponds to a typical motion pattern.

- ***Manoeuvre intention estimation***: the idea here is to first estimate the manoeuvre intention of the driver and then predict the successive physical states so that they correspond to a possible execution of the identified manoeuvre. A major advantage is that here there is no need to match the partial trajectory with a previously observed trajectory. Instead, higher-level characteristics are extracted and used to recognize manoeuvres, which makes it easier to generalize the learnt model to arbitrary layouts.
  Many cues can be exploited to estimate the manoeuvre intention of a driver, for example:
  - The current and/or past physical state of the vehicle (position, speed, heading, acceleration, yaw rate, turn signal, etc.);
  - Information about the road network (geometry and topology of the road, speed limit, traffic rules, etc.);
  - Driver behavior (head movement, driving style, etc.).

  For classifying manoeuvres, discriminative learning algorithms are very popular [32], such as Logistic regression (LR), Relevance Vector Machines (RVM) or Support Vector Machines (SVM) [37]. Another popular solution is to break down each manoeuvre into a chain of consecutive events and to represent this sequence using a Hidden Markov Model (HMM) [36]. trajectories are predicted so that they match the identified manoeuvre(s).

In practice, the assumption that vehicles move independently from each other does not hold. vehicles share the road with other vehicles, and the manoeuvres performed by one vehicle will necessarily influence the manoeuvres of the other vehicles. Inter-vehicle dependencies exist, and for example they are particularly strong at road intersections. Disregarding these dependencies might lead to erroneous interpretations of the situations, and these are the main limitation of this group of models.

Interaction-aware motion models take into account the inter-dependencies, interactions and influences between vehicles' manoeuvres, which leads to a better interpretation of their motion compared to the manoeuvre-based. There were few Interaction-aware motion models in the literature. Mostly they were based on Dynamic Bayesian Networks (DBN). However, the complexity is hardly manageable in the context of complex traffic situations and this might be a problem for real-time and low latency use cases. The Interaction-aware motion models are the most comprehensive models proposed so far in the literature. They allow longer-term predictions compared to Physics-based motion models and are more reliable than manoeuvre-based since they account for the dependencies between the vehicles. However, this exhaustiveness has some drawbacks: computing all the potential trajectories of the vehicles with these models is very complex, expensive and not compatible with real-time services. [32]
Fig. 2.7 sums up what has been discussed so far, outlining the motion models into the three families mentioned and taking into account the variables involved, the main open challenges and the tools used to implement most of them [32]. The gradual passage from objective to subjective point of view, going from the bottom of the figure to the top, follows the increasing complexity of the proposed methods, since we go through a higher degree of abstraction and we take into consideration always more symbolic quantities to be modeled and exploited, getting closer to the way of reasoning of an human being in predicting driver's behavior.

The development of Deep Learning (DL) over the past few years resulted in a

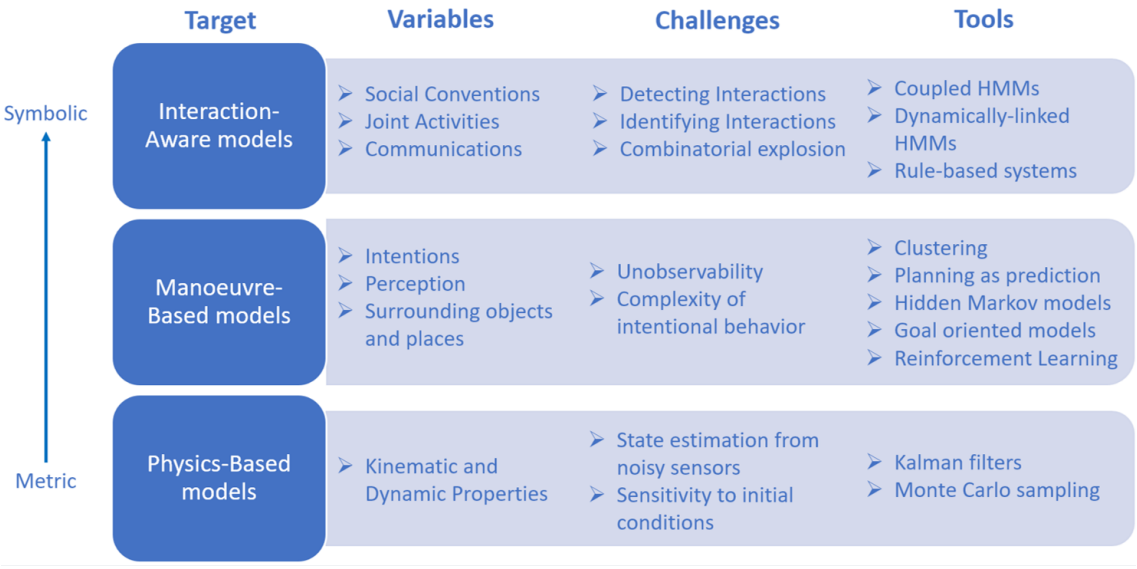


Figure 2.7. vehicular motion and trajectory prediction: modeling overview. [32]

dramatic change in many technological fields, leading to a new era also in automotive. The increasing usage of DL techniques had significant consequences in the context of vehicular trajectory prediction as well. Indeed, for vehicle behaviour prediction, DL based approaches have become popular over the recent years due to their possibility to easily embed complex features in a compact form, such as the inter-dependency among vehicles, the influence of traffic rules, the driving environment, the road topology, and

the multi-modality of vehicles behaviour, leading to promising performance in more complex and realistic environments, compared to conventional approaches. [38].
The work in [38] proposes a classification of the most recent DL based approaches in three groups, according to three different criteria: input representation, output type and prediction method.

- ***Input representation***: based on the type of input data and how it is represented, we can divide them into four sub-classes:
  - *Track History of the Target Vehicle (TV)*: The conventional approach for predicting behaviour of the TV is to use its current state (e.g. position, velocity, acceleration, heading) or track history of its states over time. For example, in [39] and [40] the track history of x-y position, speed and heading of the TV are used to predict its behaviour at different road junctions, either intersection or roundabouts. These works study the behaviour of the TV without exploiting state information about its Surrounding Vehicles (SV). Excluding the observable SV's state from the input set may result in inaccurate prediction of the TV's behaviour due to inter-dependencies of vehicles'behaviour. Indeed, although the track history of the TV has highly informative features about its short-term future motion, relying only on the TV's track history might lead to erroneous results, particularly in long-term prediction in crowded driving environments, though it allows very competitive solutions in terms of performance, computational cost and simplicity.
  - *Track History of the TV and SVs*: they consider the interaction among vehicles, explicitly feeding to the prediction model the track history of both TV and SV, which are the vehicles that can potentially impact future behavior of TV. The initial assumption for this group, and for the remaining groups where vehicle interaction is considered, is that SVs' states are available to the agent that does the estimation. If the agent is a CAV, it exploits its on-board sensors together with message exchange with other CAVs in the surrounding to estimate SVs' state. The existing studies vary in how they decide to divide the vehicles in the scene into SVs and Non Effective Vehicles (NV)s, which are the remaining vehicles in driving environment that are assumed to have no impact on the TV's behaviour, but the difficulty in finding a rule of thumb that suits the large amount of possible complex scenarios that we can encounter in a vehicular environment makes this group of solutions quite complex to implement. For example, in [41] history of states of TV and six of its closest neighbours are exploited to predict TV behaviour, while in other works chosen SVs are the three closest in TV's current lane, or the vehicles whose distance from TV is lower than a certain distance threshold. One drawback of most of these studies is that they assume that the states of all SVs are always observable, which is not a practical assumption in autonomous driving applications. A more realistic approach should always consider sensor impairments like occlusion and noise. In addition, relying only on the track history of the TV and SVs might be not sufficient for behaviour prediction, because other

factors like environment conditions and traffic rules can also modify the behaviour of vehicles.

- *Simplified Bird's Eye View*: An alternative way to consider the interaction among vehicles is by exploiting a simplified Bird's Eye View (BEV) of the environment. In this approach, static and dynamic objects, road lane, and other elements of the environment are usually depicted with a collection of polygons and lines in a BEV image. The result is a map-like image which preserves the size and location of objects (e.g. vehicles) and the road geometry. To enrich the temporal information within the BEV image, [42] uses a social tensor, known as "Social Pooling", which is a spatial grid around the target vehicle, and the occupied cells are filled with the processed temporal data (e.g., LSTM hidden state value) of the corresponding vehicles. Therefore, a social tensor contains both the temporal dynamic of vehicles and spatial inter-dependencies among them. The advantages of Simplified BEV is that it is flexible in terms of complexity of representation. Thus, it can match applications with different computational resource constraints. Second, it enables data fusion from different type of sensors into a single BEV representation. One drawback of this input representation, that applies to the previously discussed input representations as well, is that it inherits the limitations of the perception module used for estimating the states of static and dynamic objects (e.g., vehicles) in the driving environment. Therefore, an error in estimating the states, or under-representing the environment in the perception module will be cascaded to the prediction module.
- *Raw Sensor Data*: In this approach, raw sensor data is fed to the prediction model. Thus, the input data contains all available knowledge about the surrounding environment. This allows the model to learn extracting useful features from all available sensory data. Raw sensor data, compared to previous input representations, has larger dimension. Therefore, more computational resources are required to process the input data, which can make it impractical for on-board implementation in autonomous vehicles.

- ***Output type***: In this subsection, the authors classify existing studies based on how they represent a vehicle future behaviour as the output of their prediction model. We consider four sub-classes:

  - *Manoeuvre intention*: manoeuvre intention prediction is the task of estimating which manoeuvre the vehicle intends to do in upcoming time-steps. For example, in a highway scenario a vehicle can perform a lane change, while in an intersection it could be a left or right turn. To predict the intention of a vehicle approaching a T-junction, [39] defines three classes based on the destination of the vehicle, namely "east", "west", or "south". In [40], the same set of classes are used to predict the intention of a vehicle at an unsignalized roundabout. One drawback of these works is that they can only provide a high-level understanding of the vehicle behaviour. This problem can be solved by subdividing high-level manoeuvres into sub-classes that describe the behaviour more precisely. Another drawback

is the specificity of manoeuvre set to single driving environment, which can be resolved by defining a set that contains the manoeuvres in all desired driving scenarios. However, to predict a vehicle behaviour using large and in depth set of classes, a larger and more diverse training dataset is required.

- *Uni-modal trajectory*: trajectory prediction models describe the future behaviour of a vehicle by predicting series of future locations of the TV over a time window. Given a specific driving situation and history of motion for a vehicle, it might be possible for it to traverse multiple different trajectories. Uni-modal trajectory predictors are the models that only predict one of these possible trajectories.
- *Multi-modal trajectory*: They are the models that provide all the possible trajectories already mentioned in previous subclass.
- *Occupancy Map*: In these approaches, instead of predicting vehicles trajectories, the occupancy of each cell in a BEV map of the driving environment is estimated for future time steps. The drawback of such approaches is that their prediction accuracy is limited by the size of the cells in the map. Increasing the number of cells in the grid will reduce the cells' size; however, it results in higher computational costs.

- ***Prediction method***: based on the prediction model used. We can identify three classes.

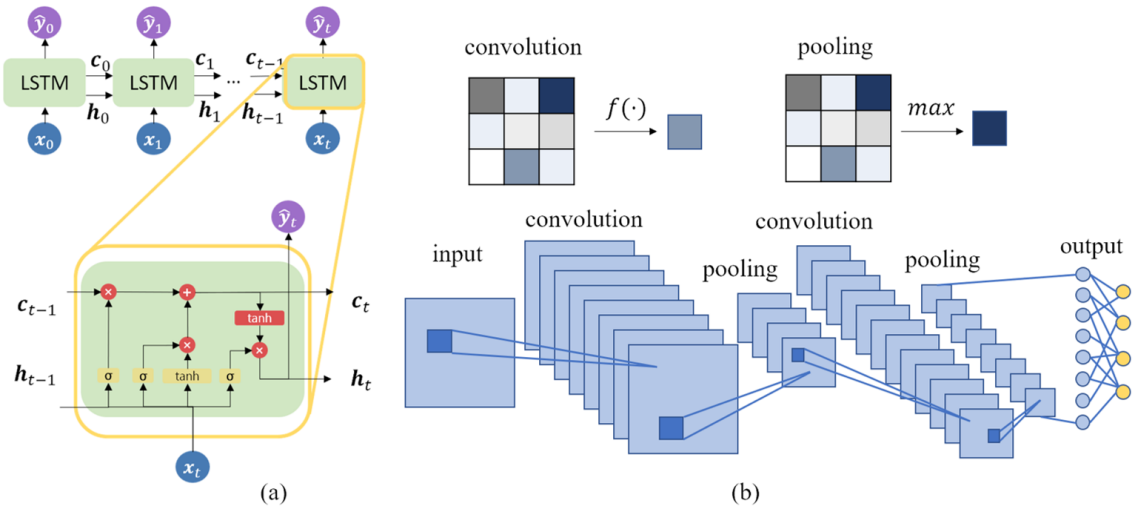


Figure 2.8. Examples of Recurrent Neural Network with Long Short-Term Memory gate cells (a) and of Convolutional Neural Network (b).

- *Recurrent Neural Network (RNN)*: The simplest RNN (e.g. Vanilla) can be considered as an extension to two-layer Fully-Connected (FC) Neural Network (NN) where the hidden layer has a feedback. This small change allows to model sequential data more efficiently. At each sequence step, the Vanilla RNN processes the input data from current step alongside the memory of past steps, which is carried in the previous hidden neurons.

However, it is difficult to train this network to learn long sequences in practice due to vanishing or exploding gradient problem, which is why gated RNNs are introduced [43]. In each cell of these networks, instead of a simple fully connected hidden layer, a gated architecture is deployed. Long Short-Term Memory (LSTM) [44] and Gated Recurrent Unit (GRU) are the most commonly used gated cells in RNNs (Fig.2.8a). In vehicle behaviour prediction, LSTMs are the most used deep models. Here, we sub-categorize recent studies based on the complexity of network architecture:

* *Single RNN*: In these models, either a single RNN is used in the simplest form of behaviour prediction (e.g., intention prediction or uni-modal trajectory prediction) or a secondary model is used alongside a single RNN to support more sophisticated features like interaction-awareness and/or multi-modal prediction. To predict the intention of vehicles, an LSTM is used in [39], [40], [41] as a sequence classifier. In this task a sequence of features is fed to successive cells of an LSTM. Then, the hidden state of the last cell in the sequence is mapped to output dimension (i.e., the number of defined classes). In [39], [40], the input is embedded using a FC layer and is fed to a three-layer LSTM; while, a two-layer LSTM without embedding is used in [41].
* *Multiple RNNs*: To deal with multi-modality and/or interaction awareness within Recurrent Neural Networks, usually an architecture of several RNNs are used in existing studies. For example, to predict multi-modal trajectories the authors in [45] use six different decoder LSTMs which correlate with six specific manoeuvres of highway driving. An encoder LSTM is applied to the past trajectory of vehicles. The hidden state of each decoder LSTM is initialized with the concatenation of the last hidden state of the encoder LSTM and a one-hot vector representing the manoeuvre specific to each decoder. The decoder LSTMs predict the parameters of manoeuvre-conditioned bivariate Gaussian distribution of future locations of the TV. Another encoder LSTM is also used to predict the probability of each of six manoeuvres.

Although RNNs are one of the main neural networks associated with data series analysis and prediction such as trajectory prediction, they have deficiency in modelling spatial relationship such as vehicles spatial interaction and image-like data such as driving scene context [38]. This explains why sophisticated solutions using RNNs usually exploit additional methods to compensate the weakness of using RNNs only.

– *Convolutional Neural Networks*: Convolutional Neural Network (CNN)s include convolution layers, where a filter with learnable weights is convolved over the input, pooling layers, which reduce the spatial size of input by subsampling, and FC layers, which map their input to desired output dimension (Fig.2.8b).
CNNs are valued in vehicle behaviour prediction for their capabilities in taking image-like data, generating image-like output, and keeping spatial relationship of the input data while processing it. These capabilities enables modelling vehicles' interaction and driving scene context and producing

occupancy map output. However, 2D CNNs lack a mechanism to model data series which is required in vehicle behaviour prediction for modelling temporal dependencies among vehicles' states over time [38].

- *Other Methods*:
  - *Fully-connected Neural Networks*: A simplistic approach for vehicle behaviour modelling is to rely only on the current state of the vehicles, which might be inevitable due to unavailability of states history of vehicles or first-order Markov assumption. In this case, the input data is not a sequence and any Feed-Forward (FF) NNs (e.g. FC) can be used instead of RNNs. In some driving scenarios, FF NNs can have competitive results with faster processing time compared to RNNs.
  - *Combination of RNNs and CNNs*: In some existing works, RNNs are used because of their temporal feature extracting power, and CNNs are used for their spatial feature extracting ability. This inspires some researchers to use both in their models to process both the temporal and spatial dimensions of the data. As an example, the work in [42], which is a sequel of [45], uses one encoder-LSTM per vehicle to extract the temporal dynamics of the vehicle. The internal states of these LSTMs form a social tensor which is fed to a CNN to learn the spatial inter-dependencies. Finally, six decoder LSTMs are used to produce the manoeuvre-conditioned distribution of the future trajectory of the TV.

  - *Graph Neural Networks*: The vehicles in a driving scenario and their interaction can be considered as a graph in which the nodes are the vehicles and the edges represent the interaction among them. Using this representation, Graph Neural Networks (GNN) can be used to predict TV's behaviour. One drawback of current graph-based approach is that static scene context is usually neglected in the modelling procedure.

2.3 Pedestrian Trajectory Prediction

For the pedestrian case, trajectory prediction becomes more challenging. Indeed, in a vehicular environment, predicting how a scene involving human agents will unfold over time is essential to better predict the vehicles' behavior.
Human motion prediction has received increased attention in recent years across several communities. The challenge of making accurate predictions of human motion arises from the complexity of human behavior and the variety of its internal and external stimuli.
As a matter of fact, Human motion behavior may be driven by many factors, such as the own goal intent, the presence and actions of surrounding agents, social rules and norms, social relations between agents, or the environment with its topology, geometry and semantics.
Most factors are not directly observable and need to be inferred from noisy perceptual cues, or modeled from context information. Furthermore, to be effective in practice,

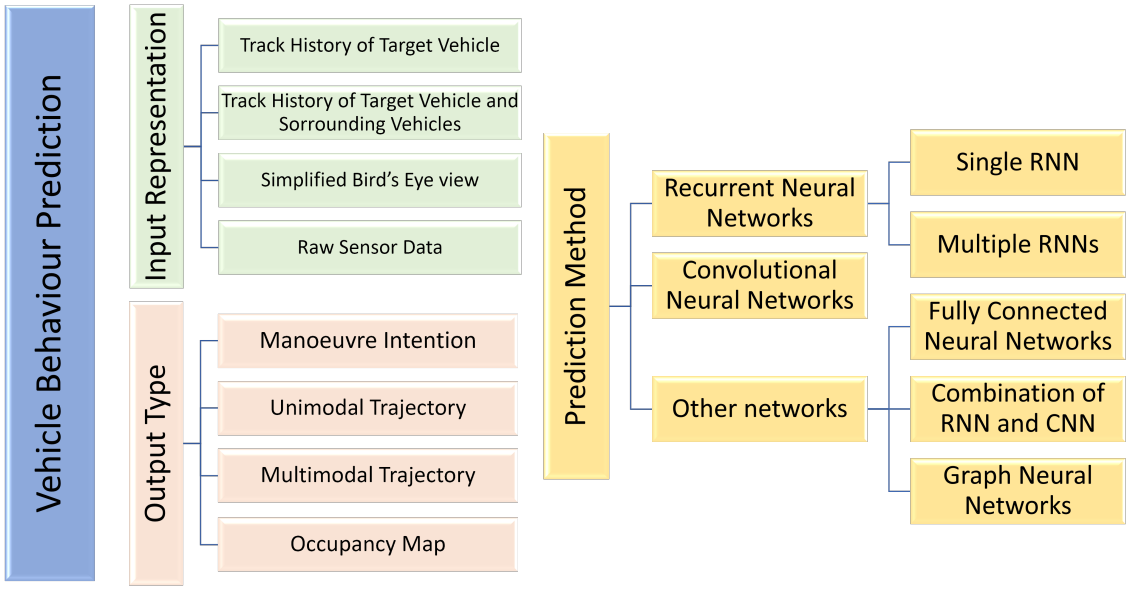


Figure 2.9. Proposed classification of SoA DL approaches for vehicle behaviour and trajectory prediction ([38]).

motion prediction should be robust and operate in real-time. Human motion comes in many forms: articulated full body motion, gestures and facial expressions, or movement through space by walking.
The work in [46] proposes a classification of the methods based in two different groups: ***motion modeling*** and ***contextual cues*** (Fig. 2.10).

- ***Motion modeling*** category subdivides the prediction approaches based either on how they represent human motion and formulate the causes thereof, and on the level of cognition involved in the prediction process (Fig. 2.10 on the left). Similarly to the vehicular case, here we can identify three sub-groups:

  1. ***Physics-based*** methods follow a reactive sense-predict scheme and define an explicit dynamical Model-based on Newton's law, so motion is predicted by forward simulating a set of explicitly defined dynamics equations that follow a physics inspired model. based on the complexity of the model, we recognize the following sub-classes:
     - *Single-model methods*: define a single dynamical motion model;
     - *Multi-model methods*: include a fixed or online adaptive set of multiple dynamics models and a mechanism to fuse or select the individual models.

     Similarly to vehicular case, also for human motion prediction *Kinematic* models are the simplest and most common *Physics-based* models, representing motion states as position, orientation, velocity and acceleration. Popular examples include the CV model, the CA model and the Coordinated Turn (CT) model. A large number of works across application domains rely on *kinematic* models for their simplicity.
     A number of approaches extend physics-based models to account for information from a map. For example [47] predicts pedestrian motion along

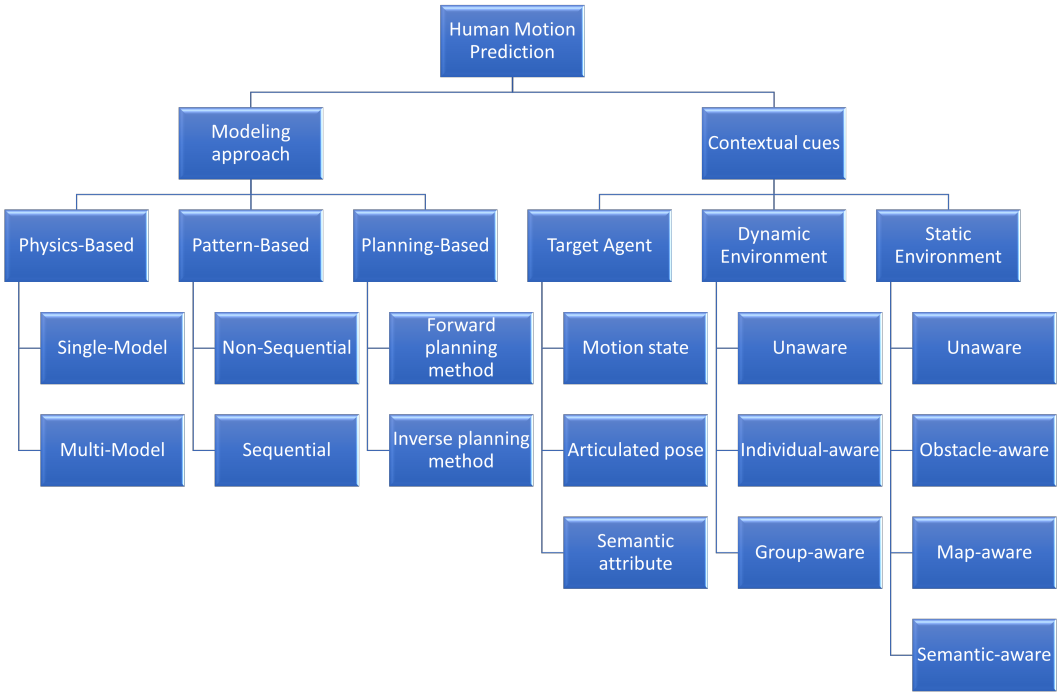


Figure 2.10. Overview of the taxonomy proposed in [46]

a graph with straight line edges centered on side- and crosswalks, with long-term predictions up to 10 seconds ahead.
Those works, however, consider only a single target agent, neglecting local interactions between multiple agents. There are also methods that add social situation awareness, predicting several target agents jointly. One example is the social force model in [48], that superimposes attractive forces from a goal with repulsive forces from other agents and obstacles, while [49] presents a model that embeds social relationships in the linear combination of predefined basic social effects (attraction, repulsion and non-interaction). The motion predictor maintains several hypotheses over the social modes, in which the pedestrians are involved.
Complex agent motion is poorly described by a single dynamical model. A common approach is the definition and fusion of different prototypical motion modes. To this end there are Multi-model (MM) methods, which generally consist of a fixed or on-line adaptive model set, a strategy to deal with the uncertainties and a mechanism to generate the overall best estimate from a fusion or selection of the individual filters. They are used to represent more complex motion, to incorporate context information from other agents and context information from the map.
The Interactive Multiple Model (IMM) is a widely used inference technique with numerous applications in tracking and predictions. An alternative approach is DBN.

2. ***Pattern-based*** methods: learn motion patterns from data of observed agent trajectories, following a sense-learn-predict scheme. They approximate arbitrary dynamics function from training data by discovering statis-

tical behavioral patterns in the observed motion trajectories.
They are separated into two categories:

- *Sequential methods*: learn conditional models over time and recursively apply learned transition functions for inference;
- *Non-sequential methods*: directly model the distribution over full trajectories without temporal factorization of the dynamics.

Learning local motion patterns, such as probabilities of transitions between cells on a grid-map, is a simple, commonly used technique for making sequential predictions. These can be extended, accounting for further future transitions and describing where the person might be in the future. Unlike the local transition patterns, which are learned and applied for prediction only in a particular environment, location independent patterns are used for predicting transitions of an agent in the general free space. Several recent sequential methods use neural networks for time series prediction, i.e. assuming higher order Markov property. Such time series-based models are making a natural transition between the first order Markovian methods (e.g. local transition patterns) and non-sequential techniques (e.g. clustering-based).
LSTM networks are becoming a popular modeling approach for predicting human motion. [50] propose a Social-LSTM which learns to predict joint location-independent transitions in continuous spaces. Each human is modeled by an individual LSTM. Since humans are influenced by nearby people, LSTM are connected in the social pooling system. The work in [51] extends the Social-LSTM proposed in [50], explicitly modeling human-space interactions by defining a "context-aware" pooling layer.

3. ***Planning-based*** methods: explicitly reason about the agent's long-term motion goals and compute policies or path hypotheses that enable an agent to reach those goals, following a sense-reason-predict scheme. Most of the methods use objective functions that minimizes some notion of the total cost of a sequence of actions (motions), and not just the cost of one action in isolation.
We classify the planning-based approaches into two categories:
   - *Forward planning methods*: make an explicit assumption regarding the optimality criteria of an agent's motion, using a pre-defined reward function;
   - *Inverse planning methods*: estimate the reward function or action model from observed trajectories using statistical learning techniques.

- ***Contextual cues*** category classifies the methods according to all relevant internal and external stimuli that influence motion behavior. Cues categorization is based on their relation to the target agent, other agents in the scene and properties of the static environment:

1. Cues of the ***target agent*** include:
   - *Motion state* (position and possibly velocity);

- *Articulated pose* (such as head orientation, which could be quite useful). Indeed, if an autonomous vehicle is capable, by means of sensors such as cameras, to infer pose properties of a full pedestrian body, understanding that it is oriented in the opposite way with respect to a crossroad in an urban scenario, then the autonomous vehicle can predict that the pedestrian is more likely not to pass through the crossroad, avoiding stopping (at least slowing down) because it is not necessary, leading to an overall very efficient system;
- *Semantic attributes* such as the age, gender and personality.

2. With respect to the ***dynamic environment*** we distinguish:
   - *Unaware* methods, which compute motion predictions for the target agent not considering the presence of other agents;
   - *Individual-aware* methods, which account for the presence of other agents;
   - *Group-aware* methods, which account for the presence of other agents as well as social grouping information. This allows to consider agents in groups, formations or convoys that move differently than independent agents.

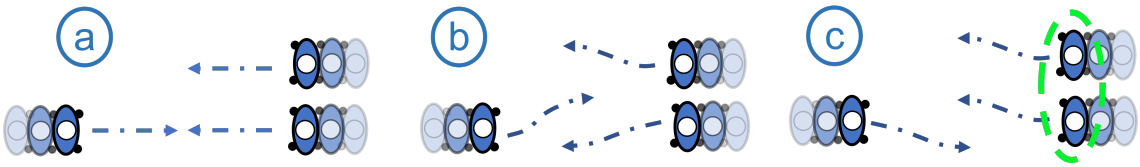


Figure 2.11. Dynamic environment cues: (a) unaware, (b) individual-aware, (c) group-aware (accounting for social grouping cues, in green) [46].

The difference in the way algorithms related to the dynamic cues predict the trajectories is evident. Consider in Fig. 2.11 the leftmost pedestrian for each case and assume that its goal is to go where the dashed arrow is directed. The same reasoning for the other pedestrians is trivial. So this pedestrian is predicted to go from the left to the right, but what is the predicted path?
It depends on the algorithm and the information exploited:

- In case (a) the algorithm is *unaware*, so it predicts the trajectory without assuming the existence of dynamic agents in the scenario, leading to a straight line. In practice, the other pedestrians exist, so the method leads to poor prediction, if not completely wrong;
- In case (b) the algorithm is *individual-aware*, so it is allowed to exploit information about the presence of the other pedestrian. This leads to an improvement in the prediction, since the arrow becomes curved, predicting that the pedestrian is very likely to avoid the other two in order to go to the right;
- Finally, if the algorithm is *group aware*, accounting for social awareness (c) then it can understand that the two pedestrians form a group, so

from a subjective point of view it is a unique entity, and predicts what an human being would do intuitively, trying not to break the couple.

3. With respect to the *static environment* we distinguish:
   - *Unaware* methods, which assume an open space environment.
   - *Obstacle-aware* methods, which account for the presence of unmodeled static obstacles not in the map.
   - *Map-aware* methods, which account for environment geometry and topology.
   - *Semantics-aware* methods, which additionally account for environment semantics or affordances such as no-go-zones, crosswalks, sidewalks, or traffic lights.

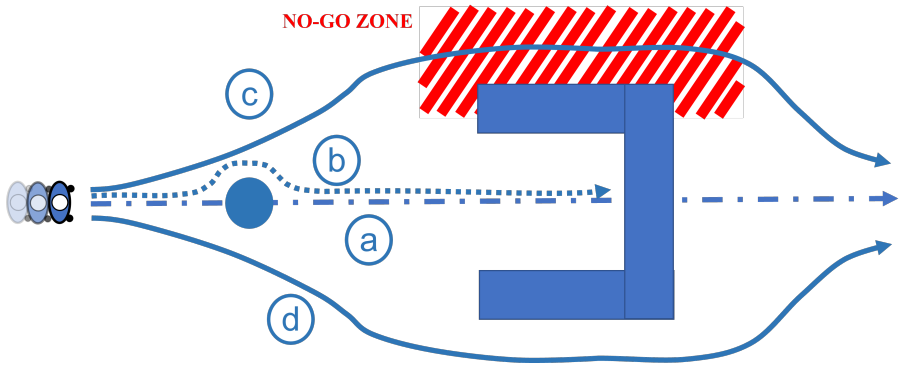


Figure 2.12. Static environment cues: (a) unaware (ignoring any static objects, dashed line), (b) obstacle-aware (accounting for unmodeled obstacles, dotted line), (c) map-aware (accounting for a topometric environment model avoiding local minima, solid line), (d) semantics-aware (solid line) [46].

The difference between these algorithms is very clear by looking at Fig. 2.12. In this figure we suppose to predict the pedestrian goes from the left to the right, but which is the predicted trajectory? If the algorithm ignores any obstacle, the prediction is a straight line, because the prediction is done assuming that no obstacle is present (a). If we use an algorithm that exploits knowledge about the presence of some obstacles, then it can understand that, in order to go from the left to the right, the pedestrian has to avoid the black circular obstacle (b). But since in the real scenario a building is present, this prediction would be wrong. Knowledge of the static contextual cues, so of the overall map of the scenario, improves the prediction (c), predicting that the pedestrian has to avoid the building to go to the right (he can't go through it, for sure). The best prediction possible comes from the knowledge also of the traffic rules within the context, so the semantics of the scenario, which leads to choose the bottom trajectory in place of the top one to avoid the building, since on the upper part there is a no-go zone (d).

Fig. 2.13 sums up the advantages, drawbacks and future direction of research of the investigated methods.

Motion models	Advantages	Drawbacks	Future direction of research
Physics-Based	• Simple and computationally fast • Ok where explicit transition functions are enough to represent motion dynamics and/or contextual cues • No need of training data	• Too simple to describe real world complex scenario • Lack of context info • Ok for short prediction horizon and relatively obstacle-free environment only	• Go beyond KF solutions, including ML and high order HMM to estimate context dependent patterns, handle complex environment and type of motions • Reason on agent intentions and actions, both single and in groups, with AI and game theory solutions
Pattern-Based	• Ok for environments with complex unknown dynamics and areas with rich semantics • Long prediction horizon	• Need of training data • Context and topology dependent • Ok for non safety critical applications	• Extension by more goal or intention driven predictions • Automatic goal inference based on the semantics of the environment
Planning-Based	• Long prediction horizon • They can be generalized to different environments	• Agents' goals explicitly defined • Map availability	• Improvement based on goal estimate on-the-fly in unknown environments

Figure 2.13. Final overview of advantages, drawbacks and future direction of research of the human motion and trajectory prediction group methods investigated in [46].

Chapter 3

Map-Assisted Vehicular Trajectory Prediction System

This Chapter introduces and describes the proposed strategy for the implementation of map-assisted VTP system to support future 6G mm-Wave V2X communications. Specifically:

- Section 3.1 describes the proposed strategy for vehicular trajectory prediction;
- Sections 3.2 and 3.3 illustrates the implemented algorithms for manoeuvre and trajectory prediction building blocks, respectively.

3.1 Proposed Strategy

The detailed investigation provided in Section 2.2 about vehicular trajectory prediction methods available in literature suggests that a solution which tries to reach a good compromise between accuracy and complexity is preferred. Indeed, on the one hand, it is crucial to implement solutions that provide high-accurate trajectory prediction in the long term in order to meet the stringent requirements needed to support URLLC vehicular services. On the other hand, we need to search also for straightforward solutions that can be easily integrated into devices with low energy and computational capabilities to provide fast outcomes at a low computational cost.
The overview showed that the most promising solutions, in terms of performance, are those able to exploit information about vehicles' interaction and inter-dependencies, traffic rules, and road topology. Indeed, knowledge about these aspects of the traffic scenario could significantly help reduce the complexity of the problem and improve the prediction accuracy at the same time.
So, a solution that can exploit information from vehicle relations, digital maps, and real-time traffic has great potential.
Fig.3.2 compactly shows the main idea for the implementation of the prediction system. We will go into the details of the proposed strategy, following the flux diagram in Fig.3.2. The related scenario is illustrated in Fig.3.1.

We initially assume that at a specific time instant t_0, we have some CAVs and non-CAVs in a traffic scenario. The prediction system knows all vehicles' current

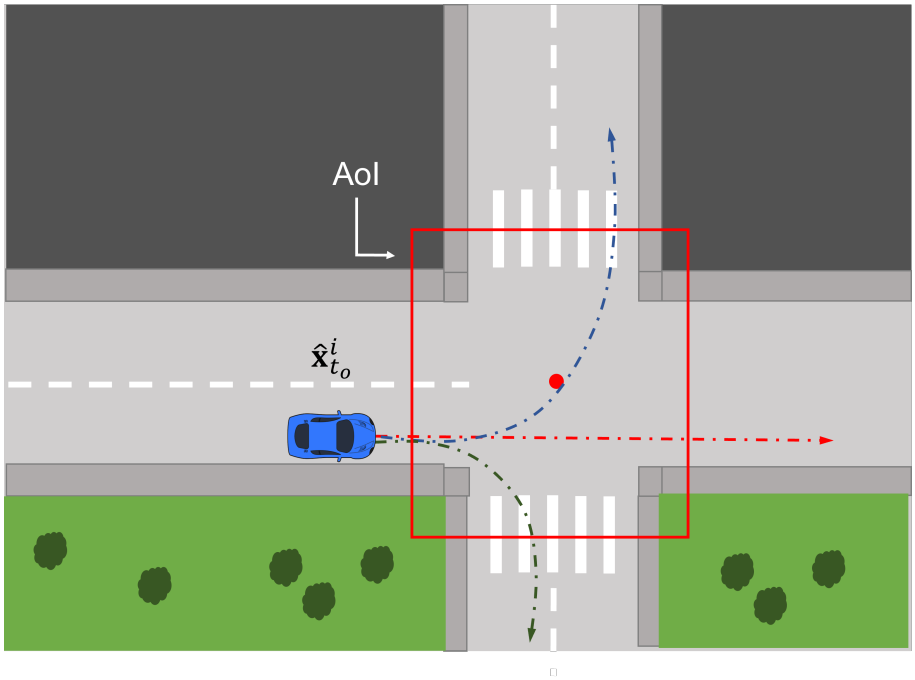


Figure 3.1. Sketch of a non-CAV approaching an intersection, with the three possible manoeuvres (go left, go right, go straight) depicted in blue, red, green, respectively.

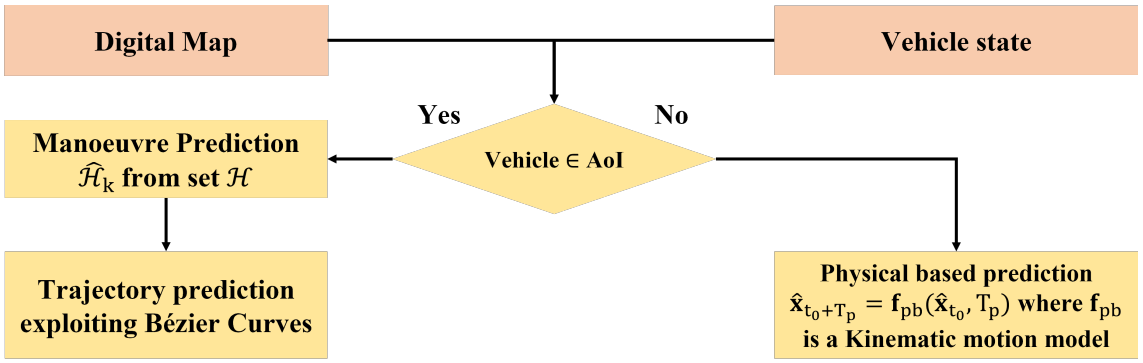


Figure 3.2. Proposed Strategy scheme for map-assisted VTP system.

and past state (position and vehicular dynamics), simulating that CAVs have already estimated and exchanged information about their current state, together with their accuracies, through a CAS procedure.
Given these assumptions, the goal is to estimate trajectories of all non-CAVs involved in the scenario up to a prediction window T_p.

For better clarity, in Fig.3.1, we report only one vehicle representing a non-CAVs agent since the reasoning is the same for all the others. We will refer to this non-CAV as TV since it is the vehicle we want to predict the trajectory.
So, given the general, non-CAVs, i-th TV, we assume to know an estimate of its state vector at time t_0, $\hat{\mathbf{x}}^i_{t_0}$, related to the true state $\mathbf{x}^i_{t_0}$ through an additive noise model that we define in 3.1, on the left. The true state vector consists of position (x and y coordinate in Universal Transverse of Mercator (UTM) reference system), heading, speed modulus, acceleration along the direction provided by the heading, and yaw

rate, which describes the heading variations in time.
Vector $\mathbf{n^i_{t_0}}$ is the noise term that represents the impairments that affect the previous state estimation procedure, and we model it as a random vector with a certain mean $\boldsymbol{\mu_n}$ and covariance matrix $\boldsymbol{\Sigma_n}$. This definition is valid in general since we are not making any other specific assumption on noise statistical properties.
Eq.3.1 shows the definition of the state and random noise vectors on the top right and the bottom right, respectively.

$$\hat{\mathbf{x}}^{\mathrm{i}}_{\mathrm{t}_0} = \mathbf{x}^{\mathrm{i}}_{\mathrm{t}_0} + \mathbf{n^i_{t_0}} \qquad \begin{cases} \mathbf{x}^{\mathrm{i}}_{\mathrm{t}_0} = \left[\mathrm{x}^{\mathrm{i}}_{\mathrm{t}_0}, \mathrm{y}^{\mathrm{i}}_{\mathrm{t}_0}, \vartheta^{\mathrm{i}}_{\mathrm{t}_0}, \mathrm{v}^{\mathrm{i}}_{\mathrm{t}_0}, \mathrm{a}^{\mathrm{i}}_{\mathrm{t}_0}, \omega^{\mathrm{i}}_{\mathrm{t}_0}\right] \\ \mathbf{n^i_{t_0}} \quad \text{r.v.} : \begin{cases} \boldsymbol{\mu_n} = \mathbb{E}(\mathbf{n^i_{t_0}}) \\ \boldsymbol{\Sigma_n} = \mathrm{cov}(\mathbf{n^i_{t_0}}) \end{cases} \end{cases} \tag{3.1}$$

In addition to the current vehicle state, the system takes input information collected from the digital maps. They may include knowledge about buildings, foliage, streets, and street rules/relations in the target vehicle neighborhood. This information will be precious for our system because we can know not only the physical position of the vehicle but also the street it belongs to, vehicle direction in the street, the future crossing area, which we call Area of Interest (AoI) from now on, and, according to traffic rules, the street destinations beyond the AoI.
So, given the information coming from the vehicle state and the digital map, the algorithm checks if the vehicle belongs to its future AoI or not. Before going on, it is worth better explain what AoI is, what it represents and why it is crucial for our trajectory prediction system.
For us, an AoI is an area in the vehicular network where many streets intersect, so a crossing area. Typically, in these areas, a vehicle has a higher degree of freedom in terms of mobility since there are many future destinations it can reach, which allows it to perform one from many non-trivial manoeuvres according to the desired destination. An example of AoI is an intersection, sketched in Fig.3.2, but it can also be a Roundabout Junction.
We introduce the AoI concept because, from a logical point of view, we can describe the vehicular network simply by identifying non-AoI and AoI areas, whatever their topology is. In the former, network topology limits mobility, in the latter, mobility is much complex, and uncertainty on trajectory prediction is higher due to many destinations available.
For example, a straight or curved road portion with only one destination is not an AoI from our point of view since the vehicle cannot do anything different than following street direction.
So, in non-AoI areas, trajectory prediction is trivial. Indeed, a prediction strategy that exploits simple, cheap, and fast algorithms might be sufficient (i.e., Physics-based prediction with Kinematic motion models is possible). In AoI areas, this strategy does not suit anymore since it cannot capture the complex reasoning behind the driver's choice to perform a specific manoeuvre, in place of possible other manoeuvres available, to cross AoI to reach a specific desired destination. This choice depends on the unavoidable interaction he has with the other drivers and the complex environment where he is involved.

So, a method that tries to catch this reasoning behind the future manoeuvre is vital. Our strategy goes in this direction because it proposes to follow a simple Physical based approach if the vehicle is still out of its future AoI and a map-assisted manoeuvre-based approach if the vehicle belongs to the AoI instead. In the latter case, the strategy combines the digital maps and Bézier curves. In this way, we can reach a good compromise between accuracy and complexity.
We summarize the proposed method in the flux diagram reported in Fig.3.2. The discriminant used to evaluate the vehicle inside or outside AoI is the distance from the AoI center, which we call Point of Interest (PoI).
To sort things out, in case vehicle belongs to AoI the algorithm:

1. exploits information from current vehicle state and the digital maps to identify the kind of AoI vehicle is belonging to (i.e: intersection);
2. defines a set of possible manoeuvres $\mathcal{H}$ according to specific AoI and street relations (i.e: for a 4-way unsignalized intersection (Fig.3.2) $\mathcal{H}$: {Go straight, Go Right, Go Left});
3. predicts the manoeuvre $\hat{\mathcal{H}}_{\mathrm{k}}$ from the set $\mathcal{H}$;
4. given the predicted manoeuvre $\hat{\mathcal{H}}_{\mathrm{k}}$, predicts the future trajectory.

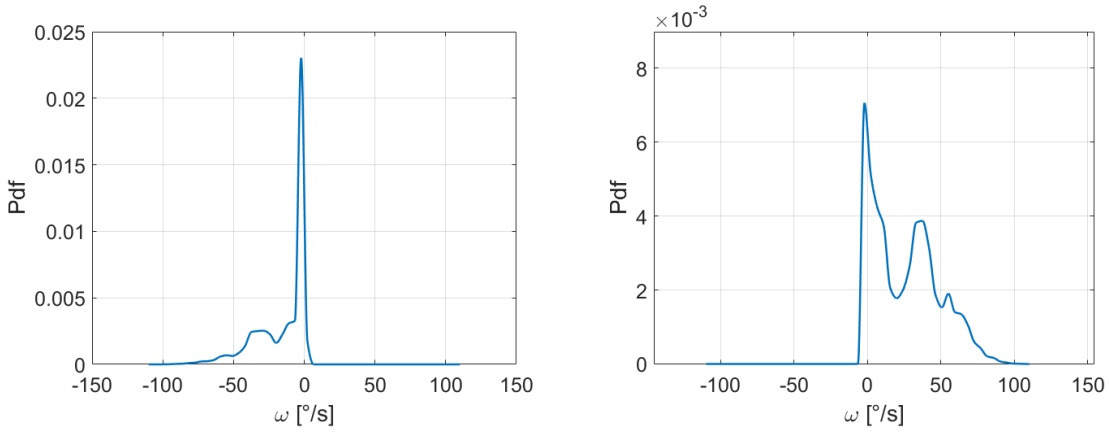
We will discuss in detail the strategies implemented for manoeuvre and trajectory prediction phases in Sections 3.2 and 3.3, respectively.

3.2 Manoeuvre Prediction

Regarding manoeuvre prediction, two strategies are implemented and compared: a model-based approach and a Machine Learning Based approach. In the former, conditional Probability Density Function (pdf) of the vehicle state variables are learned from the scenario in the AoI, and estimated manoeuvre $\hat{\mathcal{H}}_{\mathrm{k}}$ is the one that maximizes, with respect to the entire set $\mathcal{H}$, a function of these pdfs. In the latter, we develop a neural network that inputs TV history track to provide estimated manoeuvre as output.

3.2.1 Model-Based approach

For the model-based approach, we divide the AoI in different circular regions. So, we define a set of regions $\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, ..., \mathcal{R}_{\mathrm{J}}\}$, where J is the total number of regions. Given that $\mathcal{R}_{\mathrm{j}}$ is the general j-th region in the set $\mathcal{R}$, a vehicle is in region $\mathcal{R}_{\mathrm{j}}$ if, given d the vehicle distance from the PoI, it happens that $\mathrm{d}_{\mathrm{min}_{\mathcal{R}_{\mathrm{j}}}} \leq \mathrm{d} \leq \mathrm{d}_{\mathrm{max}_{\mathcal{R}_{\mathrm{j}}}}$, where $\mathrm{d}_{\mathrm{min}_{\mathcal{R}_{\mathrm{j}}}}$ and $\mathrm{d}_{\mathrm{max}_{\mathcal{R}_{\mathrm{j}}}}$ are the minimum and maximum distances from PoI to evaluate the vehicle inside region $\mathcal{R}_{\mathrm{j}}$, respectively.
Given these first assumptions, according to the specific AoI, we also define a set of manoeuvres $\mathcal{H} = \{\mathcal{H}_1, \mathcal{H}_2, ..., \mathcal{H}_{\mathrm{N}}\}$, where N is the total number of possible manoeuvres. Given that $\mathcal{H}_{\mathrm{k}}$ is the general k-th manoeuvre in the set $\mathcal{H}$, each manoeuvre $\mathcal{H}_{\mathrm{k}}$ is characterized by the a-priori probability $\mathrm{p}(\mathcal{H}_{\mathrm{k}})$. The set $\mathcal{R}$ together



(a) Left turn (b) Right turn

Figure 3.3. Pdf of ω for left turn and right turn, near the intersection PoI.

with the distance intervals and the set $\mathcal{H}$ are defined according to the specific AoI. For each region $\mathcal{R}_j$, and for each manoeuvre $\mathcal{H}_k$, the state parameters acceleration (a), speed (v), and yaw rate (ω) are collected to build and extract conditional pdfs. Specifically, we extract:

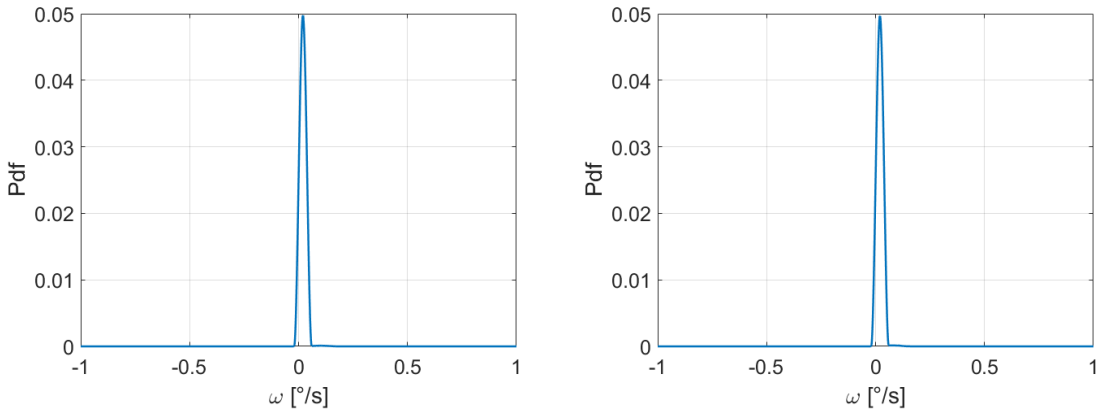
$$\begin{aligned} &p(\omega|\mathcal{H}_k, \mathcal{R}_j) \\ &p(a|\mathcal{H}_k, \mathcal{R}_j) \qquad\qquad k = 1, 2, ..., N \\ &p(v|\mathcal{H}_k, \mathcal{R}_j) \qquad\qquad j = 1, 2, ..., J \\ &p(\omega, a, v|\mathcal{H}_k, \mathcal{R}_j) \end{aligned}$$

We propose three model-based manoeuvre prediction methods that use the aforementioned conditional pdfs. The assumptions are the same described in Section 3.1 and reported in Fig.3.2, so we know an estimate of TV state vector at time t_0, $\hat{\mathbf{x}}^i_{t_0}$, and the estimated region $\hat{\mathcal{R}}_j$ the vehicle belongs to, according to estimated distance from PoI.
The three methods are:

1. **Non-Weighted Maximum A-Posteriori (MAP)**: Given $\hat{\mathbf{x}}^i_{t_0}$ and $\hat{\mathcal{R}}_j$, the estimated manoeuvre $\hat{\mathcal{H}}_k$ is the one that maximizes the non-weighted A-Posteriori mean $\forall \mathcal{H}_k \in \mathcal{H}$

$$\hat{\mathcal{H}}_k = \max_{\mathcal{H}} \left(\frac{p(\mathcal{H}_k|\hat{\omega}^i_{t_o}, \hat{\mathcal{R}}_j) + p(\mathcal{H}_k|\hat{a}^i_{t_o}, \hat{\mathcal{R}}_j) + p(\mathcal{H}_k|\hat{v}^i_{t_o}, \hat{\mathcal{R}}_j)}{3} \right) \tag{3.2}$$

   Fig.3.3 shows two examples of conditional pdfs. They are obtained from a simulation we performed in an intersection of Milan. Specifically, Fig.3.3a and Fig.3.3b show the pdf of ω when the actual manoeuvres are left turn and right turn, respectively, and the pdfs are learned in a region where the maximum distance from the intersection PoI is 10 m.



(a) Left turn (b) Right turn

Figure 3.4. Pdf of ω for left turn and right turn, far from the intersection PoI.

2. **Weighted Maximum Likelihood**: Given $\hat{\mathbf{x}}^{\mathrm{i}}_{\mathrm{t}_0}$ and $\hat{\mathcal{R}}_{\mathrm{j}}$, the estimated manoeuvre $\hat{\mathcal{H}}_{\mathrm{k}}$ is the one that maximizes the weighted sum of the likelihood functions $\forall \mathcal{H}_{\mathrm{k}} \in \mathcal{H}$

$$\hat{\mathcal{H}}_{\mathrm{k}} = \max_{\mathcal{H}} \Big(\mathrm{w}_{\omega,\mathcal{R}_{\mathrm{j}}} \mathrm{p}(\hat{\omega}^{\mathrm{i}}_{\mathrm{t_o}} | \mathcal{H}_{\mathrm{k}}, \hat{\mathcal{R}}_{\mathrm{j}}) + \mathrm{w}_{\mathrm{a},\mathcal{R}_{\mathrm{j}}} \mathrm{p}(\hat{\mathrm{a}}^{\mathrm{i}}_{\mathrm{t_o}} | \mathcal{H}_{\mathrm{k}}, \hat{\mathcal{R}}_{\mathrm{j}}) + \mathrm{w}_{\mathrm{v},\mathcal{R}_{\mathrm{j}}} \mathrm{p}(\hat{\mathrm{v}}^{\mathrm{i}}_{\mathrm{t_o}} | \mathcal{H}_{\mathrm{k}}, \hat{\mathcal{R}}_{\mathrm{j}}) \Big) \tag{3.3}$$

where weighting coefficients $\mathrm{w}_{\omega,\mathcal{R}_{\mathrm{j}}}$, $\mathrm{w}_{\mathrm{a},\mathcal{R}_{\mathrm{j}}}$, $\mathrm{w}_{\mathrm{v},\mathcal{R}_{\mathrm{j}}}$ are normalized to 1 and based on Jensen-Shannon Divergence (JSD), which is a method that provides a finite distance measure between multiple probability distributions [52][53]. For example, if we consider the yaw rate, we define the associated weighting coefficient as

$$\mathrm{w}_{\omega,\mathcal{R}_{\mathrm{j}}} = \frac{\mathrm{JSD}_{\omega,\mathcal{R}_{\mathrm{j}}}}{\mathrm{JSD}_{\omega,\mathcal{R}_{\mathrm{j}}} + \mathrm{JSD}_{\mathrm{a},\mathcal{R}_{\mathrm{j}}} + \mathrm{JSD}_{\mathrm{v},\mathcal{R}_{\mathrm{j}}}} \tag{3.4}$$

where $\mathrm{JSD}_{\omega,\mathcal{R}_{\mathrm{j}}}$, $\mathrm{JSD}_{\mathrm{a},\mathcal{R}_{\mathrm{j}}}$ and $\mathrm{JSD}_{\mathrm{v},\mathcal{R}_{\mathrm{j}}}$ are the JSD Divergences related to ω, a and v respectively.
The goal with JSD is to weigh, in the maximization procedure, the contribution that a pdf of a state parameter conditioned by specific manoeuvre and estimated region gives in the sum, according to the amount of information all the pdfs related to that specific state parameter and estimated region provide for each manoeuvre.
Fig.3.4 shows two examples of conditional pdfs obtained from the same simulation we mentioned before. Specifically, Fig.3.4a and Fig.3.4b show the pdf of ω when the actual manoeuvres are left turn and right turn, respectively, and the pdfs are learned in a region which is far from the intersection PoI. From the plots we see that, whatever the manoeuvre of the vehicle is, the two pdfs are very similar. This result is something we expect because the vehicles do not make turns far from the intersection PoI since it is too early yet. So, in this case the yaw rate does not provide me information to distinguish between the possible

manoeuvres. Then, its contribution should be weighted, and JSD Divergence is the proper tool for this operation.
The JSD related to ω and region $\mathcal{R}_j$ is defined as

$$\mathrm{JSD}_{\omega,\mathcal{R}_j} = \mathrm{H}\left(\sum_{k=1}^{N} \mathrm{P}(\mathcal{H}_k)\mathrm{p}(\omega|\mathcal{H}_k,\mathcal{R}_j)\right) - \sum_{k=1}^{N} \mathrm{P}(\mathcal{H}_k)\mathrm{H}(\mathrm{p}(\omega|\mathcal{H}_k,\mathcal{R}_j)) \quad (3.5)$$

where:

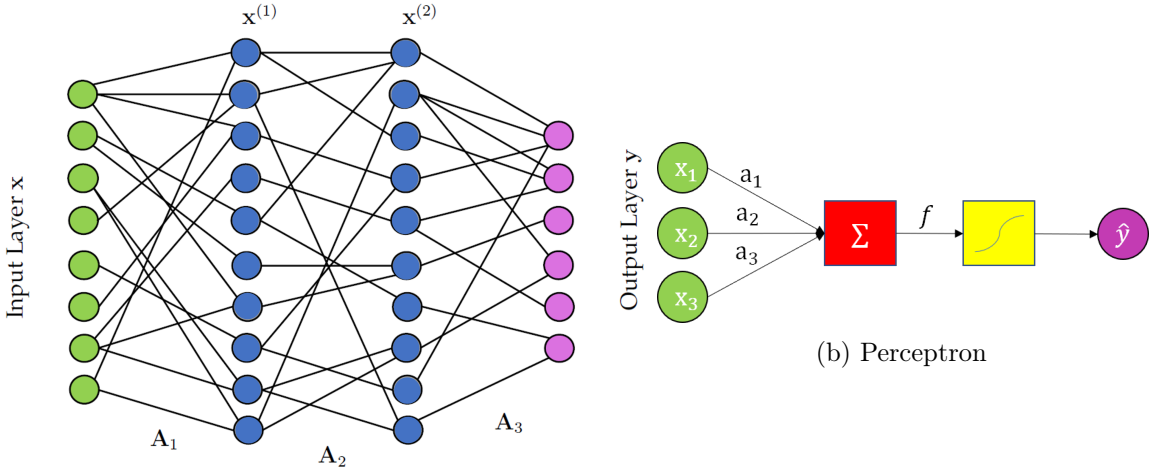
- $\mathrm{p}(\omega|\mathcal{H}_k,\mathcal{R}_j)$ is the pdf of ω conditioned by manoeuvre $\mathcal{H}_k$ and region $\mathcal{R}_j$;
- $\mathrm{H}(\mathrm{p}(\omega|\mathcal{H}_k,\mathcal{R}_j))$ is the Shannon Entropy for distribution $\mathrm{p}(\omega|\mathcal{H}_k,\mathcal{R}_j)$;
- $\mathrm{P}(\mathcal{H}_k)$ is the a-priori probability for manoeuvre $\mathcal{H}_k$.

3. **Joint MAP**: Given $\hat{\mathbf{x}}^i_{t_0}$ and $\hat{\mathcal{R}}_j$, the estimated manoeuvre $\hat{\mathcal{H}}_k$ is the one that maximizes the Joint A-Posteriori probability $\forall\mathcal{H}_k \in \mathcal{H}$

$$\hat{\mathcal{H}}_k = \max_{\mathcal{H}} \left(\mathrm{p}(\mathcal{H}_k|[\hat{\omega}^i_{t_o}, \hat{a}^i_{t_o}, \hat{v}^i_{t_o}]^T, \hat{\mathcal{R}}_j)\right) \quad (3.6)$$

3.2.2 Machine Learning based approach

For the ML based approach, we exploit a neural network to predict the vehicle manoeuvre.
A neural network is a computing system that maps some input data, whatever its nature is, to an output for classification purposes, following a certain complex law. Before reaching the output, the data passes through many internal steps, called hidden layers. For each layer, to get an output in a specific node, the network linearly combines the output values of the previous layer that are connected to that specific node, and applies a non-linear mapping function, called activation function, to the result of this linear combination. In general, the main elements that constitute a network are the nodes, or neurons, and the edges, or connections. Both of them represent numbers, or coefficients.
Fig.3.5 provides two examples of simple neural networks. Specifically, on the left, Fig.3.5a illustrates a FF network [54], with input layer $\mathbf{x}$, output layer $\mathbf{y}$, two hidden layers in the middle, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$, where $\mathbf{x}^{(j)}$ represents the output of the generic hidden layer j, while j determines the sequential ordering, and the matrices $\mathbf{A}_1$, $\mathbf{A}_2$, $\mathbf{A}_3$, where the matrix $\mathbf{A}_j$ compactly denotes the weights connecting the network from layer j to layer j+1.
Fig.3.5b, on the right, depicts the elemental network, with no hidden layers, that we call perceptron. The perceptron illustrates more in detail the mapping that occurs at each layer of the FF network. We linearly combine the inputs x_k through the coefficients a_k. Then, we input the result of the linear combination into the activation function $f(\cdot)$, which provides the final output $\hat{y}$.
A neural network can have an indefinite number of input and output nodes, hidden layers and connections. In addition, for each layer, the network can apply different



(a) Feed-forward neural network

(b) Perceptron

Figure 3.5. Two examples of neural networks: FF (Fig.3.5a) and perceptron (Fig.3.5b).

activation functions, but it is common to use the same for each layer. Typically, the most common are the sigmoid function, the hyperbolic tangent, and the Rectified Linear Unit (ReLU) function, which we report in Fig.3.6.

In the general case, the final input-output relation that represents the mapping is a

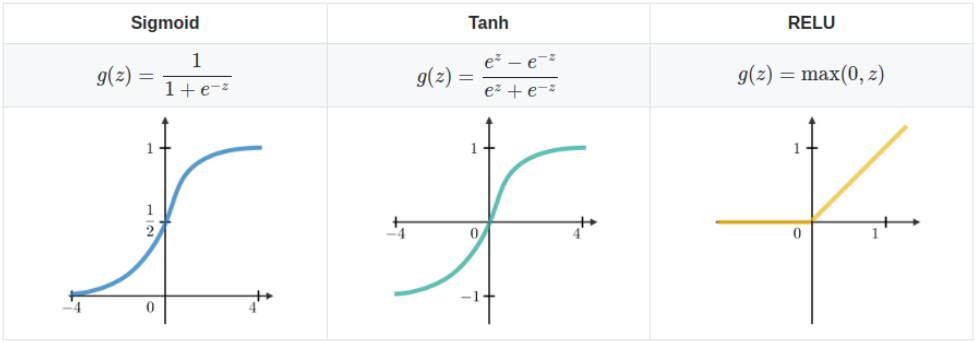


Figure 3.6. Examples of activation functions.

complex non linear composition of all the activation functions. Assuming the same architecture as Fig.3.5a with M layers, we define the composite input-output mapping as

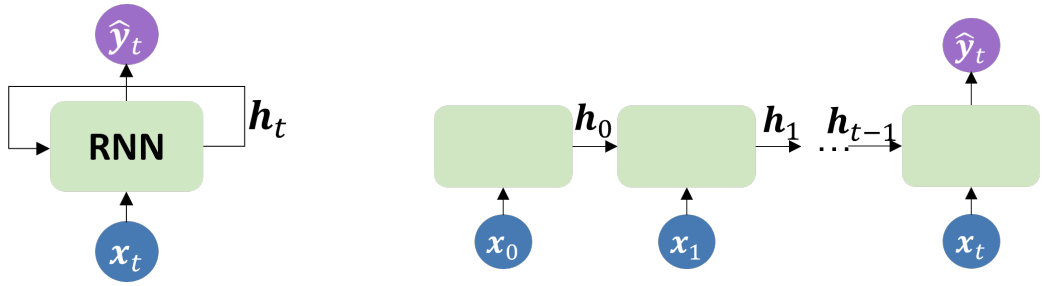
$$\mathbf{y} = \mathrm{f}_{\mathrm{M}}\left(\mathbf{A}_{\mathrm{M}}, ..., \mathrm{f}_2\left(\mathbf{A}_2, \mathrm{f}_1\left(\mathbf{A}_1, \mathbf{x}\right)\right)\right) \tag{3.7}$$

where $\mathrm{f_j}$ is the activation function used at layer j.

Goal is to provide the best mapping of $\mathbf{x}$ into $\mathbf{y}$, which means to find the weights $\mathbf{A}_{\mathrm{j}}$, $\mathrm{j} = 1, 2, ..., \mathrm{M}$ that minimize the error in the mapping procedure.

In the context of neural networks, the process that aims to find the optimal weights is called training. It consists of an iterative procedure, where we feed the neural network with some training data to optimize the classification.

The training procedure consists of two fundamental passages: the stochastic gradient descent and the back-propagation algorithms.

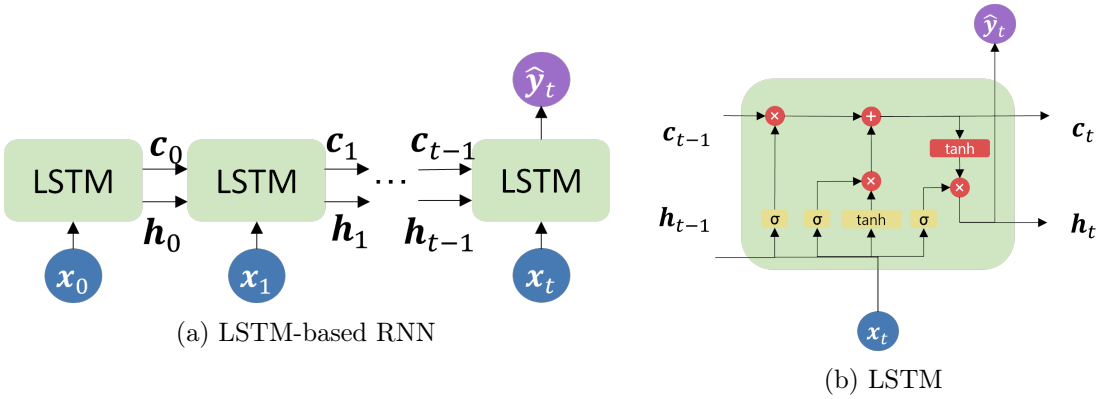


(a) RNN (compressed form)

(b) RNN (expanded form)

Figure 3.7. Example of RNN: compressed form (Fig.3.7a) and its equivalent expanded representation (Fig.3.7b).

The former is crucial to approximate the gradient with a single data point, instead of using all available data, since training procedure is computationally expensive due to the size of the neural networks and the big amount of weights that are involved in the gradient computation.
The latter is a method used for computing the gradient descent required for the training of neural networks. Based upon the chain rule, it exploits the compositional nature of neural networks in order to frame the optimization problem for updating the weights of the network [54].
There are many neural network architectures available today in scientific community. Some of them are very complex, contain many building blocks, and there is still room for new solutions.
For vehicular trajectory prediction, DL strategies have become popular over the recent years, as we discussed in Section 2.2. Mostly, the proposed solutions include RNNs, often with LSTM gate cells for a better control of information flow (Fig.2.8a), and CNNs. The former are more suited for sequential tasks, where input has a certain sequential order, and, to process data sequences, a recursive relation at each step is implemented. The latter consists of sequences of 2D convolutional and pooling layers and an output FF soft-max layer to extract compressed feature maps from data, in order to detect and classify targets [38]. For our task, we decided to implement a LSTM-based RNN since it is considered the best approach to deal with vehicle state sequences in time [38].
A RNN has a different structure, compared to a FF neural network. Indeed, it is used in problems when we deal with input data with a sequential structure. The sequence can also occur in time. In this case, each iteration of the sequence represents a single time step.
Fig.3.7 shows an example of RNN with an input time sequence. Specifically, Fig.3.7a depicts the RNN in a compressed fashion, while Fig.3.7b illustrates its equivalent expanded form. In this case, the figure shows a "many-to-one" RNN example since the input data is a sequence and the output layer is used only in the final time step. We can have also "many-to-many" networks, where the output is a sequence as well. In general, input and output may be vectors, with different dimension.
The key point of a RNN is that the input continuously flows in the network, giving a sense of 'memory' between the time steps. Following the example in Fig.3.7b, the generic input $\mathbf{x}_k$ at step k is manipulated in the green cell, which we call recurrent



(a) LSTM-based RNN

(b) LSTM

Figure 3.8. Example of LSTM-based RNN (Fig.3.8a) and LSTM internal structure (Fig.3.8b [56]).

unit in the context of RNNs, providing the output vector $\mathbf{h}_k$, that we call hidden state. Then, this output becomes the input of the network cell at step k+1, where the procedure repeats. So, for each time step, the output of a cell depends on the current input and on the hidden state of the previous time step. This is the recursive relation that occurs at each step of a RNN.
Another crucial aspect is the recurrent unit that is used at each step of the network. Typically, the LSTM gate cell is a common choice. This cell has a number of properties that make it favourable over other cells. For example, it can track information throughout many time steps. In addition, it solves the vanishing and the exploding gradient problems. Indeed, in a RNN, the input increases or decays exponentially over time, causing problems in the training procedure. An LSTM uses a gating system to overcome this issue [40][55].
Fig.3.8 shows an example of LSTM-based RNN and the LSTM structure, with the details on how the input data flows to provide the output. Specifically, Fig.3.8a depicts the LSTM-based RNN, while Fig.3.8b illustrates the typical LSTM internal structure [56].
The difference, with respect to the previous RNN example, is that an additional state flows in the network across the recurrent units. It is the cell state ($\mathbf{c}_k$ for the k-th step), that contributes to the manipulation of the input data and provides uninterrupted gradient flow [56].
The LSTM cell structure provides many advantages in terms of efficiency. Indeed, we can summarize its contribution in four passages. First, it forgets the irrelevant parts of the previous state. Second, it stores the new relevant information into the cell state. Then, it selectively updates cell state values. Finally, it controls the flow of information sent to the next time step [56].
Putting the RNN and the LSTM concepts together, we can present, in Fig.3.9, the architecture of our implemented LSTM-based RNN for manoeuvre prediction. The network consists of 3 layers of LSTM gate cells.
The input features are: the relative 2D position with respect to the PoI, speed in m/s, acceleration in m/s^2 and yaw rate in rad/s. The output is composed by the three destinations of the intersection $\{Straight, Right, Left\}$, and for the classification a final soft-max layer is applied.

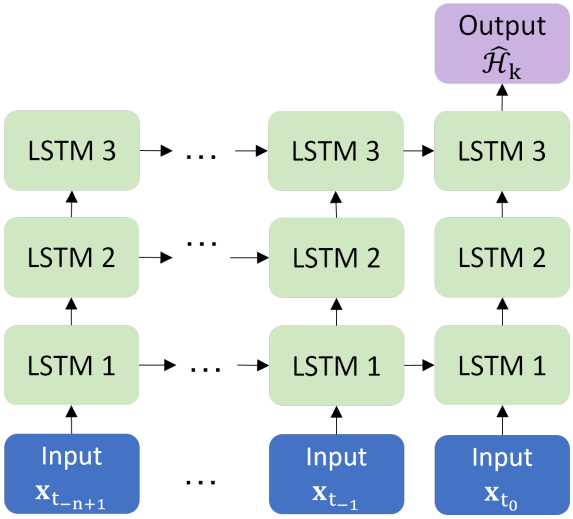


Figure 3.9. Diagram representing the proposed LSTM-based RNN.

Input data tracks are properly split and preprocessed in the same way for training and testing procedures. Each track is split into every possible sequence that can be input into the RNN.
We compactly represent the input data tracks and the split in the following two equations:

$$\mathcal{M} = \left\{ \left(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, ..., \mathbf{x}_{T_j} \right)_j, y_j \right\}_{j=1}^{N_v} \tag{3.8}$$

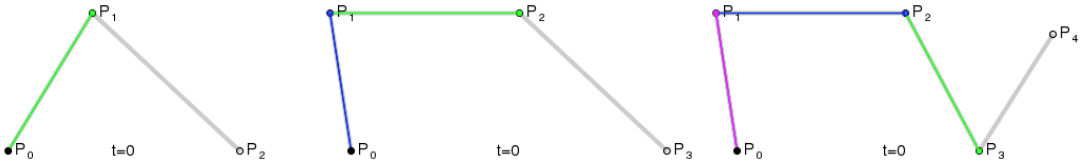
$$\mathcal{S} = \left\{ \left\{ \left(\mathbf{x}_\omega, \mathbf{x}_{\omega+1}, \mathbf{x}_{\omega+2} ..., \mathbf{x}_{\omega+n} \right)_j, y_j \right\}_{\omega=1}^{T_j-n} \right\}_{j=1}^{N_v} \tag{3.9}$$

In Eq.3.8, $\mathcal{M}$ is the set of all manoeuvre samples, T_j is the total track length of vehicle j, N_v is the total number of vehicles, $(\mathbf{x}_t)_j$ is input vector state at time t for vehicle j, y_j is true manoeuvre of vehicle j. To train and test the model, we split all the tracks into every possible consecutive sequence of length n (Eq.3.9), where n is defined as the number of steps in the RNN. To conclude, $\mathcal{S}$ is the complete preprocessed dataset, where we take consecutive samples from each track j to feed the network.

3.3 Trajectory Prediction

For trajectory prediction, the strategy involves the combination of information from digital maps and Bézier curves [9], which are parametric curves based on Bernstein Polynomials, used in graphics, animation, robotics and related fields.
A n-th order Bernstein polynomial B(t) is defined as

$$B(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} \qquad \begin{cases} 0 \leq t \leq 1 \\ \binom{n}{i} = \frac{n!}{i!(n-i)!} \end{cases} \tag{3.10}$$



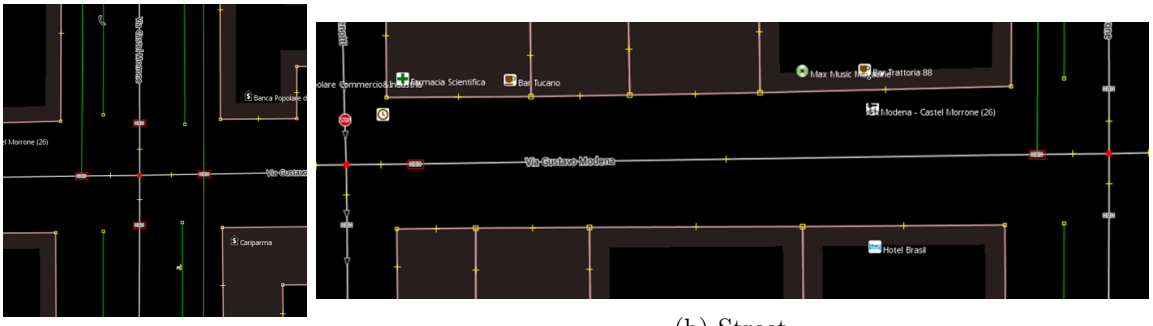
(a) Quadratic curve (b) Cubic curve (c) Quartic curve

Figure 3.10. Examples showing the output provided step by step by De Casteljeau algorithm, in the construction of quadratic (3.10a), cubic (3.10b) and quartic (3.10c) Bézier curves (from [10]).

The mathematician Paul De Casteljeau applied Bernstein Polynomials for the first time to computer-aided design, developing a numerically stable method for evaluating the curves. The algorithm, which takes his name, needs an input set of points P_i, namely "Control Points" or "polygon points", and following the structure of Bernstein Polynomials B(t), it builds the Bézier curve BZ(t) on top of these points. The order of BZ(t) is related to the number of polygon points used.
A n-th order Bézier curve BZ(t) is defined as

$$BZ(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} P_i \qquad \begin{cases} 0 \leq t \leq 1 \\ \binom{n}{i} = \frac{n!}{i!(n-i)!} \end{cases} \tag{3.11}$$

The algorithm is fast, efficient, and flexible since it allows to build complex curves as the sum of simple polynomials. In addition to this, the number of polygon points and their distribution and ordering allow to set up any possible shape of these curves, and this is one of the critical advantages of using Bézier curves. An example that shows the way the algorithm works, given the input polygon points, is illustrated in the GIFs in Fig.3.10. We refer to [10] and related references for the detailed explanation of De Casteljeau algorithm.
From these animations, we understand how Bézier curves are mighty for our scope. Indeed, if we provide to the algorithm the proper number, distribution, and positioning of polygon points in the vehicular context, we can obtain a final trajectory that is compliant both to network topology and vehicle dynamics, leading to high prediction accuracy also in the long term.
In practice, to predict vehicle trajectory in the proximity of the AoI, we need to combine physics-based and Bézier approaches because we need to consider not only current vehicle position but also where its predicted trajectory is going to fall. So, we use the physics-based approach to predict those portions of vehicle trajectory that fall out of the AoI, i.e., within current and destination streets, while we apply Bézier Curves to predict the part that falls within the AoI.
To sum up, the final trajectory near the AoI is a concatenation of three trajectories, where the external ones, situated in the current and the destination streets, come from a physics-based approach and the one in the middle, situated in the AoI, is the result of Bézier curves implementation.
The combination of the two approaches, together with the support of digital maps, is a promising solution for trajectory prediction in complex environments since we expect to provide high accurate trajectories in the long term, compliant to vehicle



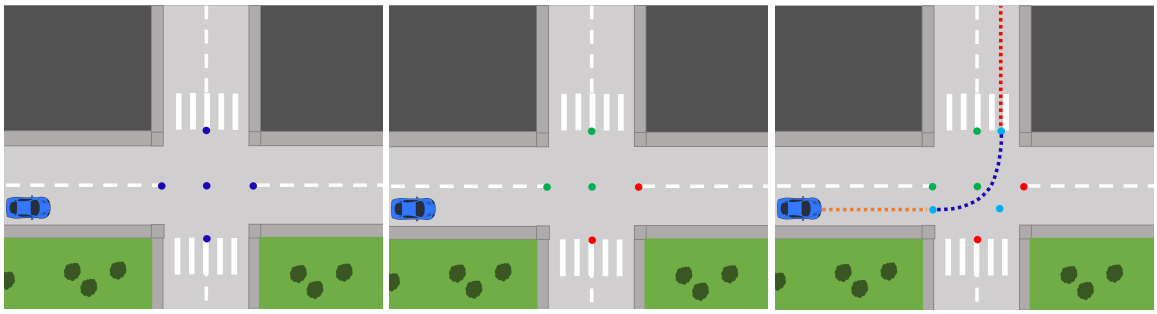
(a) intersection

(b) Street

Figure 3.11. Two urban areas of Milan, from OpenStreetMap.

dynamics and network topology, using flexible, efficient and fast algorithms.
We need to clarify how to practically identify an AoI, which is essential to establish a rule of thumb to separate streets and AoIs and set up the concatenation of the trajectories correctly. Then, we can illustrate the main passages of the strategy, focusing on the setup of the polygon points to build the trajectory within the AoI.
Regarding the first problem, Fig.3.11 can help us better understand the point. We can see two examples of the map provided by OpenStreetMap, showing two urban areas of Milan. Specifically, in Fig.3.11a we notice an intersection, a possible AoI, with a central point and some other nearby points highlighted in red. The point in the center is what we call PoI, while the other points are the points at the edge of the AoI, and for this reason, from now on, we call them edge points.
Essentially, in OpenStreetMap, the map is provided as a collection of points. Each point belongs to a specific facility (a building, foliage, or a street), and each of these facilities has an ID that identifies them uniquely. So, in OpenStreetMap, every facility is represented with an ID and consists of a set of ordered points. In a street, we have two very important points: the PoIs, related to the AoIs situated at the beginning and end of that specific street. Typically, the number of points belonging to a street can vary, but in general, we have at least four points, where two corresponds to the PoIs, and the other two are the edge points of the corresponding AoIs.
In Fig.3.11b you can see another area of Milan, actually near the area illustrated in Fig.3.11a, to better visualize how a street is represented in OpenStreetMap, together with its related four points, highlighted in red. The outer points are the PoIs, while the inner ones are their relative edge points.
To conclude, we can identify an AoI simply through the central PoI and its related edge points. These points are crucial for the setup of the polygon points, for the distinction between non-AoIs and AoI, and for the implementation of Bézier technique within the AoI.
Now, we will go into the details of our trajectory prediction strategy. For the entire explanation, we will make an example, following the sequence depicted in Fig.3.12 and the block diagram illustrated in Fig.3.13.

1. We initially assume a non-CAV is approaching a 4-way unsignalized intersection, where each street that crosses the intersection is two-way, and that the vehicle is out of the AoI. The initial setting is provided in Fig.3.12a, where we can see the vehicle and the intersection with some points, depicted in blue, representing the



(a) Initial setting (b) Manoeuvre prediction (c) Trajectory Prediction

Figure 3.12. Example showing the main passages of trajectory prediction block.

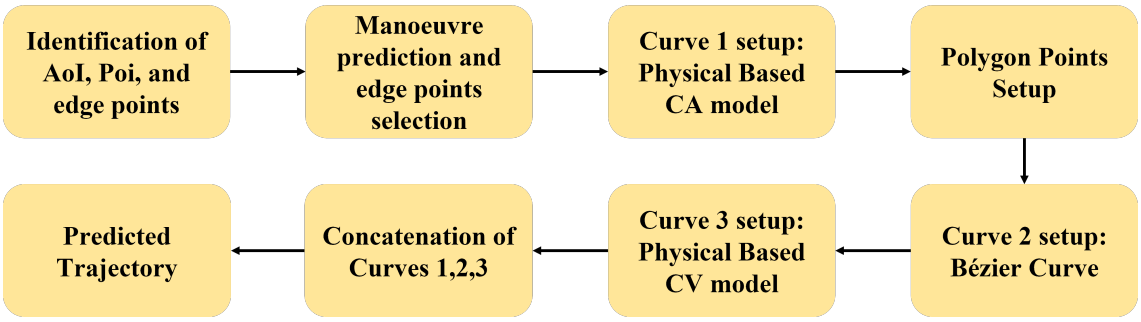


Figure 3.13. Trajectory prediction strategy.

PoI and the relative edge points depicted in Fig.3.11a. In addition, we assume the manoeuvre prediction block has already predicted that the vehicle will turn left. From this point, we consider the implementation of a trajectory associated with a left turn in the case of two-way current and destination streets. The reasoning is similar for a right turn or to go straight on, even in one-way streets, where the algorithm can be easily adapted.

2. Given these initial assumptions, according to the predicted manoeuvre, the algorithm selects only a subset of the blue points to build the trajectory within the AoI, dropping the remaining part. To distinguish the points, we show them with different colors in Fig.3.12b. Since we assume left manoeuvre, to build the trajectory, we take only the green points, which constitutes our initial set of points, while we discard the red ones.
   So, up to now, this set consists of three points: the PoI and two edge points, one belonging to the current street, the other to the destination street. We can call the former "approaching" point and the latter "leaving" point since, according to vehicle direction, the points respectively fall in the approaching and leaving areas of the AoI.

3. Given this initial set of points, we are ready to predict the trajectory as a composite curve (Fig.3.12c) combining physics-based and Bézier approaches:

   (a) First, we apply a physics-based approach, specifically a CA model, to predict the part of the trajectory out of AoI and within vehicle current street. This curve, which we call Curve 1, starts from vehicle position,

follows the direction provided by the current street and terminates in correspondence of the predicted point at minimum distance from the edge approaching point.

(b) Then, we extend Curve 1 into the AoI to setup two out of three polygon points.
The first one is the predicted point at minimum distance from the edge approaching point.
For the second point, we extend Curve 1 in the AoI and take the predicted point at minimum distance from the PoI. Given this point, our second polygon point will be the point of Curve 1 extension at a distance of two points from it, moving forward in the curve. Taking the latter point in place of the former, we avoid predicting sharp curves that might invade the opposite lanes when turning to the left since these curves would be unrealistic. For example, in the right turn case, the strategy to choose the second polygon is the same as for the left turn case, with the only difference that we move backward in Curve 1 extension to avoid the opposite problem: predict smooth curves that would inevitably invade the opposite lane, which would be unrealistic in the same way.
The third and last point is obtained by applying a physics-based model (CV or CA, it does not matter in this case) to get a straight curve that starts from the second polygon point, follows the direction provided by the destination street and terminates in correspondence of the predicted point at minimum distance from the edge leaving point. This specific point will be our third polygon point.
In this way, in total we have three polygon points, that are shown in cyan in Fig.3.12c. Once these points are available, we can feed them to De Casteljeau algorithm to predict the portion of the trajectory that falls within the AoI, which we call Curve 2.

(c) Finally, similarly to the case of Curve 1, we apply a physics-based approach, specifically a CV model, to predict the last part of the trajectory, out of the AoI and within the destination street, which starts from the third polygon point, which is the end of Curve 2, and follows the direction provided by the destination street. We call it Curve 3.

(d) Once we have Curve 1, Curve 2, Curve 3, we simply concatenate them to finally obtain output predicted trajectory.

Fig.3.12c shows the complete composite trajectory, together with the polygon points in cyan. The image is informative only, since the curve is just a sketch that represents a possible trajectory. In the figure, we can see the different parts that compose the entire trajectory with different colors for a better distinction. In particular, Curve 1 is the initial straight line depicted in orange, Curve 2 is the trajectory within the AoI, illustrated in blue, and, to conclude, Curve 3 is the final straight line, in red.

Chapter 4

Simulation Results

This chapter shows some results on manoeuvre and trajectory prediction, obtained from a simulation performed in Milan, specifically in the Dateo area near Città Studi District. We implemented the map-assisted VTP system, using SUMO for the traffic generation and OpenStreetMap for the digital maps. The Area of Interest we studied is a 4-way unsignalized intersection since it is one of the most common urban AoIs where we can obtain valuable results, and where the trajectory prediction problem is non-trivial.
We simulated one hour of traffic, with vehicles running only along the marked yellow streets (Fig.4.1a), which cross at right angles towards the considered intersection area. The average input flow along the marked yellow streets is 900 veh/h. The simulation includes many types of vehicles, going from cars to trucks, deliveries, and emergency vehicles, and for the setup of main characteristics and dynamical properties of the vehicles we follow the suggestions proposed directly by the SUMO research community. More details are in [57], where you can find the vehicle parameters and the sources the SUMO research community refers to for the parameters' setup.
First, in Section 4.1, the results on manoeuvre prediction are illustrated, comparing Model-based and ML based approaches. Then, in Section 4.2, the results on trajectory prediction are presented.

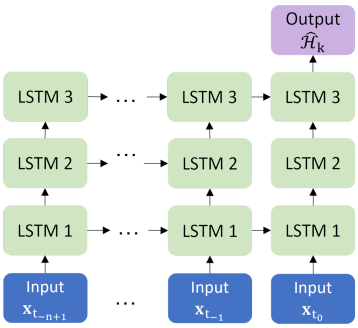
4.1 Manoeuvre prediction

For manoeuvre prediction, we define the set of manoeuvres as $\mathcal{H} = \{\mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3\}$ where $\mathcal{H}_1$ means "Go Straight", $\mathcal{H}_2$ means "Go right" and $\mathcal{H}_3$ means "Go left", and we evaluate the performance of the implemented solutions in terms of probability of Successful manoeuvre Classification, defined as:

$$\mathrm{P_s} = \sum_{\mathrm{k}=1}^{\mathrm{N}} \Pr\{\mathrm{Predictedmanoeuvre} = \mathcal{H}_\mathrm{k} | \mathcal{H}_\mathrm{k}\} \tag{4.1}$$

where N is the total number of manoeuvres in the set $\mathcal{H}$.
Regarding the Model-based approach, to divide the AoI, we define the set of regions as $\mathcal{R} = \{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3\}$ and, defining the distance from the PoI as d, a vehicle is inside region $\mathcal{R}_1$ if $\mathrm{d} \leq 10$ [m], inside region $\mathcal{R}_2$ if $10 < \mathrm{d} \leq 20$ [m] and inside region $\mathcal{R}_3$ if

(a)



(b)

Network Parameters	Network Values
Sequence Length	10 timesteps (1s)
Number of LSTM Layers	3
Number of Hidden Units	112
Dropout Layer	50%
Training Algorithm	SGDM
Learning Rate	0.1
Decay Factor	0.1
Epochs	10
Training Mini-Batch Size	128

(c)

Figure 4.1. (a) A view, from the map, of the scenario used for the simulation. (b) Diagram of proposed LSTM-based RNN architecture. (c) Network training parameters (Note: SGDM is the Stochastic Gradient Descent Method).

$20 < \mathrm{d} \leq 30$ [m].
For the ML based approach, we use a LSTM-based RNN network with 3 layers, as depicted in the diagram in Fig.4.1b. We outline network and training parameters in Fig.4.1c. Input size is fixed, since we feed the network with 1s vehicle history track. To train and test the network, we performed the simulation twice, applying a random generator in order to provide two different simulations with the same vehicle input flow characteristics. Specifically, to train the network, we used only the vehicle samples at 40 m distance range from the PoI.
Fig.4.2 illustrates the probability of Successful manoeuvre Classification for Model-based and ML based approaches, while Fig.4.3 shows the accuracy evolution during the training stage for the ML based approach.
In the former approach, we compare the implemented methods, described in Section 3.2.1, for each region, while in the latter approach the probability of success is evaluated for different distance ranges, specifically 30 m, 20 m, and 10 m, limiting the testing dataset within the corresponding range.
The plots clearly show that ML based approach completely outperforms the Model-based one. Indeed, we can see that, in the Model-based approach, the probability of success under 50% until the vehicle reaches $\mathcal{R}_1$, where, in the best case, slightly exceeds 70%. In the ML based approach, the probability of success touches a value of 70% in the worst case, at 30 m distance from the PoI, and goes beyond 80% at 10 m, which is something expected since the ML based approach exploits vehicles' history track, unlike the Model based one that uses the current state only.
So, we conclude that, for manoeuvre prediction, the ML based approach, contrary to the Model-based one, is a promising solution since it guarantees higher performance, it is scalable, and can be adapted according to the vehicular scenario and the specific AoI at hand.

4.2 Trajectory prediction

For trajectory prediction, we test the proposed approach in the same simulation used to test and compare the manoeuvre prediction strategies previously described in

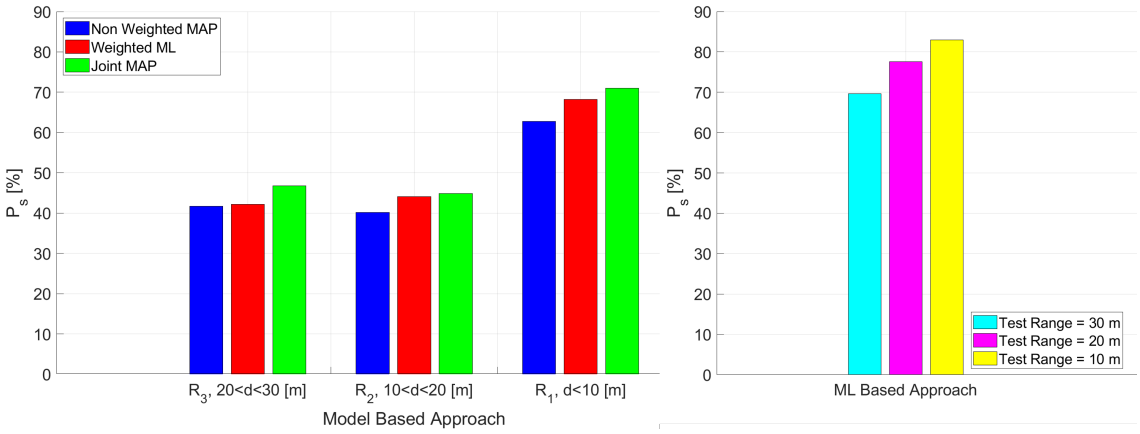


Figure 4.2. probability of Successful manoeuvre Classification for Model-based and ML based approaches on the left and on the right, respectively.

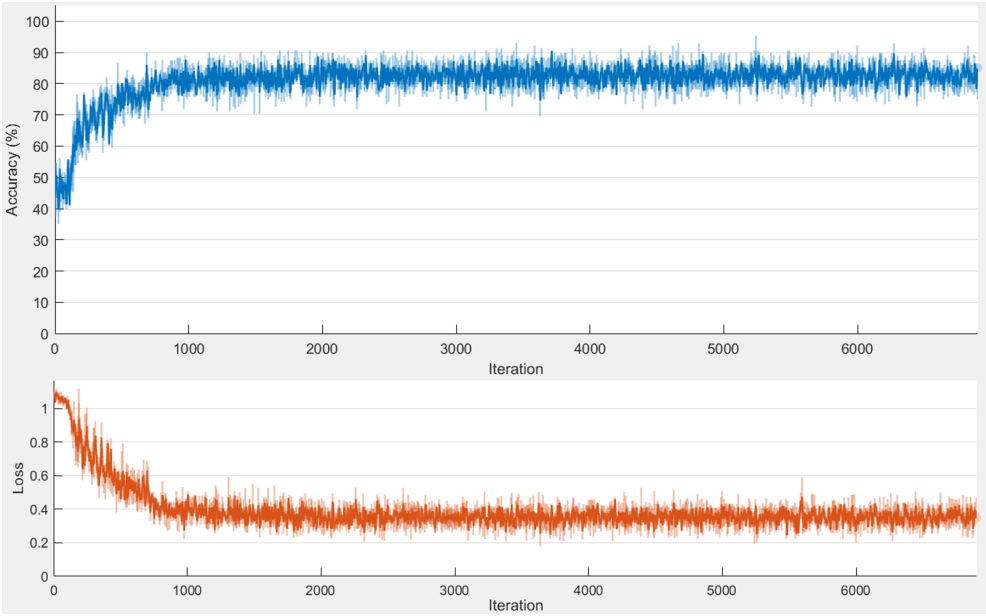
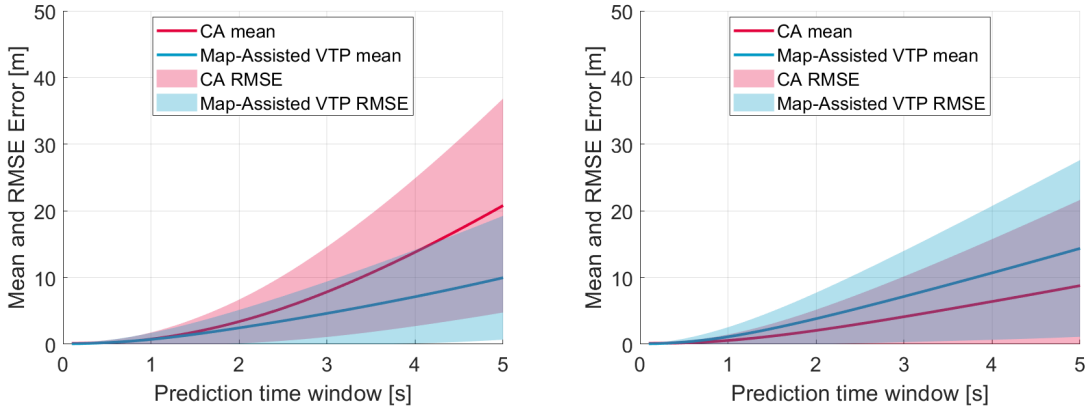


Figure 4.3. accuracy evolution during NN training Procedure.

Section 3.2.
Specifically, we evaluate the performance of the implemented algorithm in terms of prediction error for different prediction time windows. We define the error as the Euclidean Distance between the true position and the predicted one, evaluated at the same prediction time instant.
In formulas, we can write it as:

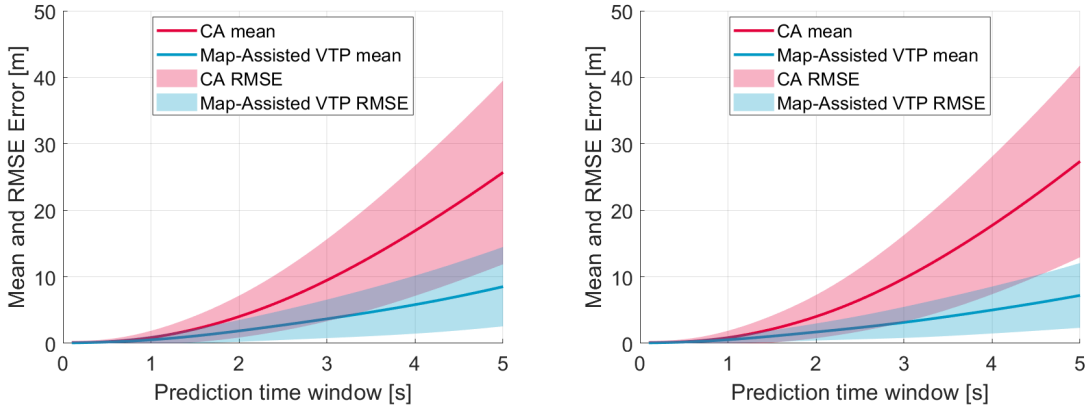
$$e_{t_i} = \|\mathbf{s}_{t_i} - \hat{\mathbf{s}}_{t_i}\| \qquad i = 1, 2, ..., p$$

where $\|\cdot\|$ is the norm operator, $\mathbf{s}_{t_i}$ is the vector containing the x and y coordinate of the true position at prediction time instant t_i, $\hat{\mathbf{s}}_{t_i}$ is the vector containing the x and y coordinate of the predicted position at prediction time instant t_i and i is the prediction step.



(a) All trajectories (b) Straight trajectories

Figure 4.4. Mean and RMSE of trajectory prediction error for general and straight trajectories and for different prediction windows



(a) Left turn trajectories (b) Right turn trajectories

Figure 4.5. Mean and RMSE of trajectory prediction error for left and right turn trajectories and for different prediction windows

Fig.4.4 and Fig.4.5 show some results on prediction error, while the GIF in Fig.4.6 allows us to practically see the implemented algorithm in action during the simulation of a vehicle that will turn left. In particular, we obtained these results by applying the algorithm only when each vehicle is at a certain distance from the intersection PoI. Defining this distance as d_{poi}, we perform trajectory prediction only when $d_{max} \leq d_{poi} \leq d_{edge}$, where $d_{max} = 25$ m, and d_{edge} is the distance between the PoI and the edge approaching point. The reason behind this choice is to evaluate the algorithm's performance only when the vehicle is near the intersection but somehow far from the PoI at the same time. Also, we decided to compare the proposed solution with a simple application of a Physics-based approach, specifically using a CA motion model as a baseline.

Going into detail, Fig.4.4 and Fig.4.5 show the plots of the mean error for both our proposed approach and the CA model, for a prediction time window up to 5 s. In addition, the plots provide also the Root Mean Squared Error (RMSE) plotted as a shaded area around the mean. In particular, Fig.4.4a illustrates the error trend for

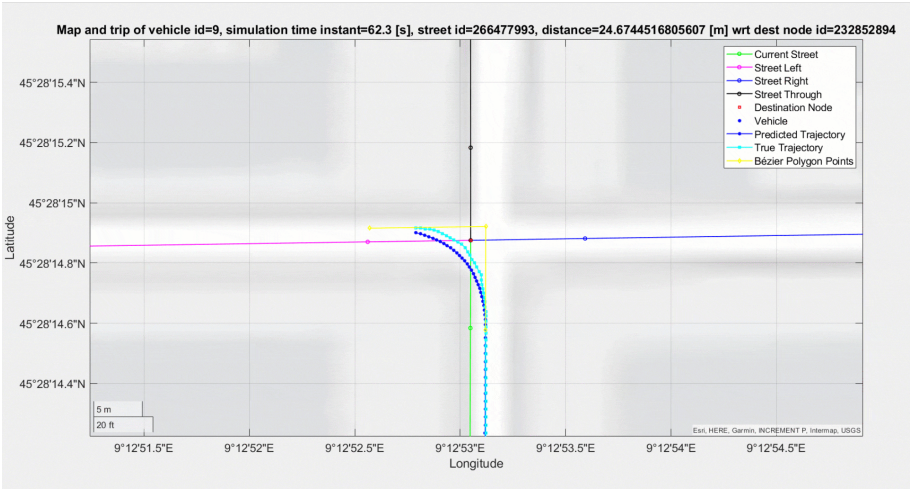


Figure 4.6. Examples showing trajectory prediction algorithm in action during the simulation.

the entire simulation, Fig.4.4a shows the error for straight trajectories, while Fig.4.5b and Fig.4.5a, on the left and on the right, show the error evolution by considering the prediction performed only for the left turn and the right turn trajectories, respectively. From the figures, we can see that our proposed approach outperforms the Physical-based one. Indeed, the mean error and the RMSE with map-assisted VTP approach are smaller than the Physics-based approach. In particular, we can notice that for all the trajectories, the map-assisted VTP mean error touches a maximum value of 10 m for a prediction horizon of 5 s, with a corresponding RMSE of 10 m at most, while for the Physics-based approach, the mean and the RMSE go over 20 and 15 m, respectively.
It is worth mentioning that the performance of the map-assisted VTP approach seems to increase when dealing with turnings and curved trajectories. Indeed, for the left case, we can notice that map-assisted VTP mean error is slightly better than the general case, and its RMSE is narrower, going to a value of 6 m more or less. For the right turns, we have a similar result for the mean, and RMSE is even better, with a value slightly lower than 5 m.

On the contrary, for the turning cases, it is quite evident that the Physics-based approach diverges a lot, with mean and RMSE values going over 25 m and 15 m, respectively. This result is expected since the CA model is purely a Physical model, based on simple Kinematic laws, which predicts a straight trajectory that can capture neither the complex curves nor the topology of the vehicular environment.
On the other hand, looking at the plots, we can state the map-assisted VTP system suits very well for the prediction of curved trajectories, because it can exploit properly the information collected from the digital maps that help us building the curves. However, from the plot in Fig.4.4b, we can say that our approach suffers from the trajectory prediction of straight curves. The reason is because we exploit Bézier curves in our strategy, which are more suited to deal with curves than straight lines.
So, the conclusion that arises from these plots is that our prediction system is a promising approach to predict vehicular trajectories in realistic traffic scenarios, especially in the case of turnings and curve trajectories, since it can capture the complexity of vehicular environment and network topology by properly using the

Table 4.1. Comparison of Physics-based and map-assisted VTP system with existing works

	RMSE [m]				
Works	**1s**	**2s**	**3s**	**4s**	**5s**
CA	1	3.29	6.76	11.1	16.04
CA left	1.02	3.14	6.17	9.8	13.81
CA right	1.04	3.24	6.54	10.35	14.42
CA Straight	0.9	3.1	6.04	9.35	12.87
map-assisted VTP	0.97	2.7	4.79	7.03	9.29
map-assisted VTP left	0.6	1.64	2.9	4.36	5.97
map-assisted VTP right	0.51	1.28	2.34	3.53	4.87
map-assisted VTP Straight	1.44	3.9	6.85	10.04	13.25
[58]	0.73	1.78	3.13	4.78	6.68
[59]	0.49	1.41	2.6	4.06	5.79
[60]	0.57	1.51	2.51	3.71	5.12
[45]	0.58	1.26	2.12	3.24	4.66
[42]	0.61	1.27	2.09	3.1	4.37
[61]	0.56	1.19	1.93	2.78	3.76
[62]	0.64	1.13	1.8	2.62	3.6

information collected from digital maps.
In Tab.4.1, you can find a comparison between both the Physics-based and map-assisted VTP approaches with some existing works. Specifically, here we compare the RMSE for different values of the prediction horizon, going from 1 to 5 s. In particular, for the two approaches evaluated in this work we reported the RMSE values obtained for the four cases we discussed (all the trajectories, left turn, right turn, straight trajectory). From the table, we can see that some works outperform our approach, while it is the opposite case in others, especially when dealing with trajectory prediction in the case of turnings. Actually, most of the cited works have to cope with trajectory prediction in more straightforward cases (predicting a lane change in a highway) or implement sophisticated solutions that provide better results but with higher complexity.
So, we can conclude that the map-assisted VTP system is a promising approach, and a good starting point, for new research activities related to trajectory prediction in complex environments, thanks to the support of digital maps and the usage of simple, fast, and very efficient techniques like the Bézier Curves, which are the main strengths of the proposed strategy.

Chapter 5

Conclusion and Future work

This Chapter aims to conclude this master thesis, summarizing what we discussed so far and proposing suggestions for future research directions.
In particular:

- Chapter 1 introduced the discussion about Mobility and trajectory prediction of non-autonomous vehicles and pedestrians in the context of V2X communications. The introduction has revealed the crucial importance and the significant impact a trajectory prediction system might have in future vehicular networks.
  Indeed, the the high dynamics of vehicular mobility and the physical characteristics of mm-Wave communication channel introduces, in V2X systems, an higher sensitivity to blockage, with significant V2X link instability, and a detrimental impact, in terms of QoS, of V2X services.
  Thus, a system that predicts future vehicle trajectories in the long-term is crucial to predict the evolution of the physical communication channel, the position of static and dynamic blockers, and, consequently, to provide the right conditions to proactively set up high data rate, low latency, robust and reliable mm-Wave V2X communication links.

- Chapter 2 provided a SoA review on vehicular mobility modeling and vehicular/pedestrian trajectory prediction.
  Specifically, Section 2.1 gives a detailed description and comparison of main mobility models available in literature, outlining the models implemented in SUMO, and proposes a methodological approach that should be followed in order to build and/or properly choose a mobility model to simulate a realistic scenario.
  Then, Sections 2.2 and 2.3 respectively propose a review on the main vehicular and human motion and trajectory prediction models available to the research community.

- Chapter 3 described the proposed strategy for the implementation of map-assisted VTP system, illustrating in detail the implemented approaches for manoeuvre and trajectory prediction in Sections 3.2 and 3.3, respectively, and Chapter 4 shows some Simulation Results on the proposed solution.
  In particular, in Section 3.2 two strategies were proposed and compared for manoeuvre prediction, a Model-Based approach and a ML Based approach, while Section 3.3 described the proposed method for trajectory prediction.

Simulation results highlighted that for manoeuvre prediction, the ML Based approach completely outperforms the Model-Based one, being a good candidate for implementing future manoeuvre prediction systems. On the other hand, for trajectory prediction, the Bézier curves provide mean and RMSE prediction errors lower than 10 m for a prediction horizon up to 5 s. So, this approach has great potential since we can obtain complex trajectories that are well-compliant to vehicle dynamics and network topology, reaching a good compromise between accuracy and complexity.

This master thesis aims to open new perspectives in the V2X communications research area. Since the topic is huge, and relevant, this work represents only the first stage of a very long journey. Indeed, there is room for improvement, and many possibilities can be explored to continue this activity.
Here we propose some possible future research directions:

- As reported in Chapter 4, we tested the system on a simulation performed in an urban area of Milan, specifically on a 4-way unsignalized Intersection.
  For sure, a possibility could be to simulate a more extended area that consists of many complex AoIs and evaluate the performance of the system for the whole scenario. In this case, the prediction system should adapt to the specific AoIs at hand. So far, the system exploits vehicle position and the digital maps to obtain knowledge about its current street and the AoI it will approach according to its direction.
  What is missing is another kind of information that is essential to deal with manoeuvre prediction in an extended scenario, which is the NN related to that specific AoI.
  As a matter of fact, in a vehicular environment, we have many possible AoIs with different characteristics. It is inappropriate to use a network trained in an AoI with different features from the AoI we are considering because it will undoubtedly provide a poor prediction. Thus, in an extended scenario, we should try something different.
  A possible solution is to train and provide an NN for each AoI. This approach is impossible because we cannot train infinite NNs. On the contrary, we should think about a solution we can easily introduce in a real-world scenario.
  A possible idea might be to identify in the scenario some prototype AoIs that, for their characteristics, they can represent almost the majority of the AoI we might encounter, and then we train an NN for each of them. Once these prototype NNs are ready to use, we can perform manoeuvre prediction whenever we encounter an AoI by analyzing its characteristics and choosing, according to them, the Neural Network suited to that AoI.
  To identify prototype AoIs, the characteristics we might consider are, for example, the number of crossing streets, the number of possible destinations from each street, the shape of the area, and its topology. Following this way of reasoning, for example, we can distinguish between AoIs with different shapes (+, T, V, Y are some possibilities) and different street rules/restrictions, i.e., denied access or presence of traffic lights.
  In this way, during the simulation in the extended scenario, whenever the vehicle

reaches an AoI, the manoeuvre prediction step comes into play, choosing to apply the network trained in the prototype AoI that presents the most similar features of the AoI vehicle is currently approaching.
This issue experienced for the manoeuvre prediction step does not affect the trajectory prediction building block. Indeed, Bézier curves have the remarkable characteristic that they can adapt to the shape of the AoI at hand. This flexibility allows us to extend their usage for more extensive scenarios without modifying the approach, whatever AoIs we might encounter.

- So far, the system works for trajectory prediction of vehicles, assuming that only vehicles are present in the traffic scenario.
  In general, the prediction of VRU agents, such as pedestrians, bikers, riders, is very challenging. Indeed, they are the agents with the highest mobility uncertainty since they can move almost in any area of the vehicular network, either streets or sidewalks, in a disorderly fashion and with variable speed and acceleration profiles.
  This situation might have a non-negligible impact on V2X communications because they become sources of link blockage in no time. Furthermore, the different nature of their mobility, as opposed to vehicles, leads to the need to use motion models and prediction strategies with a completely different approach, making trajectory prediction more difficult.
  Pedestrians are the most unpredictable VRU agents. They can move on their own or in couples; they can form bigger groups, like bee swarms, which in some cases move compactly, maintaining a certain consistency and following some logic, while in some other cases, they can break up, spread, scatter and reassemble very quickly without clear reasoning behind, which well explains the reason why they are considered quite unpredictable.
  So, a system that can predict VRUs mobility is an essential step that we should consider in future research activities.

- Last but not least, we know the prediction system can work relying on the information offered by digital maps. However, up to now, what we provide through the maps is information about static entities, like buildings, foliage, streets, street rules/restrictions that do not change in time. Actually, in real traffic scenarios, we might deal with temporary situations that lead to some map alterations, even slightly. For example, a change of travel direction, closed roads, or the presence of obstacles, static or dynamic objects that may temporary obstruct the streets (think about car accidents, where cars may obstruct the path for a long time or, even worse, the roadworks, that may last from weeks to months in most of the cases).
  Without a doubt, in this case, the topology of the vehicular network areas where these events occur might change a lot, and with static maps, we cannot provide information about these changes. A prediction system that is unaware of these changes might experience a poor prediction performance since it cannot interpret correctly the behavioral changes of non-CAVs involved in the scenario.
  Thus, a system that adapts to these changes is crucial, and it is probably one of the critical steps, if not the most important, that is missing to build a flexible

and efficient prediction system capable of reacting to the presence of temporary objects (we can also include VRU agents, linking this discussion to the point previously mentioned), road obstructions, obstacles, and to the temporary mutations in the vehicular context, a more advanced system that, in short, can manage practically all the possible situations we might encounter in a traffic scenario.
Implementing such a system is quite challenging since we should reason on the main changes that may happen and how to tackle them practically, but its benefits will undoubtedly overcome the efforts in the implementation.
Going to the point, to deal with the aforementioned temporary changes, we should rely on dynamic digital maps. For us, this practically means simply testing simulated environments where we apply some changes in network topology or the street rules and restrictions.
The proposed solution makes sense only if we can implement the system in a real environment where we are sure that a real-time updating process of the digital maps will occur. Otherwise, it is entirely useless. For example, we could propose the solution described in Section 3.1 because it assumes the possibility to rely on static maps, which are available from the edge cloud, and on TV state, that CAVs can retrieve from a CAS procedure.
Similarly, the improvement proposed here should rely on assumptions we expect to meet in future vehicular networks. This solution is feasible because, considering the V2X use cases defined and continuously updated by standardization groups like 5G Automotive Association (5GAA) [63][64], in future vehicular networks, it will be possible to rely on dynamic, up-to-date digital maps.
Indeed, CAVs can exploit their On-Board Unit (OBU) (Cameras, Lidars, Radars, or other HD sensors) to collect accurate real-time information about road hazards, car accidents, presence of static/dynamic temporary obstacles, and street rules modifications. Then, through a CAS procedure and cooperation with network infrastructure, the CAVs can exchange and share this information with HD Map providers, i.e., edge cloud servers, that analyze and merge the collected data to build and quickly provide real-time refreshed HD maps.
This solution can provide positive fundamental implications in V2X networks development, traffic safety, and comfort of mobility, opening new promising perspectives and great opportunities in V2X research activities.

Bibliography

[1] J. G. Andrews S. Buzzi W. Choi S. V. Hanly A. Lozano A. C. Soong and J. C. Zhang. "What will 5G be?" In: *IEEE J. Sel. Areas Commun.* 32.6 (2014), pp. 1065–1082.

[2] SAE International. "Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems". In: (2014).

[3] ETSI. "5G; Service requirements for enhanced V2X scenarios". In: *ETSI TS 122 186 V15.4.0* (2018).

[4] 3rd Generation Partnership Project. "Study on enhancement of 3GPP support for 5G V2X services (Release 16)". In: *TR 22.886* ().

[5] P. Popovski et al. "5G wire- less network slicing for eMBB, URLLC, and mMTC: a communication-theoretic view". In: *IEEE Access* 6 (2018), pp. 55765–55779.

[6] SUMO. *SUMO index*. URL: https://sumo.dlr.de/docs/index.html.

[7] Pablo Alvarez Lopez et al. "Microscopic Traffic Simulation using SUMO". In: *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. URL: https://elib.dlr.de/124092/.

[8] OSM. *OSM index*. URL: https://www.openstreetmap.org/.

[9] J. Choi, R. Curry, and G. Elkaim. "Path Planning Based on Bézier Curve for Autonomous Ground Vehicles". In: *Advances in Electrical and Electronics Engineering - IAENG Special Edition of the World Congress on Engineering and Computer Science 2008*. 2008, pp. 158–166. DOI: 10.1109/WCECS.2008.27.

[10] Wikipedia. *Bézier curve — Wikipedia, L'enciclopedia libera*. URL: http://https://en.wikipedia.org/wiki/B%C3%A9zier_curve.

[11] J. Harri, F. Filali, and C. Bonnet. "Mobility models for vehicular ad hoc networks: a survey and taxonomy". In: *IEEE Communications Surveys Tutorials* 11.4 (2009), pp. 19–41. DOI: 10.1109/SURV.2009.090403.

[12] P. Manzoni et al. "Mobility models for Vehicular Communications". In: ().

[13] Olariu Stephan and Weigle Michele C. *Vehicular Networks: From Theory to Practice ch.12*. 1st. Chapman Hall/CRC, 2017. ISBN: 1138116599.

[14] S. Krauss, P. Wagner, and C. Gawron. "Metastable states in a microscopic model of traffic flow". In: *Phys. Rev. E* 55 (5 May 1997), pp. 5597–5602. DOI: 10.1103/PhysRevE.55.5597. URL: https://link.aps.org/doi/10.1103/PhysRevE.55.5597.

[15] Kai Nagel and Michael Schreckenberg. "A cellular automaton model for freeway traffic". In: *Journal de Physique I* 2 (Dec. 1992), p. 2221. DOI: 10.1051/jp1:1992277.

[16] Wiedemann Rainer. *Simulation des Straenverkehrsflusses*. 1st. 1974.

[17] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. "Congested Traffic States in Empirical Observations and Microscopic Simulations". In: *Physical Review E* 62 (Feb. 2000), pp. 1805–1824. DOI: 10.1103/PhysRevE.62.1805.

[18] P.G. Gipps. "A behavioural car-following model for computer simulation". In: *Transportation Research Part B: Methodological* 15.2 (Apr. 1981), pp. 105–111. URL: https://ideas.repec.org/a/eee/transb/v15y1981i2p105-111.html.

[19] Debashish Chowdhury, Ludger Santen, and Andreas Schadschneider. "Statistical physics of vehicular traffic and some related systems". In: *Physics Reports* 329.4 (2000), pp. 199–329. ISSN: 0370-1573. DOI: https://doi.org/10.1016/S0370-1573(99)00117-9. URL: https://www.sciencedirect.com/science/article/pii/S0370157399001179.

[20] Sven Maerivoet and Bart De Moor. "Traffic Flow Theory". In: *Physics* 1 (Aug. 2005).

[21] Sven Maerivoet and Bart De Moor. "Transportation Planning and Traffic Flow Models". In: (July 2005).

[22] "Vehicular Networks: Macroscopic and Microscopic Mobility Models". In: *Mobility Models for Next Generation Wireless Networks*. John Wiley and Sons, Ltd, 2012. Chap. 12, pp. 153–158. ISBN: 9781118344774. DOI: https://doi.org/10.1002/9781118344774.ch12. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118344774.ch12. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118344774.ch12.

[23] "Microscopic Vehicular Mobility Models". In: *Mobility Models for Next Generation Wireless Networks*. John Wiley and Sons, Ltd, 2012. Chap. 13, pp. 159–172. ISBN: 9781118344774. DOI: https://doi.org/10.1002/9781118344774.ch13. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9781118344774.ch13. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781118344774.ch13.

[24] SUMO. *SUMO car-following-models*. URL: https://sumo.dlr.de/docs/Definition_of_Vehicles,_Vehicle_Types,_and_Routes.html#car-following_models.

[25] Stefan Krauß. "Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics". PhD thesis. Mathematisches Institut, Universität zu Köln, 1998, p. 116.

[26] SUMO. *SUMO sublane models*. URL: https://sumo.dlr.de/docs/Simulation/SublaneModel.html.

[27] Jakob Erdmann. "Lane-Changing Model in SUMO". In: *SUMO2014*. Vol. 24. Reports of the DLR-Institute of Transportation SystemsProceedings. Deutsches Zentrum für Luft- und Raumfahrt e.V., May 2014, pp. 77–88. URL: https://elib.dlr.de/89233/.

[28] SUMO. *SUMO lane change models*. URL: https://sumo.dlr.de/docs/Definition_of_Vehicles,_Vehicle_Types,_and_Routes.html#lane-changing_models.

[29] SUMO. *SUMO Intersection model*. URL: https://sumo.dlr.de/docs/Simulation/Intersections.html.

[30] Daniel Krajzewicz and Jakob Erdmann. "Road Intersection Model in SUMO". In: *SUMO2013 - 1st SUMO User Conference*. Vol. 21. Reports of the DLR-Institute of Transportation Systems. DLR, May 2013, pp. 212–220. URL: https://elib.dlr.de/84363/.

[31] SUMO. *SUMO Pedestrian model*. URL: https://sumo.dlr.de/docs/Simulation/Pedestrians.html.

[32] Stephanie Lefevre, Dizan Vasquez, and Christian Laugier. "A survey on motion prediction and risk assessment for intelligent vehicles". In: *Robomech Journal* 1 (July 2014). DOI: 10.1186/s40648-014-0001-z.

[33] R. Schubert, E. Richter, and G. Wanielik. "Comparison and evaluation of advanced motion models for vehicle tracking". In: *2008 11th International Conference on Information Fusion*. 2008, pp. 1–6.

[34] Jihua Huang and Han-Shue Tan. "Vehicle future trajectory prediction with a DGPS/INS-based positioning system". In: *2006 American Control Conference*. 2006. DOI: 10.1109/ACC.2006.1657655.

[35] G. R. de Campos et al. "Collision avoidance at intersections: A probabilistic threat-assessment and decision-making system for safety interventions". In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. 2014, pp. 649–654. DOI: 10.1109/ITSC.2014.6957763.

[36] S. Atev, G. Miller, and N. P. Papanikolopoulos. "Clustering of Vehicle Trajectories". In: *IEEE Transactions on Intelligent Transportation Systems* 11.3 (2010), pp. 647–657. DOI: 10.1109/TITS.2010.2048101.

[37] Hiren M. Mandalia and Mandalia Dario D. Salvucci. "Using Support Vector Machines for Lane-Change Detection". In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 49.22 (2005), pp. 1965–1969. DOI: 10.1177/154193120504902217. eprint: https://doi.org/10.1177/154193120504902217. URL: https://doi.org/10.1177/154193120504902217.

[38] Sajjad Mozaffari et al. "Deep Learning-Based Vehicle Behavior Prediction for Autonomous Driving Applications: A Review". In: *IEEE Transactions on Intelligent Transportation Systems* (2020), pp. 1–15. ISSN: 1558-0016. DOI: 10.1109/tits.2020.3012034. URL: http://dx.doi.org/10.1109/TITS.2020.3012034.

[39] Alex Zyner, Stewart Worrall, and Eduardo Nebot. "A Recurrent Neural Network Solution for Predicting Driver Intention at Unsignalized Intersections". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 1759–1764. DOI: 10.1109/LRA.2018.2805314.

[40] Alex Zyner et al. “Long short term memory for driver intent prediction”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. 2017, pp. 1484–1489. DOI: 10.1109/IVS.2017.7995919.

[41] Derek J. Phillips, Tim A. Wheeler, and Mykel J. Kochenderfer. “Generalizable intention prediction of human drivers at intersections”. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. 2017, pp. 1665–1670. DOI: 10.1109/IVS.2017.7995948.

[42] Nachiket Deo and Mohan M. Trivedi. *Convolutional Social Pooling for Vehicle Trajectory Prediction*. 2018. arXiv: 1805.06771 [cs.CV].

[43] Alex Graves. “Long Short-Term Memory”. In: *Supervised Sequence Labelling with Recurrent Neural Networks*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 37–45. ISBN: 978-3-642-24797-2. DOI: 10.1007/978-3-642-24797-2_4. URL: https://doi.org/10.1007/978-3-642-24797-2_4.

[44] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-term Memory”. In: *Neural computation* 9 (Dec. 1997), pp. 1735–80. DOI: 10.1162/neco.1997.9.8.1735.

[45] Nachiket Deo and Mohan M. Trivedi. “Multi-Modal Trajectory Prediction of Surrounding Vehicles with Maneuver based LSTMs”. In: *2018 IEEE Intelligent Vehicles Symposium (IV)* (June 2018). DOI: 10.1109/ivs.2018.8500493. URL: http://dx.doi.org/10.1109/IVS.2018.8500493.

[46] Andrey Rudenko et al. “Human motion trajectory prediction: a survey”. In: *The International Journal of Robotics Research* 39.8 (June 2020), pp. 895–935. ISSN: 1741-3176. DOI: 10.1177/0278364920917446. URL: http://dx.doi.org/10.1177/0278364920917446.

[47] I. Batkovic et al. “A Computationally Efficient Model for Pedestrian Motion Prediction”. In: *2018 European Control Conference (ECC)*. 2018, pp. 374–379. DOI: 10.23919/ECC.2018.8550300.

[48] Dirk Helbing and Péter Molnár. “Social force model for pedestrian dynamics”. In: *Physical Review E* 51.5 (May 1995), pp. 4282–4286. ISSN: 1095-3787. DOI: 10.1103/physreve.51.4282. URL: http://dx.doi.org/10.1103/PhysRevE.51.4282.

[49] X. Yan, I.A. Kakadiaris, and S.K. Shah. “Modeling local behavior for predicting social interactions towards human tracking”. In: *Pattern Recognition* 47.4 (2014), pp. 1626–1641. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2013.10.019. URL: https://www.sciencedirect.com/science/article/pii/S0031320313004482.

[50] A. Alahi et al. “Social LSTM: Human Trajectory Prediction in Crowded Spaces”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 961–971. DOI: 10.1109/CVPR.2016.110.

[51] F. Bartoli et al. “Context-Aware Trajectory Prediction”. In: *2018 24th International Conference on Pattern Recognition (ICPR)*. 2018, pp. 1941–1946. DOI: 10.1109/ICPR.2018.8545447.

[52] J. Lin. "Divergence measures based on the Shannon entropy". In: *IEEE Transactions on Information Theory* 37.1 (1991), pp. 145–151. DOI: 10.1109/18.61115.

[53] Wikipedia. *Jensen-Shannon Divergence*. URL: https://en.wikipedia.org/wiki/Jensen-Shannon_divergence.

[54] Steven L. Brunton and J. Nathan Kutz. *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2019. DOI: 10.1017/9781108380690. URL: http://databookuw.com/databook.pdf.

[55] Alex Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. Vol. 385. Jan. 2012. ISBN: 978-3-642-24796-5. DOI: 10.1007/978-3-642-24797-2.

[56] Christopher Olah. *Understanding LSTM Networks*. 2015. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[57] SUMO. *SUMO Vehicle Type Parameters*. URL: https://sumo.dlr.de/docs/Vehicle_Type_Parameter_Defaults.html.

[58] Florent Altché and Arnaud de La Fortelle. "An LSTM network for highway trajectory prediction". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. 2017, pp. 353–359. DOI: 10.1109/ITSC.2017.8317913.

[59] Long Xin et al. *Intention-aware Long Horizon Trajectory Prediction of Surrounding Vehicles using Dual LSTM Networks*. 2019. arXiv: 1906.02815 [cs.LG].

[60] Tianyang Zhao et al. *Multi-Agent Tensor Fusion for Contextual Trajectory Prediction*. 2019. arXiv: 1904.04776 [cs.CV].

[61] Shengzhe Dai, Li Li, and Zhiheng Li. "Modeling Vehicle Interactions via Modified LSTM Models for Trajectory Prediction". In: *IEEE Access* 7 (2019), pp. 38287–38296. DOI: 10.1109/ACCESS.2019.2907000.

[62] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. "GRIP: Graph-based Interaction-aware Trajectory Prediction". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. 2019, pp. 3960–3966. DOI: 10.1109/ITSC.2019.8917228.

[63] 5GAA. *C-V2X Use Cases and Service Level Requirements Volume 1*. URL: https://5gaa.org/wp-content/uploads/2020/12/5GAA_T-200111_TR_C-V2X_Use_Cases_and_Service_Level_Requirements_Vol_I-V3.pdf.

[64] 5GAA. *C-V2X Use Cases and Service Level Requirements Volume 2*. URL: https://5gaa.org/wp-content/uploads/2021/01/5GAA_T-200116_TR_C-V2X_Use_Cases_and_Service_Level_Requirements_Vol_II_V2.1.pdf.