



Crossing of Road Intersections: Decision-Making Under Uncertainty for Autonomous Vehicles

Mathieu Barbier

► To cite this version:

Mathieu Barbier. Crossing of Road Intersections: Decision-Making Under Uncertainty for Autonomous Vehicles. Artificial Intelligence [cs.AI]. Université Grenoble Alpes, 2019. English. NNT : 2019GREAM058 . tel-02945070v2

HAL Id: tel-02945070

<https://theses.hal.science/tel-02945070v2>

Submitted on 22 Sep 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



THÈSE

Pour obtenir le grade de

DOCTEUR DE LA COMMUNAUTE UNIVERSITÉ GRENOBLE ALPES

Spécialité : Mathématiques et Informatique

Arrêté ministériel : 25 mai 2016

Présentée par

Mathieu BARBIER

Thèse dirigée par **Emmanuel MAZER**, DR

et codirigée par **Christian LAUGIER**, Directeur de recherche,
INRIA

et **Olivier SIMONIN**, INSA LYON

préparée au sein du **Laboratoire Laboratoire d'Informatique de
Grenoble**

dans l'**École Doctorale Mathématiques, Sciences et
technologies de l'information, Informatique**

Traversée d'une intersection : prise de décision sous incertitudes pour les véhicules autonomes

Crossing of Road Intersections: Decision- Making Under Uncertainty for Autonomous Vehicles

Thèse soutenue publiquement le **11 décembre 2019**,
devant le jury composé de :

Monsieur CHRISTIAN LAUGIER

DIRECTEUR DE RECHERCHE EMERITE, INRIA CENTRE DE
GRENOBLE RHÔNE-ALPES, Co-directeur de thèse

Monsieur MARCELO H. ANG JR

PROFESSEUR ASSOCIE, UNIVERSITE NATIONALE DE SINGAPOUR,
Rapporteur

Monsieur FAWZI NASHASHIBI

DIRECTEUR DE RECHERCHE, INRIA CENTRE DE PARIS, Rapporteur

Monsieur JAMES L. CROWLEY

PROFESSEUR, GRENOBLE INP, Président

Madame VERONIQUE BERGE-CHERFAOUI

PROFESSEUR, UNIVERSITE DE TECHNOLOGIE DE COMPIEGNE,
Examineur

Monsieur OLIVIER SIMONIN

PROFESSEUR, INSA LYON, Examineur

Monsieur JAVIER IBANEZ-GUZMAN

DOCTEUR-INGENIEUR, TECHNOCENTRE RENAULT, Guyancourt,
Examineur

Abstract

Autonomous vehicles navigation implies that decisions are taken continuously based on a partial and uncertain knowledge of their environment. This is constrained by limited digital representations of the environment and uncertainty associated with the perception process. Further, it is difficult to predict the behaviour of the perceived entities. This is highlighted at crossroad intersections, where most road accidents occur. This thesis proposes a decision-making process that reasons with different types of uncertainties including the behaviour of the observed drivers to plan the vehicle motion.

To understand the context and the behaviours of the observed driver's behaviour, a machine learning approach is proposed. The result is used by the decision-making process to build probabilistic estimation of the environment. The vehicle motion is planned taking into account the effect this might have when the vehicle interacts with other entities. Our approach rewards actions that promote interaction and reduces risk. The system behaviour is analysed by using a set of metrics derived from the scenario analysis as well as safety and operational constraints.

To infer road context, Gaussian Processes are applied to learn motion patterns from simulated trajectories, which included the effect of vehicle interactions with other entities. The resulting patterns are segmented into areas and used to understand the behaviour of a vehicle approaching an intersection. Then, Random Forest Classifiers are applied to estimate the driver manoeuvre in each area. The dataset used for this training is built using data recorded from road trials and simulations. These classifiers infer lateral and longitudinal manoeuvre by extracting features from the vehicle trajectories. This approach shows that the road context improves the manoeuvre classification and that by mixing few real and simulated trajectories, it is possible to classify the manoeuvres of drivers arriving at an intersection.

The decision-making process is built upon Partially Observable Markov Decision Process and uses the output of the manoeuvre understanding. This probabilistic framework reasons including perception and behaviour uncertainty in the environment models. These are used to predict the likely consequences of the autonomous vehicle actions on its immediate environment. However,

the evaluation of all combinations of actions and state estimations is complex, therefore, an online solver is used to obtain an approximation of each action value. The vehicle actions are evaluated using a set of rewards, namely: collision risk, behavioural risk, comfort and traffic rules. A weighted sum of linear functions is used to balance each component of the reward function with respect to the vehicle distance to the intersection. It allows to adapt the behaviour of the automated vehicle to different scenarios.

To validate the system performances and to determine causes of failure and success, Key Performance Indicators associated to the scenario are proposed. These are part of a generic testing architectures. The approach is applied to cross-cutting scenarios at road intersections, considered very complex and hazardous. Simulation techniques have been used to evaluate the proposed framework, to examine the largest number of scenarios and to be tested in safe conditions.

The result of this thesis shows that it is possible to reason with other vehicle behaviour in the decision-making process while approaching a road intersection crossing. While classical methods fail to evaluate the system behaviour, the proposed validation method gives more insights. It will allow to test the system on the real road as well as using more advance data driven methods in the decision-making models.

Résumé

La navigation pour les véhicules autonomes implique que des décisions soient prises en permanence, sur la base d'une connaissance partielle et incertaine de leur environnement. Ceci est limité par les représentations numériques de l'environnement et par l'incertitude associée au processus de perception. De plus, il est difficile de prédire le comportement des entités perçues, particulièrement aux intersections. Cette thèse propose un processus de prise de décision prenant en compte différents types d'incertitudes, notamment le comportement des conducteurs observés, pour planifier le mouvement du véhicule. Pour comprendre le contexte et les comportements des conducteurs observés, une approche par apprentissage automatique est proposée. Le résultat est utilisé par le processus de prise de décision pour construire une estimation probabiliste de l'environnement. Le mouvement du véhicule est planifié en tenant compte l'effet que celui-ci pourrait avoir lorsque le véhicule interagit avec d'autres entités. Notre approche récompense les actions favorisant l'interaction et réduisant les risques. Le comportement du système est analysé à l'aide d'un ensemble de métriques dérivées de l'analyse du scénario.

Pour déduire le contexte routier, des processus gaussiens apprennent des modèles de mouvement à partir de trajectoires simulées, qui incluent l'effet des interactions entre les véhicules. Les modèles sont segmentés en zones et utilisés pour comprendre le comportement d'un véhicule approchant d'une intersection. Ensuite, des forêts d'arbres décisionnels sont appliqués pour estimer la manœuvre du conducteur dans chaque zone. L'ensemble de données utilisé pour l'entraînement est construit à l'aide de données enregistrées lors d'essais sur route et de simulations. Ces classificateurs infèrent les manœuvres latérales et longitudinales en extrayant des caractéristiques des trajectoires du véhicule. Cette approche montre que le contexte routier améliore la classification des manœuvres et qu'en mélangeant quelques trajectoires réelles et simulées, il est possible de différencier les manœuvres de conducteurs arrivant à une intersection.

Le processus de prise de décision repose sur un processus de décision markovien partiellement observable et les résultats de la compréhension de la manœuvre. Ce cadre probabiliste permet de prendre en compte l'incertitude de la perception et du comportement. Ceux-ci sont utilisés pour

prédire les conséquences probables des actions des véhicules autonomes sur son environnement immédiat. L'évaluation de toutes les combinaisons d'actions et d'estimations d'état est complexe. Par conséquent, un solveur online est utilisé pour obtenir une approximation de la valeur de chaque action. Elles sont évaluées à l'aide d'un ensemble de récompenses, à savoir : risque de collision, risque comportemental, confort et règles de circulation. Une somme pondérée de fonctions linéaires est utilisée pour équilibrer chaque composant de la fonction de récompense par rapport à la distance du véhicule à l'intersection.

Pour valider les performances du système et déterminer les causes d'échec et de succès, des indicateurs de performance associés au scénario sont proposés. Ceux-ci font partie d'une architecture de tests génériques pour la validation de système autonome. Des techniques de simulation ont été utilisées pour évaluer le cadre proposé, pour examiner le plus grand nombre de scénarios et pour être testé sans danger.

Le résultat de cette thèse montre qu'il est possible de raisonner avec le comportement d'autres véhicules dans le processus de prise de décision à l'approche d'une intersection. Alors que les méthodes classiques ne permettent pas d'évaluer le comportement du système, la méthode de validation proposée donne davantage d'informations. Cela permettra de tester le système sur la route ouverte et d'analyser plus en profondeur des méthodes basées sur l'intelligence artificielle.

Acknowledgement

Getting to finish this thesis was a long and hard work which, without the support of a lot people, would have not been possible. First and foremost, I would like to express my gratitude to my supervisors Javier Ibanez-Guzman, Christian Laugier and Olivier Simonin for guiding me along this journey, but also for having shared with me their knowledge and passion. I would like to thank Renault for financing this thesis and for offering me a position to continue my research alongside the same team. There is also all the people who work there - Emmanuel, Alexandre, Clement, Vicente, Fady, Lucille, Arun, David, You. It takes a lot of people to make a vehicle move, thank you all for your help. There are also all the people from INRIA with whom I shared coffee and worked with: Jean-Alix, Lucas, Jerome, Amaury, Dominique, Mario, Victor, Thomas, Alessandro.

I was not alone in experiencing the highs and lows of working on a thesis, Yrvann, Edouard, Matthieu Frank, Ludo. I am really happy that we got through this together. More specifically I would like to thank Pavan and David for all the good times we had hiking, travelling, exploding kittens and working out.

To my familly and friend, thank you for always being there and supportive.

I would not be here if the woman who is now my wife, Danxi, would not have been so patient and helpful during these years. Thank you so much.

Contents

1	Introduction	12
1.1	Autonomous Vehicles: Social and Financial Rational	13
1.2	Road Intersections Crossing	17
1.2.1	Crossing at Road Intersection Compared to Other Scenario	17
1.2.2	Accident Statistics at Road Intersections	19
1.2.3	Decision-Making for Crossing at Road Intersection	22
1.3	Problem Formulation	25
1.3.1	Problem Statement	25
1.3.2	Thesis Approach	26
1.3.3	Contributions	28
1.4	Thesis Content	29
2	Related Work on Behaviour Estimation and Decision-Making for Road Intersection Crossing	31
2.1	Introduction	31
2.2	Context Formulation: The Navigation Function In Intelligent Vehicles	32
2.2.1	Classification of Intelligent Vehicle of Architectures	32
2.2.1.1	Vehicle Centric	33
2.2.1.2	Driver Centric	34
2.2.1.3	Network Centric	35
2.2.2	Discussion	36
2.2.3	Decision-making for Vehicle Centric Architectures	37
2.2.4	Discussion	43
2.3	Behaviour Understanding	43
2.3.1	Definitions	44
2.3.2	Methods for Behaviour Understanding at Road Intersections	47
2.3.2.1	Classification	49
2.3.2.2	Probabilistic Graphs	50

Contents

2.3.3	Discussion	52
2.4	Approaches to Decision-Making at Road Intersection	53
2.4.1	Rule-based	54
2.4.2	Probabilistic Approach: Partially Observable Markov Decision Process . .	57
2.4.3	Machine Learning	60
2.4.4	Discussion	64
2.5	Conclusion	65
3	Manoeuvre Classification Applying Learned Functional Discretisation	67
3.1	Introduction	67
3.2	Functional Discretisation	68
3.2.1	Motion Patterns With Gaussian Processes	71
3.2.2	Functional Discretization Framework	74
3.2.2.1	Dataset Acquisition	74
3.2.2.2	Gaussian Process Training	80
3.2.2.3	Discretization of a Merging and Overlapping Trajectory	82
3.2.2.4	Discretization of an Approaching Trajectory	82
3.2.3	Discretization Results	83
3.2.4	HD-Map Storage	85
3.3	Manoeuvre Classification	86
3.3.1	Approach	87
3.3.2	Random Forest Classifier	89
3.3.3	Dataset	91
3.3.3.1	Real Environments	91
3.3.3.2	Simulated Environments	92
3.3.3.3	Hybrid Dataset	92
3.3.3.4	Features	93
3.4	Results	94
3.4.1	Random Forest Topology	96
3.4.2	Functional Discretization Compared to Rectangular Discretization	98
3.4.3	Comparison With Baseline Approach	99
3.4.4	Results With the Hybrid Dataset	100
3.4.5	Discussion	101
3.5	Conclusion	104

4	Probabilistic Decision-Making, Applied to Crossroad Intersections	106
4.1	Introduction	106
4.2	Context and Integration into Vehicle Centric Architectures	107
4.2.1	Road Intersection Crossing Scenario	107
4.2.2	Integration of the Probabilistic Decision-Making into the AV Architecture	110
4.3	Decision-Making Framework	112
4.3.1	State-Space	114
4.3.2	Action-Space	114
4.3.3	Transition Model	116
4.3.4	Observation-Space	121
4.3.5	Observation Model	121
4.3.6	Reward Function	123
4.4	Partially Observable Monte-Carlo Planning: An Online POMDP Solver	127
4.4.1	POMDP Solver	127
4.4.2	POMCP Elements	129
4.4.2.1	Monte-Carlo Search Tree	129
4.4.2.2	Upper Confidence Bound	130
4.4.3	POMCP Algorithm	131
4.5	Conclusion	134
5	Implementation, Tests and Analysis	136
5.1	Introduction	136
5.2	Testing Framework	138
5.2.1	Testing Context	138
5.2.2	Testing Architecture	140
5.3	Key Performance Indicators to Evaluate Decision-Making System	145
5.3.1	Approach	146
5.3.2	Safety KPI	148
5.3.3	Trust KPI	148
5.3.4	Comfort KPI	150
5.3.5	Navigation KPI	150
5.3.6	Perspectives on Key Performance Indicators	151
5.4	Experiments and Results	152
5.4.1	Parameters Exploration	152
5.4.2	Behaviour Examples	154
5.4.3	Evaluation of the Decision-Making Performance.	158

Contents

5.4.4	Statistical Model Checking	160
5.5	Conclusion	164
6	Conclusion	165
6.1	Summary of Findings	165
6.2	Conclusions	167
6.3	Perspectives	168
6.3.1	Behaviour Understanding	169
6.3.2	Probabilistic Decision-making	169
6.3.3	Validation	170
	Bibliography	172

1 Introduction

1.1	Autonomous Vehicles: Social and Financial Rational	13
1.2	Road Intersections Crossing	17
1.2.1	Crossing at Road Intersection Compared to Other Scenario	17
1.2.2	Accident Statistics at Road Intersections	19
1.2.3	Decision-Making for Crossing at Road Intersection	22
1.3	Problem Formulation	25
1.3.1	Problem Statement	25
1.3.2	Thesis Approach	26
1.3.3	Contributions	28
1.4	Thesis Content	29

Decision-Making is one of the most challenging tasks in autonomous driving. Vehicles need to adapt their behaviours in real-time based on their situation understanding. The difficulties in building and understanding this digital representation of the immediate environment come from the uncertainties associated with the perception and the behaviour estimation of the entities sharing the same road network.

The focus of this thesis is decision-making, as applied to one of the most complex road network segments, road intersections. The convergence of multiple mobile platforms and vulnerable entities onto a single road segment results in multiple collisions, due to the complexities of the situation understanding. In this chapter, the rationale for the deployment of autonomous vehicle is formulated and the decision for selecting the crossing of road intersections as the focus of this research is provided. The decision-making problem is then formulated. It includes the thesis scope. Finally, the contributions are listed as well as the organization of the thesis outlined.

1.1 Autonomous Vehicles: Social and Financial Rational

The paradigms for human mobility have changed during the past 10 years. Whilst in post-industrial countries, the need for personal vehicles is decreasing due to the accessibility to public transport and the increasing cost of owning a personal vehicle. In other countries, like China, Brazil or India, vehicles are still seen as a sign of a higher social status, thus there is a growing demand for firsthand vehicles. Everywhere, road networks are insufficient to support the growing number of vehicles and congestions costs annually \$1 trillion worldwide [1]. The impact of road transportation on people's health and the environment is taking a central stage on today's political agenda worldwide.

To respond to these challenges, a first transformation started within the automotive industry with the electrification of passenger vehicles. It was supported by governments, giving incentive to help customers to change polluting vehicles for cleaner ones. It was a small success for companies like Renault-Nissan, BMW that started to propose electric vehicles (e.g. leaf, BMW i3, Zoé...). New initiatives from companies, such as Volvo, aim to propose only electric/hybrid vehicles from 2019 [2]. However, these are insufficient to solve pollution problems.

Another major challenge is to reduce the number of accidents. The target fixed by the World Health Organization (WHO), and followed by the European union, is to halve the number of accidents by 2020 [3]. However, as illustrated in Figure 1.1, a plateau was reached in 2013. The predicted impact of Advance Driving Assistant System (ADAS) and autonomous vehicle technologies shall reduce the number of accidents. However, ADAS systems are mostly bought by the middle-age population (because of their cost) and does not help to accidents that are mainly caused by young and old people [4]. Statistics show that most road fatalities are due to human errors [3]. Therefore, removing the human from the vehicle control loop could result in safety improvements.

1 Introduction

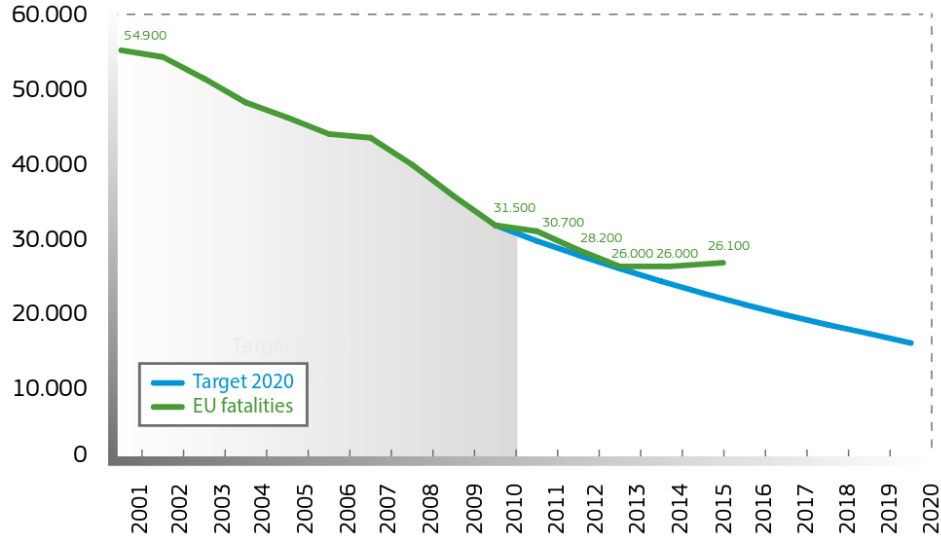


Figure 1.1 Road fatalities within the EU since 2001, after [3]

Research on autonomous vehicle came to a well-known success in the late eighties. E.Dickmans, as part of the Prometheus project, had a passenger vehicle driving autonomously on a motorway near Paris in 1994 [5]. Progress in autonomous driving could be measured by the following milestones: The DARPA Grand Challenge in 2005, in which for the first-time autonomous passenger unmanned vehicles travelled more than 100 km in the desert [6]. The DARPA Urban Grand Challenge, where in 2007 several unmanned vehicles operated in a mock urban area [7], in Figure 1.2a. Finally, the obtaining of a license plate in California (USA) for the Google car in 2012 after 3 years of development and tests on open road [8], in Figure 1.2b. The whole sent a clear message to industry on the potential of autonomous vehicles. Since that date global interest on this domain has exploded.

1 Introduction



Figure 1.2 (a) Boss vehicle, Carnegie Mellon University at DARPA Urban challenge November 2007 [7]. (b) Steve Mahan the first legally blind person to ride in Google's autonomous car, March 2012 [9]

There is a perceptible acceleration in this domain due to the strong interest by technological companies like Google or mobility providers like Uber and Lyft. They are developing autonomous vehicles technologies with substantial capital investments. In addition, there is the emergence of a new generation of start-ups developing the related technologies. The automotive industry seeks today to become mobility providers rather than vehicle suppliers and thus change their business model [10]. A new industrial segment is emerging around the domain of autonomous vehicle technologies to provide software, computing hardware, and sensors.

The benefit of autonomous driving for the public means that more time will be available for work or leisure inside a vehicle. Therefore, longer commutes could be more acceptable. The provision of land transportation services for those which cannot drive like the young, elderly and disabled will increase their life quality. For professional, delivery truck drivers could be removed, reducing prices and likely increasing goods transportation efficiency.

To classify autonomous vehicle systems (AVS), the Society of Automotive Engineers (SAE) proposed 5 different levels of autonomy [11]. Level 2 with a control of the longitudinal and lateral motion but with the situational awareness capabilities monitored by a human driver is available on today's high-end vehicles (Tesla, Audi). The system complexity emerges from the interaction between the car and driver, because the system must ensure that the driver can respond adequately when required. A level 3 prototype has the situation understanding and car decisions handled by vehicle systems. The driver is in charge to monitor the system and, in case of disengagement or dangerous situation, corrects the vehicle trajectory. This type of prototype might never be distributed for public uses because regular driver might over trust the system

1 Introduction

or be distracted. The recent accident in Arizona between a Uber vehicle and a pedestrian, has shown the gravity of resulting accident [12].

In an early stage of development, Google/Waymo has identified the problem of human drivers to be unsolvable and aimed for level 4 AVS. It implies that under nominal condition (good weather, known area) the vehicle is able to navigate without human intervention. Waymo successfully demonstrated autonomous ride-sharing system without a safety driver in Austin (USA) in November 2017 [13]. Despite the 7 million of kilometres driven in public road and billions in simulated environment, they are still reluctant to commercialize an unmanned vehicle related service [14]. Several OEMs have endorsed the same path and are now focusing on this level. The large ongoing effort without commercially available service demonstrates that remains several issues to solve before full deployment. A level 5 vehicle would be able to drive under adversarial condition such as weather conditions and would require new methods of sensing.

Beyond the above-mentioned technical challenges, additional issues also have to be addressed:

- Cost for hardware and sensors needed to be installed in a vehicle to become autonomous are very high. This price is above the average amount people are willing to pay, \$4900, for full automation in a passenger car [15]. The industrialization of certain processes might reduce cost. However, there are still sensors to be designed, software to be developed, etc to reach higher level of autonomy.
- Validation and verification for the automotive industry are subject to strict norms. Each of autonomous systems added to the vehicle should to be tested and validated under procedures yet to defined [16]. The domain of Automated Cyber-Physical system (ACPS), not only in the automotive industry, lacks validation methods when it comes to machine learning algorithms and non-deterministic approaches.
- Social acceptability, it has been estimated that such technologies will directly affect the work of thousands of people (taxi, Uber, truck drivers, etc) as well as other domains like real estate, energy supply, etc. With no humans to supervise the decision of the vehicle, the question of responsibility in case of accident is complex. Accidents might continue to happen, therefore governments, insurers and other stack holders shall discuss responsibility implication

The consequences of AVS on current transportation system has been studied and they are estimated to change our way of life. Preliminary results, have shown that multiple technical challenges remain, and some situation are more complex than others.

1.2 Road Intersections Crossing

Road intersections represent one of the most difficult segments in road networks. It is the crossing of different traffic flows that at a certain point could converge into a collision if appropriate decisions are not taken. Traffic refers not only to powered vehicles or two-wheeler but also to the presence of pedestrians. Each agent at an intersection can be considered as having different behaviour in many cases unmeasurable which makes situations very complex. Further, there are the issues of temporary occlusions, weather conditions, poor lighting conditions and the violation of the traffic rules for some parties. In addition, when crossing an intersection, drivers negotiate through hidden gesture or behaviours. For autonomous vehicles to be deployed, they need to cross road intersection safely if they are to be accepted into current traffic conditions.

1.2.1 Crossing at Road Intersection Compared to Other Scenario

The different road situations that an autonomous vehicle can encounter can be classified into five major groups as shown in Figure 1.3.

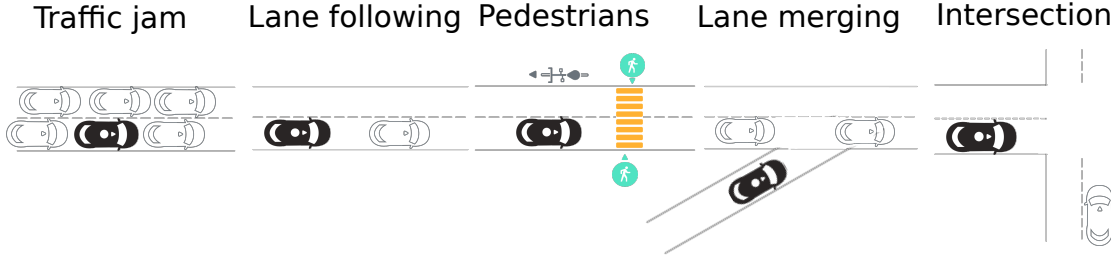


Figure 1.3 Classes of driving situation, an autonomous vehicle (in black) can encounter (assets from [17])

A summary description of this classification and related complexity is given by:

- A) **Traffic jam**, the velocity of the subject vehicle is small as for other vehicles. Consequently, their behaviour is to remain in their lane or to wait for the traffic jam to end. Some drivers might try to change lane, but collisions are mitigated thanks to low speed.

1 Introduction

- B) **Lane following and highway driving** are complex because of the relatively high speed of vehicles in the scene and the potential severity of resulting accidents. The set of possible behaviours for each vehicle is relatively small, the vehicle has to stay within their lane or to change lane if the traffic ahead is too slow.
- C) **Driving in the presence of pedestrians**, that are perceived as the most vulnerable road users, in particular children and elderly. Pedestrians can have an erratic behaviour thus driving next to them is difficult. At intersections, pedestrians can be occluded, can cross at the last minute, etc. For the perception systems, this is a challenge. Autonomous vehicles will slow down as they are next to pedestrians
- D) **To merge with traffic**, the subject vehicle must estimate the gap between two vehicles. It is common for human driver to engage a merging with a small gap and to expect the other vehicle to allow the merging. This behaviour estimation is complex as drivers might not be willing to cooperate, resulting in different manoeuvres.
- E) **Road intersections**, the subject vehicle is left with different manoeuvres (stop, yield, cross, turn left/right), that will affect the behaviour of surrounding vehicle. Further, some vehicles might be occluded or their behaviour variable, leaving the autonomous vehicle to decide and adapt constantly.

The most difficult situation arises when the situation complexity and uncertainty are high. This is the crossing of road intersection as shown in Figure 1.4.

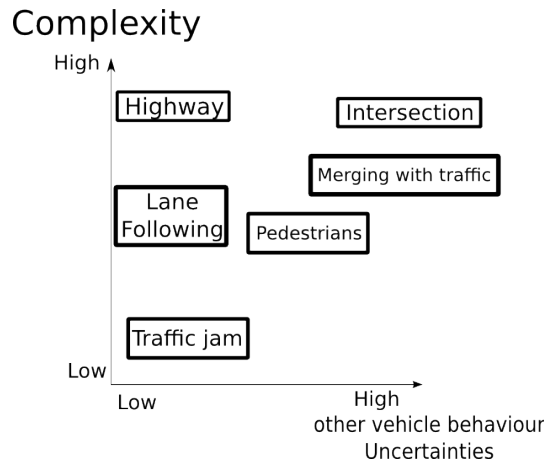


Figure 1.4 Road situations complexity and uncertainty

The figure 1.5 shows the different variables that create difficulty at an intersection. The most

1 Introduction

often studied change includes variation of traffic sign and the presence of other vehicles. The presence of pedestrian or two wheeled vehicles can make scenarios very complex. It is difficult to design a situation understanding and a decision-making system capable to address such complex situations.

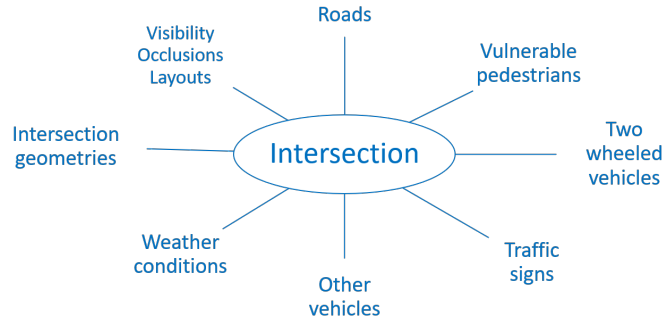


Figure 1.5 Element and entities providing variety to the intersection use case

1.2.2 Accident Statistics at Road Intersections

Road injury is the 8th cause of death across the world population, thus governmental and non-governmental organizations keep track of accident reports to raise awareness and guide projects.

In the EU almost 20% of fatal accidents occur at road junctions [18], as shown in Figure 1.6. For the purpose of the thesis, three types of intersection have been identified: T-junctions, crossroads and roundabouts. With new legislation and new vehicle safety device, the number of fatal accidents at junctions has been reduced by 40% since 2006 to reach 5000 in 2015.

1 Introduction

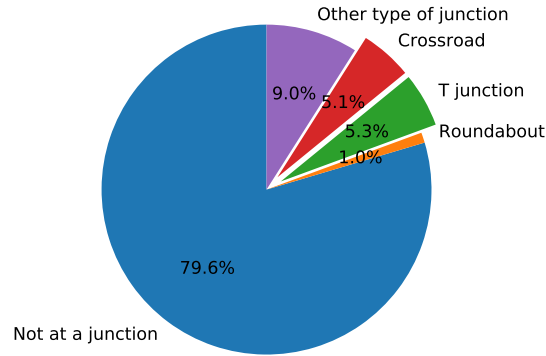


Figure 1.6 Percentage of fatalities by type of junction in the EU, after [18]

There is a geographic disparity with respect to these accidents. For example, in the UK, junctions related accidents represented a third of total road fatalities, whereas in France, they accounted for 14%. Further, 10% of road-fatalities, were at junctions outside urban areas, whilst in urban area they were twice as much.

The CARE project records all accidents across the EU. It provides information and statistics that lead to a better understanding of accidents at road intersections [19]. It shows that interactions between vulnerable users and vehicles are an aggravating cause of fatalities at intersections, that is 60% of fatal accidents occur with vulnerable road users (pedestrians, motorcycle, bicycle, etc). The weather and night conditions are not an aggravating factor of accident at junctions compared to other locations.

Observing the critical event that lead to a collision, the most frequent problem is timing: drivers did not perform any action (24%), had a premature action (21%) or acted too late (14%), further details are found in Figure 1.7. The two main causes of this critical event are a faulty understanding of the situation and/or a missed observation. It shows that humans are likely to behave differently to what can be expected regarding the context, resulting in dangerous situations. Thus, human errors are likely at the origin of most collisions at road intersections.

1 Introduction

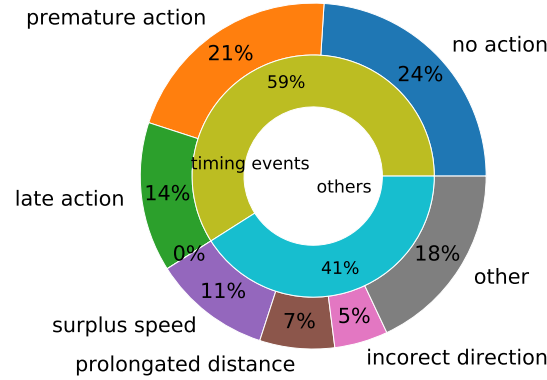


Figure 1.7 Distribution of critical events at road junctions, after [18]

The layout of a typical crossroad intersection results in 24 likely collision points. Road designers introduced roundabout which reduce them to 4, as shown in Figure 1.8. From a safety perspective, roundabouts reduce side collision that represent the most mortal situation when roundabouts did not exist [20]. However, drivers have much difficulties on understanding situation when entering roundabouts.

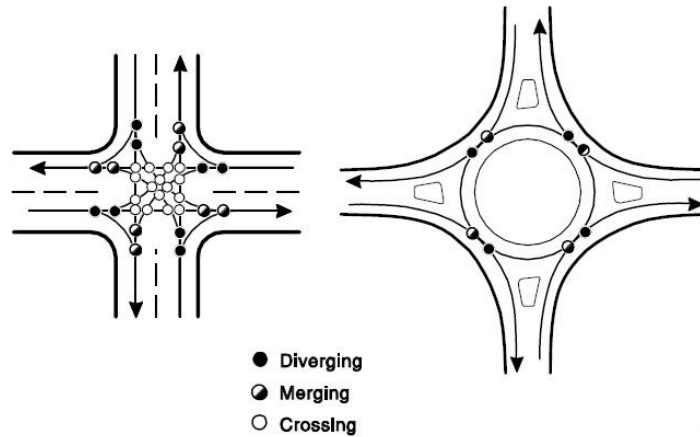


Figure 1.8 Vehicle likely collision points comparison at road intersection and roundabout, after [21]

Even if no statistics exist for autonomous vehicle, they have been often involved in accident

1 Introduction

at crossroads. The first collision between two autonomous vehicles happened at the Urban DARPA challenge of 2007 [22]. The accident, illustrated in Figure 1.9, was due to the vehicle Skynet stopping caused by incoherent data association between the map and perception. At the same time, the Talos vehicle proceeded to overtake the immobilized Skynet, entering its blind spots. After Skynet finally found a path to quit the intersection, it collided with Talos that expected Skynet to remain static. In this example, sensing uncertainties played a key role in the collision. During the event, there were also 5 reported close calls or collisions that occurred at road intersections.

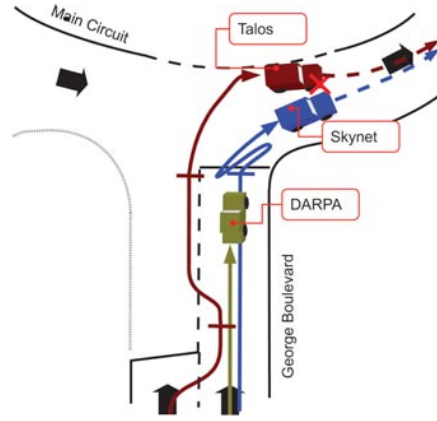


Figure 1.9 Collision scenario between Talos and Skynet during the urban DARPA challenge 2007, after [22]

Autonomous vehicles functionalities have been tested on public roads since 2010. The Tesla accident (May 2016) was due to driver inattention whilst the autopilot did not detect the truck ahead crossing lanes [23]. At other instances, collisions happened but the autonomous vehicle was not at fault [24]. For example, in September 2016 a Lexus from Waymo was involved in a broadside collision, caused by another driver crossing a red light. In March 2017, a driver failed to yield to the Uber vehicle at an intersection. The example shows that, even if the behaviour of the autonomous vehicle is correct, the understanding other vehicle intentions is key to safe road intersection crossing.

1.2.3 Decision-Making for Crossing at Road Intersection

Interactions between an intelligent agent and the environment are made of three functions: perception, decision and control, as shown in Figure 1.10. Decision-making is the most complex

1 Introduction

as it depends of the performance and error of the other system. For both humans and autonomous systems, wrong decisions could lead to accidents.

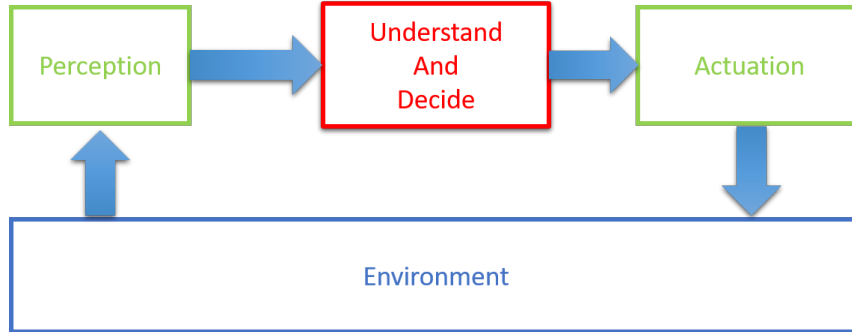


Figure 1.10 Interactions between an intelligent agent and the environment

At road intersection, the first decision taken is the direction that will achieve the driver's destination. This decision takes the form of a route, a sequence of road, that is chosen from map information and traffic monitoring to reach an objective considering travel time, price, pollution exposition, ... From this route the position of intersections can be found.

At the local level of the intersection, the vehicle must decide on its trajectory to cross the intersection. This trajectory is composed of the path, a collection of position, and associated temporal information (speed, or time). The path is found from the lateral manoeuvre that decide if it is necessary to move away from the current lane. The temporal component of trajectory is computed after deciding on the longitudinal manoeuvre, that is at road intersection: to stop, to cross or to yield.

Furthermore, both manoeuvres are chosen based on the understanding of the environment surrounding the vehicle. Several entities interact at an intersection: vehicles, pedestrians, traffic infrastructure, etc. These entities affect the autonomous vehicle manoeuvre, whilst its manoeuvre also affects the environment. Thus, decisions must be constantly updated to react appropriately to these interactions. Some manoeuvre and interaction are undesirable, for example two vehicles interacting together and attempting to enter the intersection is undesirable. Therefore, the autonomous vehicle shall iteratively adapt its trajectory or change manoeuvre.

Perception systems are imperfect, as a result a degree of uncertainty arises. This affects the understanding of the situation, thus the decision-making. The effect of decisions of the subject vehicle on itself is difficult to predict because of the vehicle dynamic response. Further, the

1 Introduction

predicted effect of an action on other environment elements is also difficult to predict because of interactions.

Sources of uncertainties that challenge decision-making are shown in Figure 1.11. These sources can be divided in four categories [25]:

1. Configuration sensing affects the estimation the system has on its own state. It is caused by proprioceptive sensor noise.
2. Configuration predictability comes from the difficulty to estimate actuators responses and affect the estimation of future vehicle states.
3. Environment sensing is subject to sensor noise and environmental condition. It affects how well the real scene is estimated.
4. Environment predictability accounts for interaction and other vehicle behaviour that makes future situation difficult to predict.

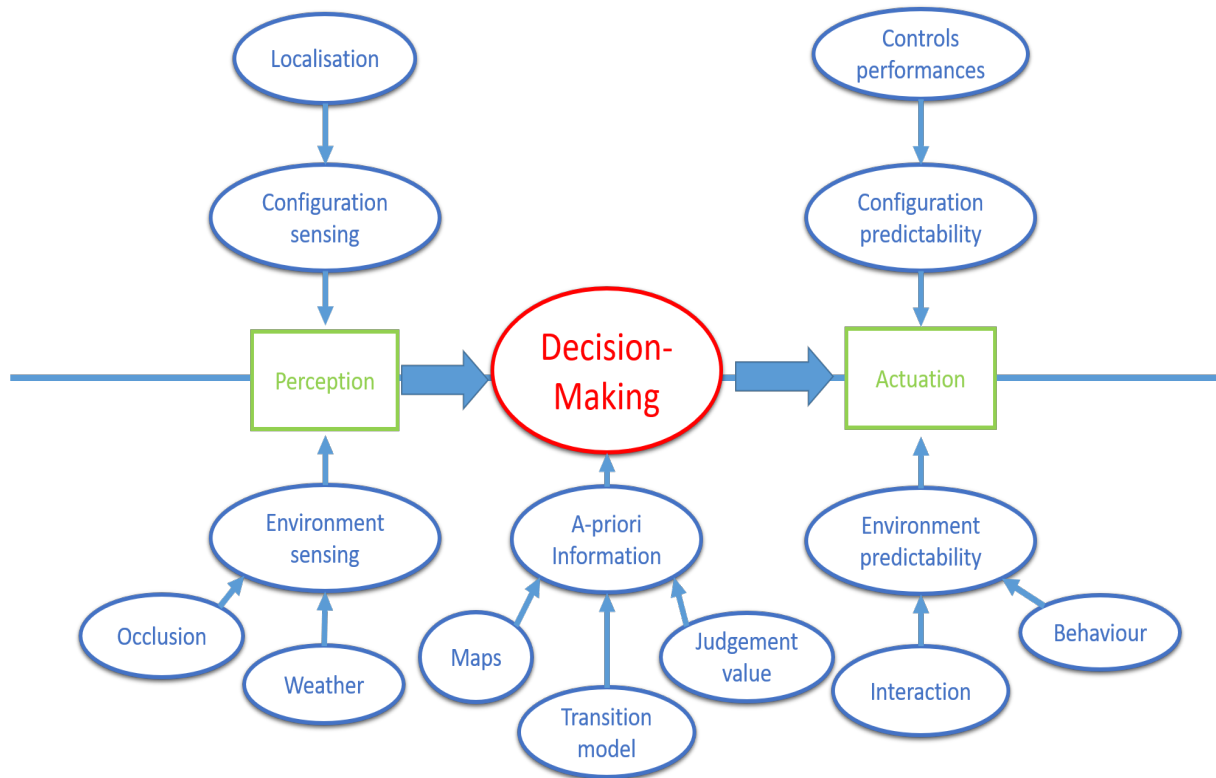


Figure 1.11 Sources of uncertainties and information that challenge the decision-making system

1 Introduction

The decision-making system relies on a priori information to facilitate the selection process. From maps the priority regulation or the intersection layout can be known in advance. Learned models are used to enhance the understanding available on the situation (e.g. to infer other drivers behaviour).

1.3 Problem Formulation

1.3.1 Problem Statement

The major task for an autonomous vehicle is to interact with its environment. The resulting behaviour leads to whether such vehicles can be socially acceptable in terms of performance, safety, understandable for other road user, etc. This is the product of the decision taken by the vehicle as it navigates to its destination.

The decision-making process takes place in real-time through the observation and understanding of an incomplete and uncertain digital representation of the environment (world modelling). There is uncertainty associated with the environment that results from sensor physics and algorithms constraints. In addition, there is uncertainty with respect to the environment predictability caused by the effect of a vehicle actions and the behaviour of other interactive agents. Combined with the uncertainty associated with the subject vehicle configuration, the ensemble of possible futures is complex to explore. Consequently, only an estimated value of a decision is available for the selection. The problem resides on how the expected value of a decision can be estimated so the corresponding vehicle actions can be chosen safely.

Determining whether a solution to the decision-making problem is also complex. If one limits to crossing only road intersections, the scenario space rapidly becomes very large. Testing with physical vehicles becomes impossible, there is no repeatably and even setting simple interaction requires multiple resources. Therefore, means to assess such decision-making system are needed. The problem arises on the manner that decision-making system can be evaluated to provide trust on the system under test.

The problem addressed in this thesis centres on the manner an autonomous vehicle decides taken into account contextual information and uncertainties associated with the vehicle behaviour and its operational environment. A consequence of the stochastic nature associated with the uncertainty is that the performance analysis of the resulting decision-making system must include the non-deterministic result of the approach.

1 Introduction

The purpose is that current decision-making approaches fail to reason with uncertain information, thus their behaviour might not be socially acceptable or understandable to pedestrians, drivers and passengers. Furthermore in such condition, it is indeterminate if there exists a single optimal policy that satisfies all the requirement for the driving task. Thus, the system has to select a solution amongst a set of valid policies. The road intersections crossing use-case offers complex and challenging scenarios for autonomous vehicles that requires the interpretation of driver's intentions and to deal with complex interactions.

The objectives were first to investigate potential mathematical formulation of decision-making that can be applied to autonomous vehicle navigation system and more specifically how they were used at road intersection crossing. Second, to propose a novel method for driver's behaviour understanding dependent of the environment context. Thirdly, to design a decision-making system that can use the driver's behaviour information and deals with the many uncertainties. Emerging from the difficulty to evaluate the performances of the system, the last objective was to find a systematic way to evaluate the performances of the proposed solutions.

1.3.2 Thesis Approach

The approach undertaken follows the industrial objective to obtain practical implementation of the developed solution and the scientific objective to find solution for decision-making under uncertainties. The first step was to determine and formulate the road intersection use case to be used as the focus of the development in terms of the decision-making, solution, experimental basis and evaluation. It includes an analysis from the literature with regards to accidentology data and its complexity. This is centred on the crossing of the intersection by the subject vehicle (SV) whilst considering the likely presence of Other Vehicle (OV) crossing or stopping perpendicular to SV.

The second step consisted in a recollection of field experience data from crossing road intersections applying driving assistance functions and autonomous behaviour amongst Renault engineers as well as preliminary experiences. The purpose was to understand the physical constraints related to vehicle systems. They also provided some insight on perception, localisation and map systems.

From there a concurrent literature review was done on the approaches used for decisions-making applied to intelligent vehicles and behaviours estimation. Out this study a focus was made on the theory associated with a probabilistic approach in order to determine the techniques that includes uncertainty and incorporate it to the decision-making process. The results led to model

1 Introduction

the system under study based on Partially Observable Markov Decision Process (POMDP). To represent the uncertainty associated with the OV, machine learning algorithms are used to estimate its behaviours. The proposed solution is then formulated as a decision-making framework together with its underlying theory.

Figure 1.12 shows an overview of the framework as part of an autonomous system vehicle architecture to provide context.

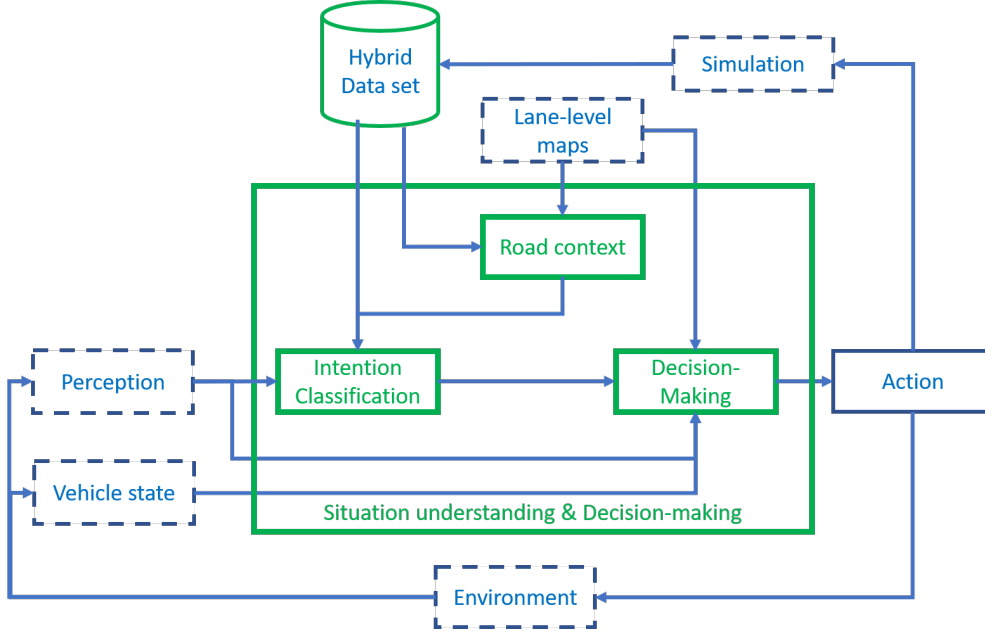


Figure 1.12 The situation understanding and decision-making framework within the context of the autonomous vehicle, in green elements with the thesis contribution

The solution pertinence needs to be validated. As autonomous vehicle are complex systems their validation is difficult and remains an open question. In an effort to address this issue, a systematic approach for testing was developed. It resulted in the definition and selection of Key Performance Indicators (KPI) and the usage of statistical model checking to get more insight on the performances of the system in the road intersection use case. The figure 1.13 shows elements of the evaluation framework and their interaction with the system under test.

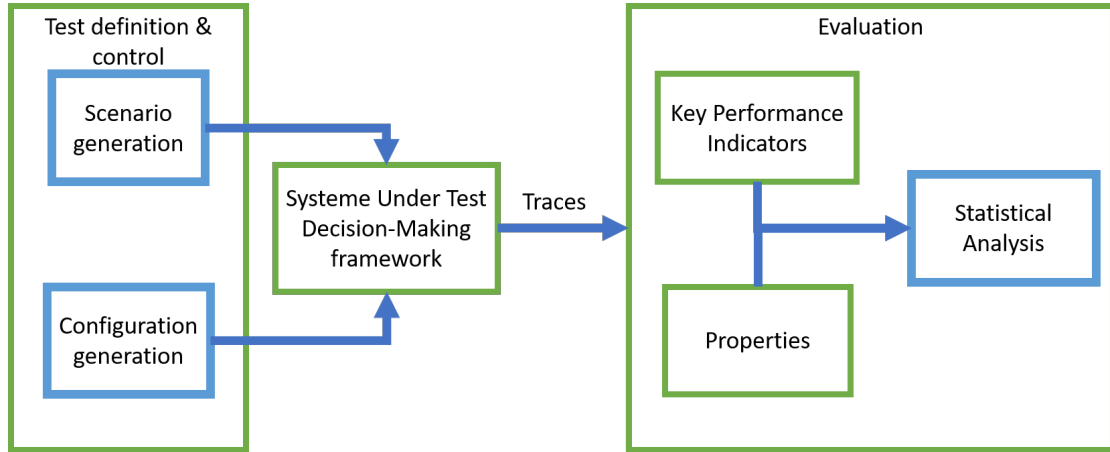


Figure 1.13 Evaluation pipeline for the decision-making framework that is the system under test. The test generation part allows an exploration of the scenario space and reconfiguration of the system. After multiple experiments traces are analysed.

1.3.3 Contributions

The basis of the proposed solution resides on a probabilistic decision-making framework applicable to road intersection crossing. It provides a new approach for behaviour understanding. Methods from other domains has been used to understand the performances of the decision-making framework. Contributions are as follows:

- A framework to build a functional discretization of the driving space is presented in chapter 3.2 [26]. It is based on the use of Gaussian processes to learn the motion patterns. It can be applicable to any type of road intersection. The compatibility of this representation with respect to lane level maps used in Renault’s vehicles has been shown.
- The inference of driver’s intention approaching an intersection is presented in chapter 3 [27]. It classifies the lateral and longitudinal behaviour from features of the driver’s trajectory. A hybrid dataset including simulated and real trajectories was used for training purpose.
- A decision-making framework for road intersection crossing that includes reasoning with configuration and predictability uncertainties is presented in chapter 4 . It uses high-level behaviour observations from the lateral and longitudinal classifier. It adapts the vehicle velocity with respect to various driving criteria (e.g. comfort, risk...).
- A framework to test decision-making systems. It includes the formulation of Key Performance Indicators to interpret the system performance.

1 Introduction

These works resulted in 4 publications at international conferences and 1 presentation in a workshop:

- M. Barbier, C. Laugier, O. Simonin, and J. Ibañez-Guzmán, “Functional discretization of space using Gaussian processes for road intersection crossing”, in 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), 2016, pp. 156–162.
- M. Barbier, C. Laugier, O. Simonin, and J. Ibañez-Guzmán, “Classification of drivers manoeuvre for road intersection crossing with synthetic and real data”, in 2017 IEEE Intelligent Vehicles Symposium (IV), 2017, pp. 224–230.
- M. Barbier, C. Laugier, O. Simonin, and J. Ibañez-Guzmán, “Probabilistic Decision-Making at Road Intersections: Formulation and Quantitative Evaluation”, in 2018 International Conference on Control, Automation, Robotics and Vision.
- M. Barbier, J. Quilbeuf, L. Rummelhard, A. Renzaglia, A. Paigwar, C. Laugier, O. Simonin, A. Legay, and J. Ibañez-Guzmán, “Validation of Perception and Decision-Making Systems for Autonomous Driving via Statistical Model Checking”, in 2019 IEEE Intelligent Vehicles Symposium (IV), 2019.
- J. Quilbeuf, M. Barbier, L. Rummelhard, C. Laugier, A. Legay, B. Baudouin, T. Genevois, O. Simonin and J. Ibañez-Guzmán, “Validation of Perception and Decision-Making Methods for Autonomous Driving using Statistical Model Checking”, in 10th Workshop on Planning, Perception and Navigation for Intelligent Vehicles at IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, October 2018, Madrid, Spain

1.4 Thesis Content

The remaining of the thesis comprises in six chapters:

Chapter 2 presents related work on decision-making applied to the crossing of road intersection, the first section reviews approaches for behaviour estimation and the importance of the road context. The second section presents classical approaches and key advancements on decision-making at road intersections. Emphasis is made on probabilistic approaches because of their practical representation of uncertainty. The last section discusses the chosen approach with respect of the state of the arts and provides the rationale.

1 Introduction

Chapter 3 presents a manoeuvre classification framework built upon a functional discretization. The first section presents principle for learning motion pattern with Gaussian process. Their application to discretise the space of road intersections is then presented. The second section presents the use and advantage of a random forest classifier for manoeuvre classification. To overcome the need for large data, we show the advantage of using a hybrid dataset containing simulated and real data to train the classifier. To conclude this chapter a quantitative analysis of the overall framework is made against baseline classification methods and other types of space discretization.

Chapter 4 presents the decision-making process considering manoeuvre uncertainty. In the first section, the integration of a Partially Observable Markov Decision Making process in a vehicle centric architecture is discussed. The second section describes each element of the Partially observable decision process and their model chosen is discussed. To conclude, the online solver used for the application is presented.

Chapter 5 presents a unique evaluation approach applied to validate the proposed decision-making framework. The detail of the testing framework is detailed. Key Performance Indicators developed to quantify evaluated the proposed approach are detailed. Results obtained from a large number of simulations are discussed. We also applied Statistical Model Checking on our result and we show how it helps to understand the result of our approach.

Chapter 6 concludes the thesis by summarizing the contribution, includes a critique on the findings as well as a description of future work.

2 Related Work on Behaviour Estimation and Decision-Making for Road Intersection Crossing

2.1	Introduction	31
2.2	Context Formulation: The Navigation Function In Intelligent Vehicles	32
2.2.1	Classification of Intelligent Vehicle of Architectures	32
2.2.2	Discussion	36
2.2.3	Decision-making for Vehicle Centric Architectures	37
2.2.4	Discussion	43
2.3	Behaviour Understanding	43
2.3.1	Definitions	44
2.3.2	Methods for Behaviour Understanding at Road Intersections	47
2.3.3	Discussion	52
2.4	Approaches to Decision-Making at Road Intersection	53
2.4.1	Rule-based	54
2.4.2	Probabilistic Approach: Partially Observable Markov Decision Process	57
2.4.3	Machine Learning	60
2.4.4	Discussion	64
2.5	Conclusion	65

2.1 Introduction

The crossing of a road intersection starts with the behaviour estimation the perceived entities, and then deciding what actions to take. The approaches taken for these functions depend on

the manner vehicles are controlled. Whilst intelligent vehicle can be designed to be controlled only by a computer or to benefit of communication with other vehicles and the infrastructure, situation understanding and navigation system are similar.

In this Chapter, the navigation architecture for an intelligent vehicle is presented in the first section, then the driver's behaviour estimation to decide to cross an intersection is studied. This is followed by a study of the decision-making process with regards to the functional architecture of an intelligent vehicle.

In order to decide, the system requires to build a digital representation of the environment to understand its situation. At a road intersection, it is necessary to understand what other road users intend to do, and possible interactions. This is possible applying probabilistic graphs and classification methods. The former uses streams of measurements to infer the driver's behaviour. The latter uses features from the measurement stream and its similarity with previous knowledge. The second section discussed these approaches and provides the rationale for choosing machine learning to infer the longitudinal and lateral behaviour of drivers.

Over the years, different approaches have been developed for decision-making applied to intelligent vehicles. Initial endeavours relied on finite state machines to determine actions required by the situation plus a trajectory planner to drive the vehicle [7]. However, these required precise representation of the environment and when forced with complex situation did not scale [22]. Probabilistic reasoning allows the uncertainty of an information to be represented and used for decision-making. It has successfully been applied to many driving scenarios [28, 29, 30]. It also requires human knowledge to build parts of the models. This has been improved using machine learning approach to learn models from data [31]. In most advance works, learning algorithm can learn by itself all the decision-process [32]. These approaches are presented in section 2.4 and rational to apply probabilistic reasoning are discussed.

2.2 Context Formulation: The Navigation Function In Intelligent Vehicles

2.2.1 Classification of Intelligent Vehicle of Architectures

The control of an intelligent vehicle can be shared between different entities, namely the driver, the vehicle and through the communication network. Regarding the emphasis on whom is in-

involved in the navigation task, different architecture designs can be chosen. These can be classified in three categories: vehicle centric, driver centric and network centric [33].

2.2.1.1 Vehicle Centric

A vehicle centric architecture implies that humans are outside the control loop, that is the vehicle is under full computer control. The system builds a digital model of the environment by fusing data from embedded exteroceptive sensors as well as a-priori knowledge from maps and models. This type of architecture, whilst it can use when available information from wireless communications, it does not depend on it. Figure 2.1 shows a typical vehicle centric architecture.

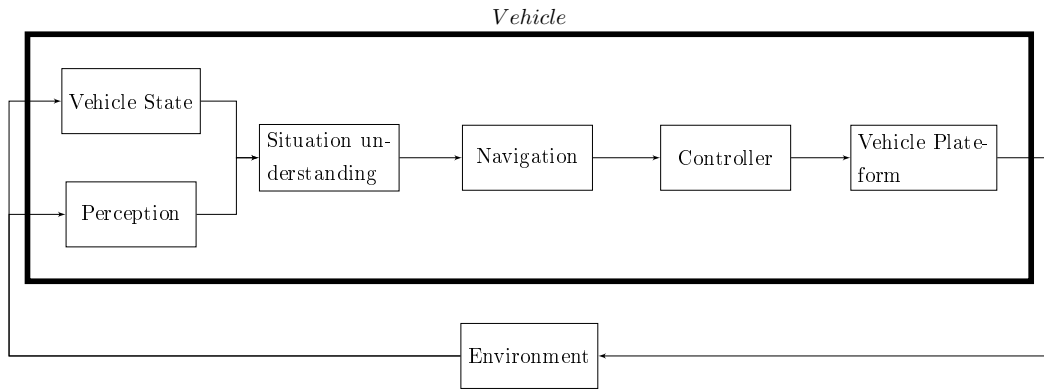


Figure 2.1 Vehicle centric architecture

For several developments of AD level-4 vehicles (e.g. Waymo), it is the preferred AD architecture. That is to operate on a stand-alone basis. However, their validation remains challenging, as many difficult safety problem (e.g. software redundancy, malicious attacks) must be covered.

Whilst there has been much progress, multiple challenges remain [13]. A major issue is machine perception. Sensor layout implies that several occluded areas remain in dense and urban situations. The response from sensor is far from ideal due to their bandwidth, resolution, acquisition rates, etc. Algorithm of machine learning methods still result in false positive and false negative. There is much uncertainty with regard to the digital representation of the environment.

Further, the behaviour of human entities populating the environment is difficult to predict and interactions resulting from vehicles decisions must be socially acceptable. Another challenge has emerged, the capability of qualifying the system performances in order to guarantee its behaviour and safety.

2.2.1.2 Driver Centric

To facilitate the deployment of intelligent vehicles, a driver can keep a certain degree of control on the navigation system. The driver is expected to perform some navigation tasks (e.g. at road intersection crossing, parking ...), to monitor the system or to act as a fall-back solution. This architecture requires driver monitoring, to check if the situation encountered is understood and infer if the driver is ready to take control back. It implies that decisions are needed to determine when the driver is needed and whether or not he is ready to act. Figure 2.2 shows a typical driver centric architecture.

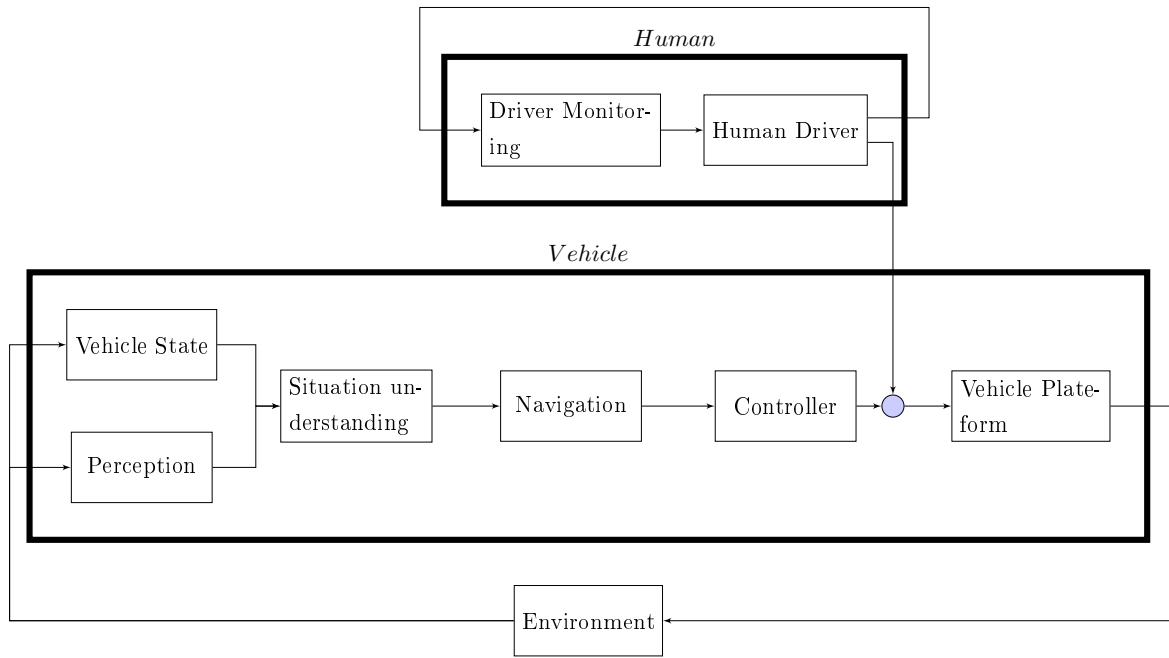


Figure 2.2 Driver centric architecture

The most well-known example is the Tesla Autopilot that can operate on motorways. Figure 2.3a shows its HMI and messages that indicate drivers to take control back. This system maintains the vehicle in its lane and maintains a safe distance with other traffic entities.

There is limited interaction between driver and vehicle in the Tesla example. A more interactive system that includes driver monitoring function has been proposed by the HCAI team at the MIT [34]. Figure 2.3b shows the monitoring of the driver activity and risk associated with this activity and the driving situation. This information allows to decide whether or not the driver should take control back.

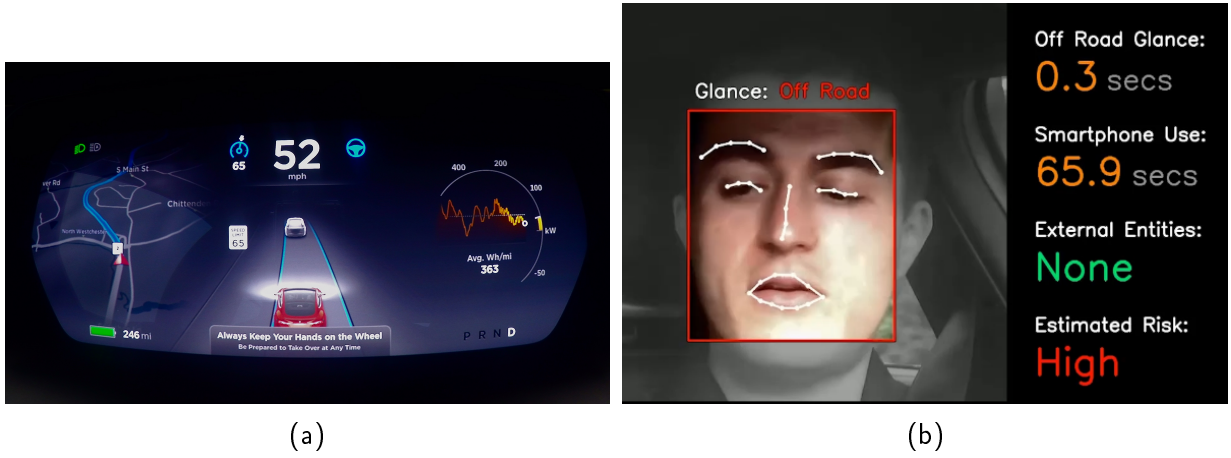


Figure 2.3 Interfaces for driver centric intelligent vehicles with (a) the HMI for the Tesla Autopilot asking the driver to put his hand on the steering wheel, (b) driver monitoring display, after [34]

The problem keeping humans within the navigation loop is that drivers task sharing is difficult. Over-trusting the capabilities of vehicles is likely when drivers are exposed to a few challenging situations [35]. Huang et al. proposed to show drivers challenging situation for the decision-making system to study how trust can be built [36]. When the system decides to pass the control to the driver, the time taken to understand the situation and to find the correct decision is difficult to estimate. It depends on the driver cognitive load and its physical state. If the transition can be known in advance (e.g. leaving an AV compliant area), the transition can be prepared. However, during emergency situations, up to 10 seconds of response time can be expected when a passenger is distracted and might not be sufficient for safety critical situations.

2.2.1.3 Network Centric

For driver and vehicle centric approaches, sensors and decision systems are on-board the vehicle. These functions can be enhanced by sharing information from other vehicles or infrastructures via wireless link. The adoption of wireless vehicle communications on-board motor vehicles, for navigation purposes, is an enabler. This comes under the umbrella of different standards, like the IEEE 802.11p. A vehicle can share information with other vehicles (V2V) and with the infrastructure (V2I), these are grouped under the abbreviation V2X. Figure 2.4 shows a typical network centric architecture.

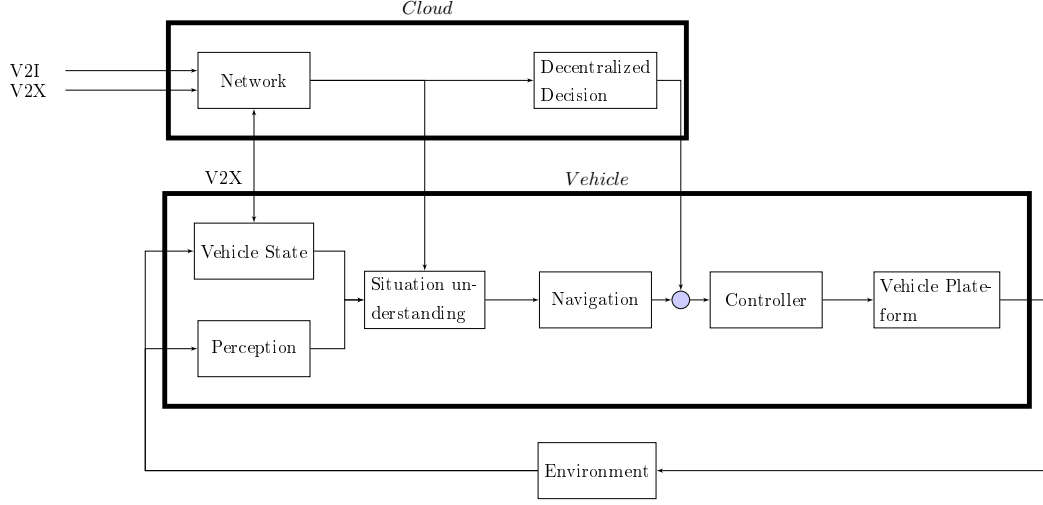


Figure 2.4 Network centric architecture

By introducing information from outside the ego vehicle, the construction of the world model can be enhanced and thus situation understanding improved. V2X depends mainly on knowing the state of the emitting vehicles. Moreover, cooperative decision-making can be applied to optimize the general traffic flow in a decentralized manner [37, 38].

The vehicles under this type of operation are known as network centric. The ultimate aim being to partition the vehicle intelligence between the on board computer and the cloud as new communication links appear as 5G [39]. This is expected to have high impact on the deployment of intelligent vehicle.

The impact of communications to achieve safer road has been explored in different European initiatives [40, 41]. A state-of-the-art on cooperative decision-making at intersection has been done by Chen et al., it showed the potential to improve safety and traffic flow [42].

A new issue to be considered is the network security to avoid malicious attacks. Further, shared information is subject to other forms of uncertainties associated to the communication channels and shared information. Today major issue is the availability of communicating vehicles and equipped infrastructure, they are rare and costly.

2.2.2 Discussion

The classification into different architectures for intelligent vehicles, from a navigation and decision-making perspectives, provides us with an insight into the level of situation understand-

ing, and decision-making required if these is to be made by a computer.

The scope of this thesis, centres on vehicle centric architecture. The rational is as follows: an autonomous vehicle that has drivers as part of the control loop has different implications more from an ergonomics perspective due to the grey zone on the vehicle controllability. Network centric vehicles are fundamentally vehicle centric, with different constraints that can be taken into account. That is communication issues (e.g. delays, blackout, etc).

While any of these two approaches provides efficient solution to the navigation and decision-making problem, works on vehicle-centric system are more likely to increase the autonomy of intelligent vehicles. The development of the three systems shall coexist in order to address associated challenges. However, developments for vehicle centric systems remains a core difficulty for the deployment of autonomous vehicles for Level 3 and 4. Such vehicles drive alongside human drivers that have complex behaviour and need interact with it. The understanding of human behaviour is discussed in Section 2.3 and a contribution on the detection of driver's behaviour is presented in Chapter 3. A decision-making process for vehicle centric system needs to include interaction and driver's behaviour in order to drive in mixed and uncertain environments. This is presented in Chapter 4 with a POMDP to navigate at road intersection crossings.

2.2.3 Decision-making for Vehicle Centric Architectures

In a vehicle centric architecture, the navigation system makes plans in order to reach a destination in a safe manner [43]. It takes advantage of a digital representation of the environment built by the perception system. It also needs to understand its own state and current dynamics to plan the vehicle motion. The state of the navigation function system can be supervised by another system to change its configuration. At last, trajectories generated by the navigation function are sent to the vehicle control system to create the appropriate actuator commands. Interactions between each system are shown in Figure 2.5 .

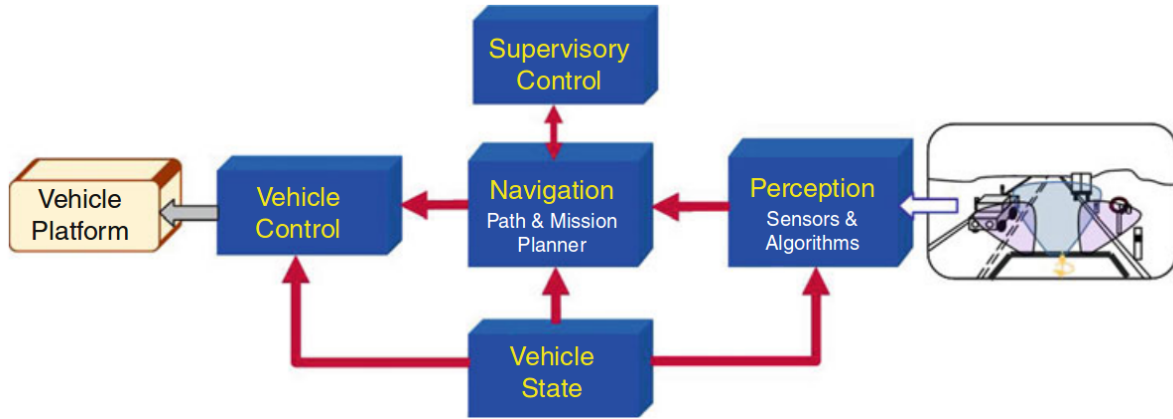
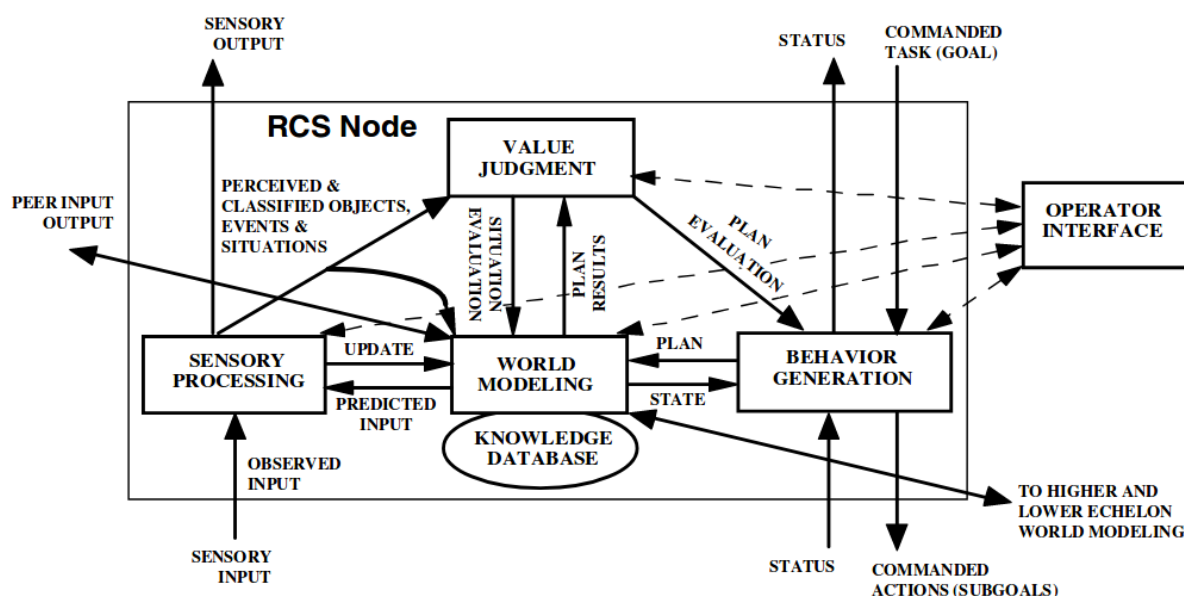
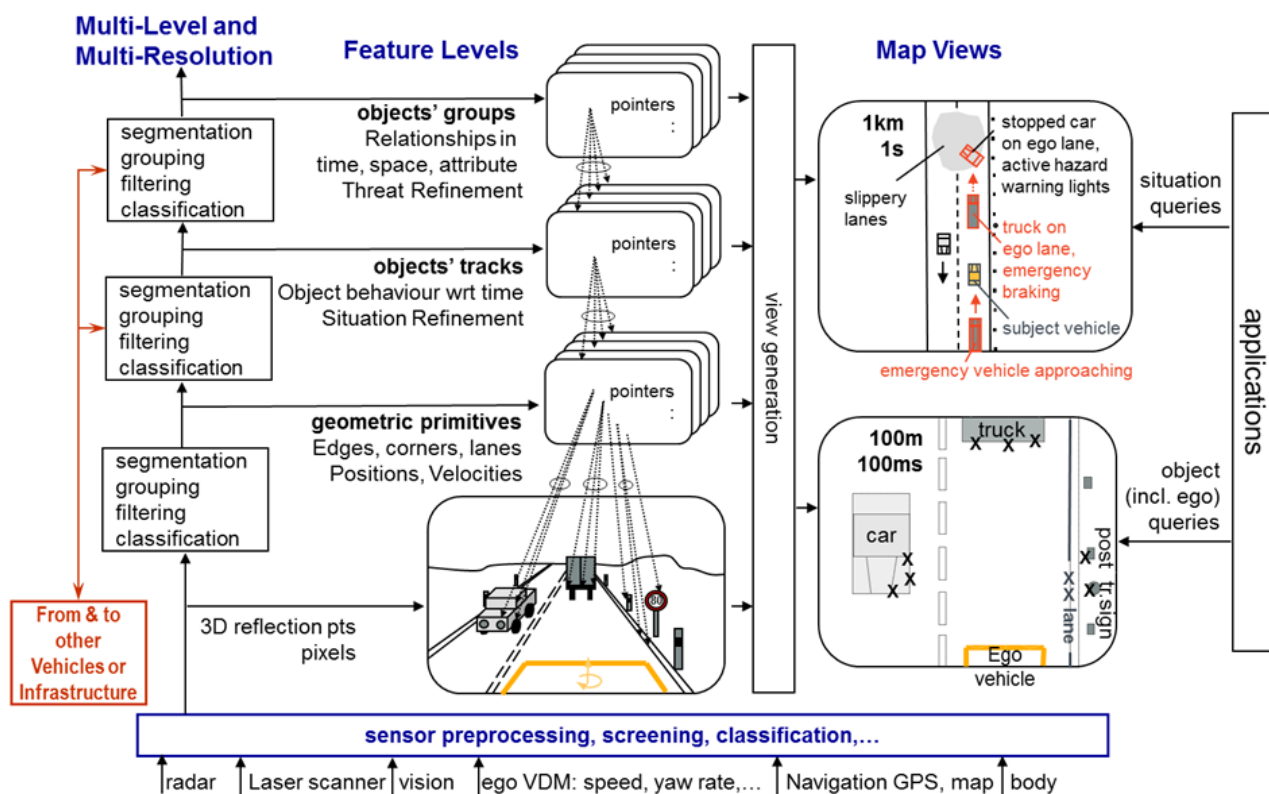


Figure 2.5 Vehicle centric architecture, after [33]

The navigation module maintains plans on different time and spatial scales. The architecture proposed by Meystell et al. distributes and groups these planning functions given the time and space scale [44], as shown in Figure 2.6. Functions are implemented as computational nodes using the real-time control system (RCS) architecture. Each node processes its sensory input differently as they operate on a different time scale. Short term (1-3 seconds) planning requires simple but accurate environment representations, whereas decision-making nodes reason with long term predictions and uncertain information, an example is shown in Figure 2.6b. The subprocesses of a RCS node, shown in Figure 2.6a, are required for any decision-making system. There exist interconnections between layers to maintain a coherent plan if one layer is unable to execute the request task. It shows that the decision-making is not an isolated task. Internally, it consists of different functions that maintain an appropriate representation of the environment, the necessary knowledge to evaluate plans and a process to generate them. Externally, it works with other decision-making systems that work on different scales.



(a) RCS node internal structure, after [44]



(b) Layered architecture for situation understanding, after [33]

Figure 2.6 4D/RCS Reference Model Architecture for unmanned vehicle

2 Related Work on Behaviour Estimation and Decision-Making for Road Intersection Crossing

In vehicle centric architectures, the navigation system is often divided into three sub-systems that operate at different rates and use different scales of world representation as in the 4D/RCS:

- The route planner finds the path from one location to another using maps.
- The behavioural planner determines the correct actions to be executed in a specific situation (e.g. crossing an intersection, Figure 1.3). Its environmental model is made using lane-level maps, perception outputs and behaviour models to understand human driver's motion.
- The trajectory planner generates a dynamically feasible safe trajectory with respect to behaviour planner actions and to immediate collision dangers.

Their outputs are used by the other sub-system as goal to be achieved. Whereas the role of the route and trajectory planner are commonly agreed on, the behavioural planner is harder to portray. Compared with the trajectory planners that react to the immediate situation, the behaviour planner takes a deliberative approach to plan with long-term prediction and multiple hypothesis. Thus, its output can be of many forms like way points, motion recommendations, motion constraints, accelerations, etc.

Figure 2.7 shows the underlying architecture of a typical navigation system for a vehicle centric solution, including the information received from other system.

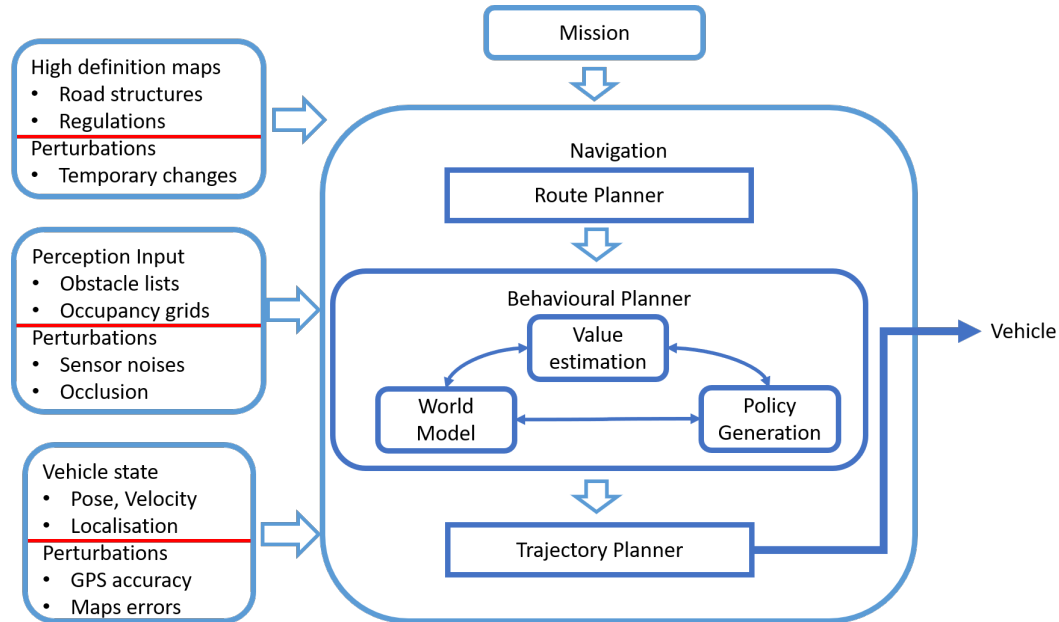
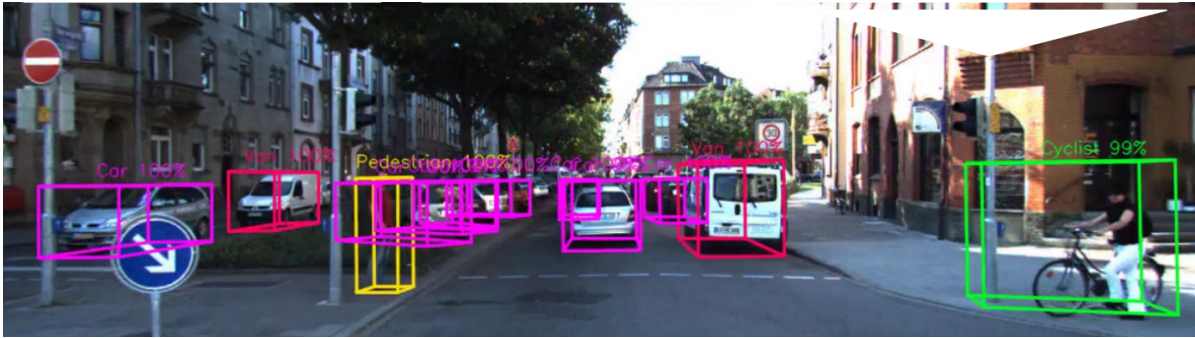


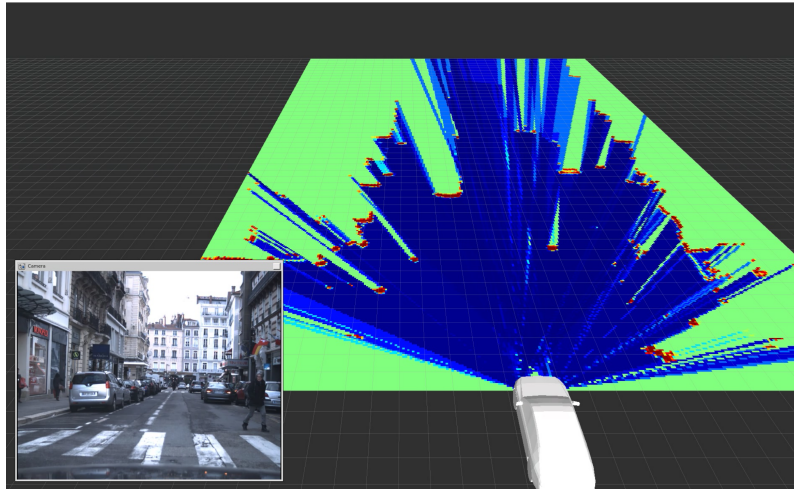
Figure 2.7 Navigation function of a vehicle centric system with inputs and perturbations

2 Related Work on Behaviour Estimation and Decision-Making for Road Intersection Crossing

The perception system generates a list of obstacles representing the perceived entities and associated attributes, as per the examples in Figure 2.8a. Attributes associated with these objects include: type (e.g. vehicle, pedestrian, etc.), pose (position and orientation), velocity, etc. This information is associated with a knowledge data-base to provide context (e.g. maps) and behavioural attributes (with respect to learned models). This type of representation is used for the behavioural planner for long term predictions. The trajectory planner operates at a higher rate and can benefit of simpler representations such as dynamic occupancy grid or short-term prediction, as shown in Figure 2.8b.



(a) Object list with classified object type, after [45]



(b) Probabilistic occupancy grid with dynamic cells (red), free space (blue) and unknown (green), after [46]

Figure 2.8 Different types of output from perception systems

A navigation system uses a-priori knowledge of its immediate surrounding in the form of High-

Definition maps (HD-maps). These are precise lane-level representations of the road geometry together with attributes that indicate the expected driver behaviour at a given position. An example of such representation is shown in Figure 2.9.

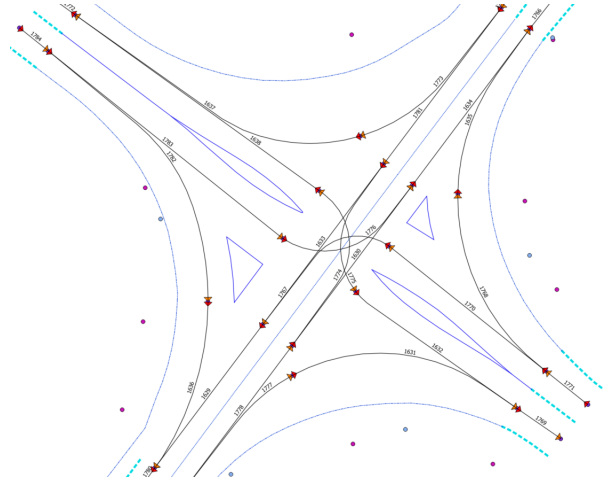


Figure 2.9 High-definition map from Renault of a crossroad that includes direction allowed and lanes marking (in blue)

Information on the vehicle state is also updated as this determines the vehicle pose and velocity. Vehicle location and heading can be estimated using GNSS receiver, vehicle odometry, structure from motion cameras and point clouds [47, 48]. Figure 2.10 shows a typical structure of a vehicle state estimation system for autonomous vehicles.

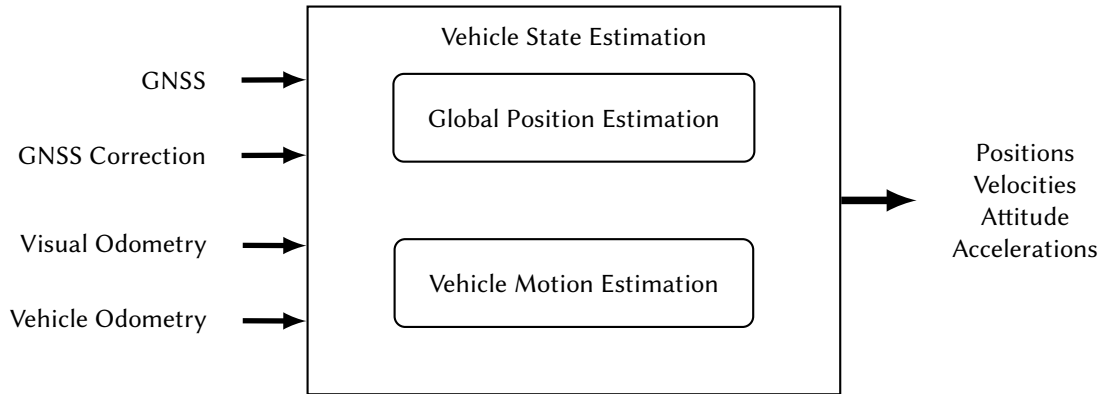


Figure 2.10 Functions of a vehicle state estimation system with its typical inputs and outputs

This information is used to perform the different navigation planning tasks. For the road intersection crossing scenario, the route planner level provides a path that joins the current vehicle position with the desired intersection exit. The behavioural planner selects the longitudinal motion (e.g. velocity profiles, motion constraint) to be performed given the priority rules and other vehicles behaviour. The trajectory planner executes that plan, maintains the vehicle in its lane and modifies the trajectory when unexpected obstacles or aberrant obstacle actions are detected.

2.2.4 Discussion

When few uncertainties are considered, the situation understanding for the behavioural planner can use simple prediction models to constrain the trajectory planner in time and space. However, at road intersection occluded vehicle and aberrant driver behaviour are important sources of uncertainties in mix traffic. These are perturbations for planners or force the system to adopt conservative behaviour.

Considering probabilistic inputs and predictions into the evaluation of the decision allow to search for a trade-off between risk and performances [49]. The behaviour planner can balance low probability events with serious consequence and typical events with no threat to the vehicle. As the vehicle moves, multiple interactions will occur. It makes the probabilistic evaluation of driver's behaviour difficult and subject to temporal change.

The different planners that consist of the navigation function uses the many available inputs to generate and evaluate policies to be executed. None of these inputs can be precisely known. Thus, planners must be able to understand and reason with uncertainties associated with measurement and driver behaviour to be able to balance between risk and performances.

One of the difficulties to apply probabilistic reasoning of the inputs is the lack of probabilistic models for the other system on-board. These are often approximated with Gaussian distributions. Other types of uncertainties are created by measurement processing such as false positive and false negative, if not dealt with the planner might be erratic and dangerous.

2.3 Behaviour Understanding

In a vehicle centric architecture, the creation and evaluation of a policy requires estimations and predictions of other entity states. Simple dynamic models can explain their short-term motion,

however to predict on a longer horizon, behaviour understanding is necessary [50]. This section presents definitions of terms associated with driver's behaviour in a road intersection crossing scenario and the main approaches for its understanding.

2.3.1 Definitions

To study the behaviour, it is necessary to understand the cognitive process from which a human generates actions. It can be divided into three stages: Skill, Rule and Exploration [51]. Figure 2.11 shows the reasoning which results in an action caused by a new sensory input. The immediate response from the skill level is important for collision avoidance and short-term motion planning (e.g. lane following). However, these actions are not the result of a long-term plan. Within the other two levels, humans need to understand the situation and actions of other drivers to generate the desired motion.

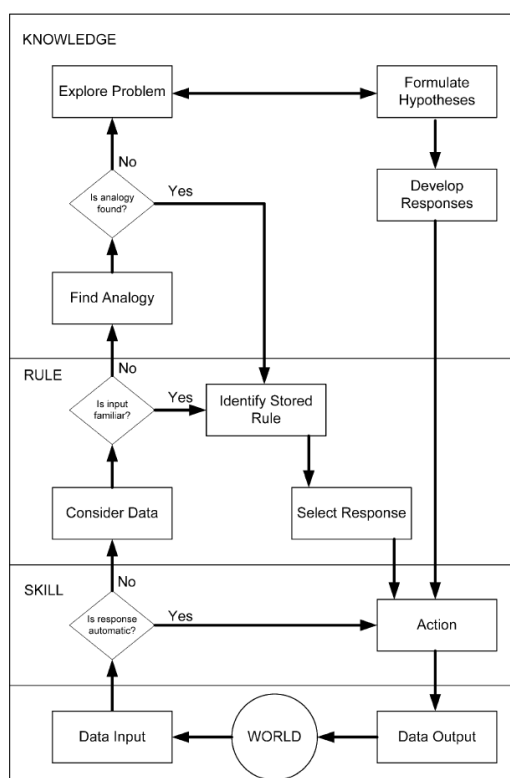


Figure 2.11 Levels of cognitive control, on how humans process information from the environment and generate decisions and actions, after [52].

Limitation of human perception and drivers' inattention lead to a partial understanding of situations, thus could lead to wrong decisions and dangerous behaviours. Further, the result of the rule and knowledge stages depends on the driver experience. It is then difficult to be understood from the point of view of another entity. Between two human drivers, interactions are understood through a non-verbal exchange, it represents a major challenge for autonomous systems as machines are far from being able to achieve such interaction. Human response evolves as new sensory input is processed, it can result in behaviour change. The combination of potential human errors and versatile behaviour makes situation understanding and predictions difficult for intelligent vehicles. There is a certain degree of randomness in real-traffic conditions.

Terms such as behaviour and manoeuvre are often used to refer to different aspects of the driver state. They correspond to the response of the knowledge and rule stage of the cognitive control that is used to generate actions. For consistency and within the scope of this work, the following definitions are used.

Driver *behaviour* explains the driver's sequence of actions or policy, over a period of time. It consists of a manoeuvre and attitude that is used. The *manoeuvre* is strictly dependent of the situation. It can be divided into the lateral and longitudinal manoeuvres. The *Lateral manoeuvre* results in policies that deviate the vehicle position perpendicularly from the centre of the current driving lane (e.g. to overtake or change lanes). The *Longitudinal manoeuvre* changes the vehicle speed along its direction of motion, to adapt to the traffic conditions or to the signalization.

At road intersection, driver manoeuvre can be regarded as the combination of the following:

- The lateral manoeuvre is constrained by the road geometry at the crossroad. In a nominal scenario the lateral intention can be defined by the following set that represent possible direction

$$\mathcal{M}_{latt} = \{Turn\ right, Turn\ left, Go\ straight\} \quad (2.1)$$

- The longitudinal manoeuvre can be of three types:
 - *Stop manoeuvre* to obey traffic signals the signalization (traffic light or stop sign) or to wait for a sufficient gap before crossing the intersection. The vehicle velocity is null for a period of time at the intersection entrance.
 - *Yield manoeuvre* to allow vehicles with the right of way to cross the intersection first. The vehicle might not necessary come to a stop as drivers could intent to merge with the traffic.

- *Cross manoeuvre* when traffic signal or situation gives the right of way to the driver. The vehicle velocity at the intersection could be high, as drivers expect other vehicles to yield. The Figure 2.12 shows typical velocity profiles of vehicles approaching an intersection and performing the described manoeuvres [27]. The resulting manoeuvre set for the longitudinal manoeuvre is:

$$\mathcal{M}_{lon} = \{Stop, Yield, Cross\} \quad (2.2)$$

At road intersection, the lateral manoeuvre changes the priority that apply. It is necessary to understand it prior to the longitudinal manoeuvre.

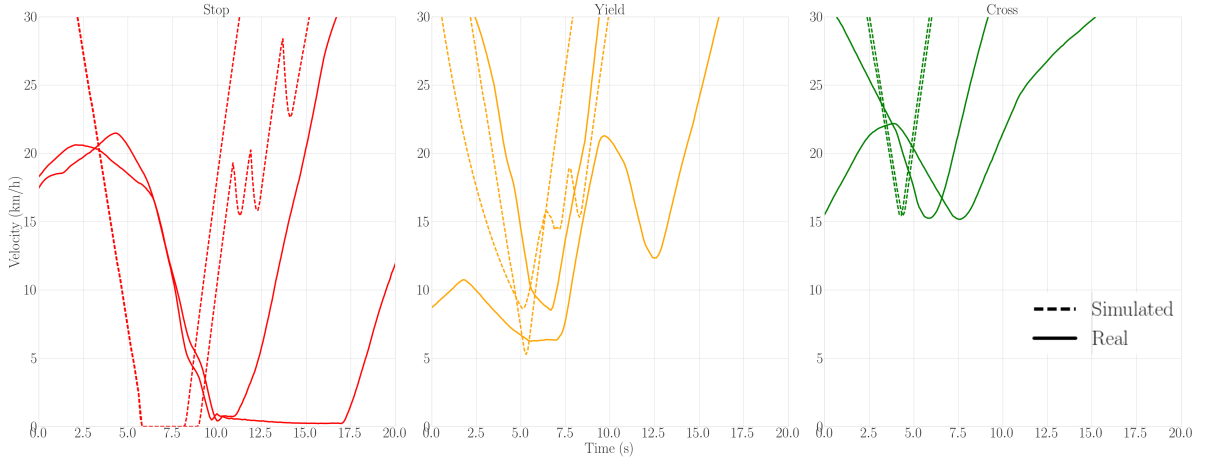


Figure 2.12 Vehicle velocity profiles when driving toward an unsignalized intersection. These are extracted from the dataset presented in Chapter 3.

The driver *Attitude* indicates how the driver is responding to the situation. When executing a manoeuvre, the attitude of the drivers give information about the aggressivity, the mental charge of the driver or the coherences of his actions with the situation.

For most of the scenarios, a manoeuvre can be found to avoid collisions and to maximize the travel time, however, humans are often unable to adopt the adequate attitude, willingly or unwillingly. Consequently, 60% of accidents are due to driver's actions, as documented in Section 1.2.2. Behaviour attitude is a major source of predictability uncertainty.

At road intersection, the description of driver attitude can be of several types, such as aggressive, hesitating, sportive, inattentive, etc [53, 54, 55].

The attitude, as described by Lefevre et al., differentiate the driver *intention* and *expectation* to infer a level of risk [53]. The intended manoeuvre represents what is the driver's current manoeuvre. The expected manoeuvre is the probable manoeuvre required by the situation. The probability of the two variables being different is used as an indicator of risk associated attitude of the other driver.

Another aspect of driver attitude is the *driving style*. It helps to predict sudden changes of action or actions outside their typical ranges. There are various ways to describe it such as: Aggressive/Normal/Hesitating [55] or Average/Sporty/Relaxed [54].

Relation between each element of the behaviour is synthesized in Figure 2.13.

It is necessary to evaluate these aspects of the behaviour in order to increase the prediction horizon but also to have a better situation awareness.

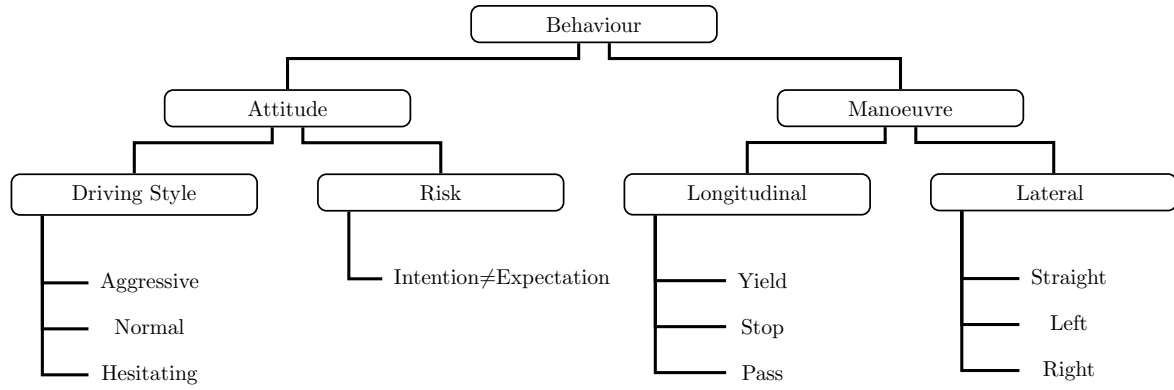


Figure 2.13 Elements to describe a driver's behaviour

2.3.2 Methods for Behaviour Understanding at Road Intersections

Prior to the behaviour understanding function, two fundamental processes are needed: the vehicle detection and vehicle tracking steps [56]. The former identifies obstacles out of the sensors measurements. Current state-of-the-art methods uses learning algorithms to train models to detect vehicle salient features in measurements [45]. The latter tracks vehicles across multiple frames. It allows to re-identify and to obtain the past trajectory of the vehicle. It uses estimation methods (Kalman filter) to fuse and compensate measurement errors from multiple sensors [57]. Vehicles detection and tracking functions are considered as part of the vehicle perception system,

whereas the vehicle behaviour recognition part of the world modelling function. These steps and processed information are shown in Figure 2.14.

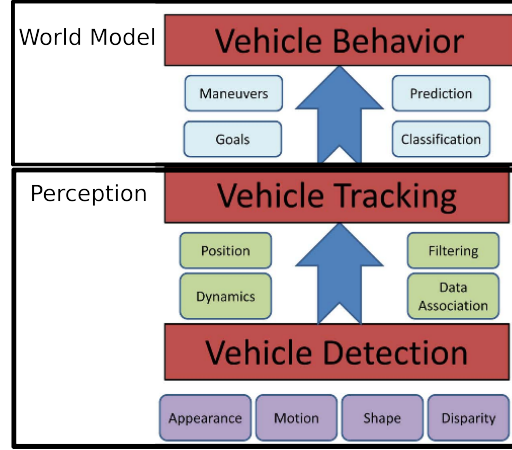


Figure 2.14 Vehicle understanding analysis, from [56]

Behaviour identification from past situations is mainly done applying classification approaches and probabilistic graphs. The differences between the two approaches are illustrated in Figure 2.15. Both approaches are described in detail in the following sections.

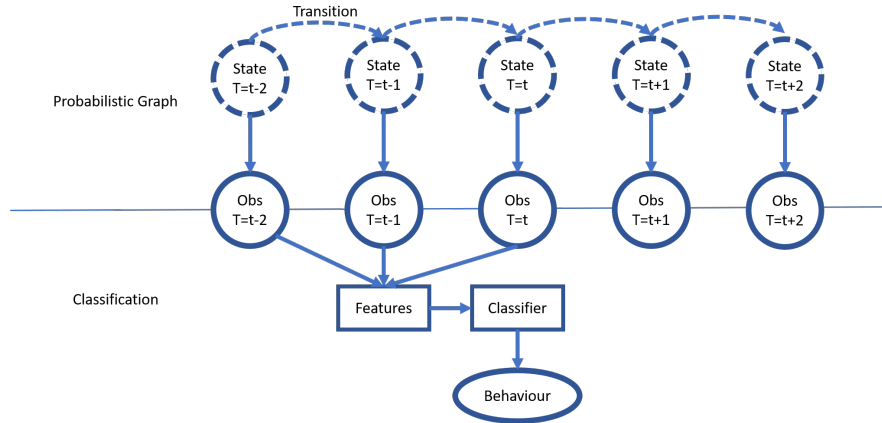


Figure 2.15 Top layer represents a probabilistic graph approach, the dash circles describe the hidden variables. The bottom layer shows the classification steps. Features are computed from past observations, these are used by the classifier from which the behaviour identification is done

2.3.2.1 Classification

A classification approach uses features that can be computed out of observations made on the vehicle trajectory to differentiate between different classes of longitudinal and lateral manoeuvres.

In general, most estimates are based on proprioceptive vehicle sensors that allow to measure velocities, accelerations and pose [58, 59]. Heading and position in the lane have more importance for lateral manoeuvres classification [60]. Observations on the environment and HD-map information provide context, particularly when arriving to complex areas as intersections, improving the manoeuvre classification process [59]. CAN bus information, a perception system on the car or installation on the infrastructure can be used to build training dataset. The driver state also contributes to the classification, that is its gaze, actuation on the vehicle controls (e.g. accelerator or brake pedal) [61]. However, it causes privacy issues and such classifiers are relatively dependent of a specific driver.

Lateral manoeuvre at road intersection crossing can be classified with different algorithms: Support Vector Machine (SVM) [62], Random Forest Classifiers (RFC) [58], neural network [60, 63]. Current state of the art for classification are neural net used by Philips et al. to classify the lateral manoeuvre at multi-lane intersections [60]. They extract features of the vehicle state for the last 3 seconds and the traffic state recorded from video camera surveillance. They trained Long Short-Term Memory networks (LSTMs), obtaining an accuracy of 85% at up to 150m from the intersection. During the training the position of the vehicle weights more than any other feature to understand the lateral manoeuvre, furthermore at multi-lane intersection. It questions the necessity to use 104 features to train their network.

A simpler way to collect a dataset has been proposed by Gross et al. which uses GPS information from a fleet of vehicles at a crossroad intersection [58]. To address GPS uncertainty (several meters of errors in the vehicle position), they pre-processed position measurements with a map-matching algorithm. Information from three intersections were used to train a Random Forest Classifier to detect the lateral manoeuvre. An accuracy of 80% at up to 40 meters from the intersection entrance was obtained. They showed that a model trained using information from different intersections can accurately classify the lateral manoeuvre at several intersections with similar layouts.

The longitudinal manoeuvre is mostly influenced by the distance from the intersection and the state of the environment. For example, Garcia et al. used multi-layer perceptron neural network, to predict behaviour manoeuvre at a controlled intersection using as input the traffic light phase

[59]. This is similar to a yielding situation since the driver's behaviour is changing because of the signalization. In their scenario, it was observed that the vehicle only reacted to the passage from green light to red light. Thus, the behaviour is highly dependent of the traffic light state. Also at traffic light-controlled intersection, Aoude et al. classified the stopping intention of drivers using a SVM completed with a Bayesian filter [64], its structure is shown if Figure 2.16.



Figure 2.16 Stopping intent classifier using SVM and Bayesian Filter, after [64]

The Bayesian filter uses the current and past SVM output to avoid sudden change of classification. Their dataset contains 300 000 trajectories recorded at a single intersection. After lateral manoeuvre classification, the longitudinal stop manoeuvre is classified using the same dataset [58]. In this example, the velocity and acceleration were the most important features and similar accuracy to the one for the lateral manoeuvre was obtained.

The driver attitude can be found comparing the output of the classification with a different indicator. Aoude et al. compared the output of their classification with the remaining time to stop to find non-compliant drivers (i.e. those that have not the intention stop) when close to a traffic light turning to red [64]. Driver attention, if measured, can be a good indicator of his attitude. Tawari et al. recorded the head position of a driver approaching an intersection as well as traffic light status and trajectory [61]. The trained random forest classifier was able to identify 80% of unexpected manoeuvre up to 2 seconds before the manoeuvre event.

2.3.2.2 Probabilistic Graphs

Probabilistic graphs, i.e. Dynamic Bayesian network (DBN) and Hidden Markov Model (HMM), have been used to infer state of variables that will generate a sequence of observations. The behaviour is modelled as hidden variables that influences transitions of the system to other states.

HMM has been used to find driver's lateral manoeuvre [64, 65, 66]. They all used velocities, accelerations and yaw rate as observations. For example, an HMM for each of the intended

route is trained with data from a single manoeuvre [65]. Then, a probability of belonging to the HMM trained-with the manoeuvre is found for, each of the observed sequences. They managed to accurately classify 90% of drivers lateral manoeuvres. However, they used an unbalanced dataset (i.e. more driver going straight than turning) and observed that the understanding of people going straight is harder due to high variance in accelerations and velocities observed. For comparison purposes, Aoude et al. used two HMMs each one trained with one of the two attitudes [64]. In comparison to the SVM and Bayesian filter method, HMMs were less accurate.

Instead of training from a dataset, a priori information such as maps can be used to set models for inference in a Bayesian network [67]. Lefevre et al. extended these models to include the vehicle kinematics to detect incoherence between the expected behaviour and the current intention of the driver [53], where these differences are used to characterize a risk. The driver's attitude is modelled as a difference between the intention and expectation manoeuvres. Turning lights is also included, to understand incoherent manoeuvres [68].

Another application of these models is to obtain better predictions of the future vehicle state [66, 69]. However, the difficulty with probabilistic graphs is to include interaction between entities without creating loops [66]. Interactions are linked to conflict areas the likely path of two vehicles could overlaps. For each conflict, priority orders are found using HD-map. Then, the likely longitudinal manoeuvre is found considering pending conflicts. From the inferred behaviour, actions (accelerations) for each vehicle are found with an adapted Intelligent driving model. The structure of their DBN is shown in Figure 2.17.

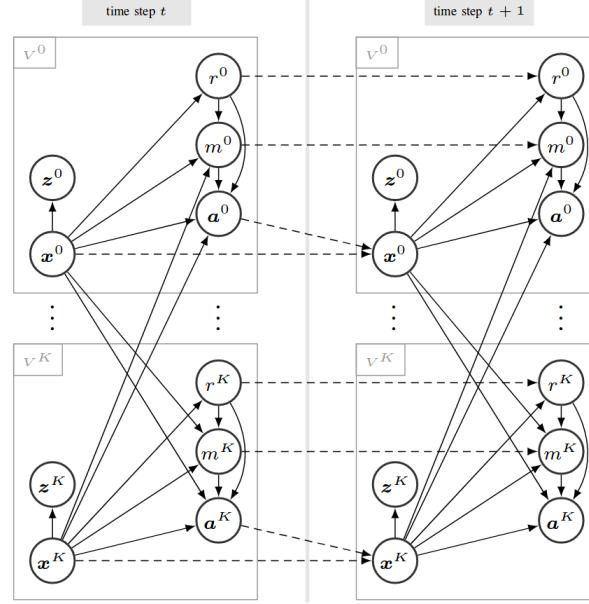


Figure 2.17 DBN structure to infer driver behaviour considering interaction. Each square contains state variables and the manoeuvre of a vehicle. The manoeuvre of each vehicles depends on other vehicle states, after [66]

It enables prediction of vehicle motion over 5 seconds. However, the scalability of these models to situation including numerous element is unknown. Most of the time particle filter are used to represent these complex probabilistic distribution [53, 66]. It allows an online implementation that works with few vehicles, but performances could degrade with scene complexity.

2.3.3 Discussion

The main difference between the two approaches resides in the length of information required to estimate the behaviour. Classification algorithm handles large feature vectors without increasing the complexity of the prediction phase. By contrast, for probabilistic graphs, the complexity will grow with the number of states and observations. Therefore, classifier could address more complex situations. In processes where the succession of action is important, probabilistic graph would be more efficient as transitions are already integrated in the algorithm. In the next section about decision-making, probabilistic graphs are used to represent situation transition. Classifiers could be used to find the current behaviour with multiple features from the situation, and combined with a probabilistic graph to propagate the effect of this behaviour on future situations.

For machine learning, the training datasets are as important as the selection of methods applied to it. The difficulty for road intersection crossing is to obtain sufficient trajectories due to costs as well as sufficient variety of crossroad intersection layouts [70]. For example, the dataset used in [64] includes 3 million trajectories, out of which only 15 thousand were used. A less costly method will be to use GPS dataset, however it requires lot of pre-processing [58]. Furthermore, the dataset must be balanced and with a significant number of trajectories for each class, else, some manoeuvre features will be ignored resulting in less accuracy [65]. Creating the data set from the point a view of the vehicle is difficult because of measurement noise and the difficulty to observe a complete trajectory.

The understanding of the lateral manoeuvre is easier, as features like the position of the vehicle at a multi-lane intersection is highly discriminant. Nevertheless, it will be more difficult to infer at single lane intersections, as changes to the position and heading occur close to the intersection entrance. Turning lights are an indicator of the lateral manoeuvre [68]. This is often an unreliable source of information due to its erratic usage [71].

The longitudinal manoeuvre set is often restricted to the *Stop* and *Cross* manoeuvres. It is sufficient for controlled intersection (stop sign or traffic lights). However, at uncontrolled intersections, it could be insufficient for the decision-making. For example, the behaviour of a vehicle with a relatively high speed could be understood as intended a *Cross* manoeuvre whereas its real intention is to yield, thus the subject vehicle will be expected to come to a stop. The resulting situation will be that both vehicles slow down and come to a stop. If the yield manoeuvre would have been recognized, the subject vehicle could have predicted that a sufficient gap will be available in a close future and try to cross the intersection.

The behaviour of a vehicle is difficult to be inferred. Any approach discussed attain more than 90% accuracy. Therefore, it is important for the decision-making system to reason with errors of the behaviour understanding process and drivers' attitude that are important sources of predictability uncertainty.

2.4 Approaches to Decision-Making at Road Intersection

There exist three categories of decision-making for vehicle navigation: rule based, probabilistic, and machine learning. The difficulties to categorize them are the existence of many hybrid approaches, different types of output and systems that operate on different levels. What binds

them is that they control the motion of the vehicle using processed perception information that has been enriched with behaviour understanding or other situation understanding information.

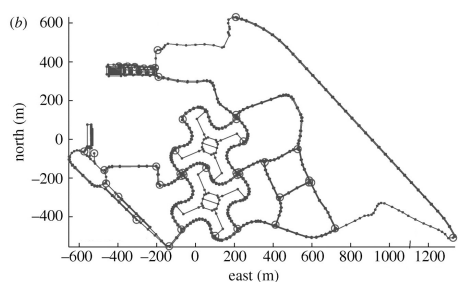
2.4.1 Rule-based

Rule-based method models human knowledge about a decision-making process as set of conditions and deterministic relations. These are also often called expert system. Driving is regulated by rules defined by the government and mostly applied in structured environment. Thus, this problem is well adapted to rule-based approaches but are challenged by humans uncertain behaviour and perception limitations.

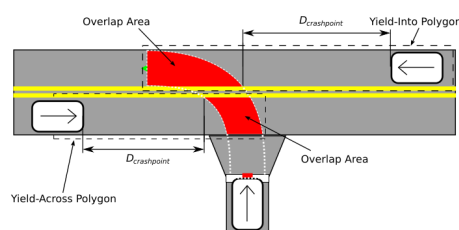
The 2008 DARPA Urban Challenge demonstrated the limited performances of rule-based methods. Situation that could be encountered and how to resolve them was stated prior to the challenge, thus similar behaviour could be expected from other participants. The winner used a high level behaviour planner to stop the vehicle at each intersection entrance [72]. It used a map with attributes associated to intersections (illustrated in Figure 2.18a) and vehicle position to plan trajectories that stopped the vehicle at the desired position. To decide whether to enter the intersection, two estimators must be valid: the precedence and the clearance. The precedence criterion is valid when every vehicle stopped at the intersection arrived after the subject vehicle. The clearance estimator builds a polygon that should not be crossed by any vehicle while crossing the intersection (shown in Figure 2.18b). If one of the moving entities in the scene will cross this polygon in the time needed to cross, the estimator will be invalid and the subject vehicle waits.

Even if most of the participant used a similar approach, some used other methods. The team ODIN used the DAMN architecture to generate their behaviour, that was also applied successfully as part of the Crossing America project [73, 74]. Multiple processes vote to control different outputs and an integrator decides which to follow, the system structure is shown in Figure 2.18c. In the circumstance of the challenge, rules and situations were sufficiently simple to be implemented by simple state machines. However collisions happened during the final of the challenge, with for example the collision between the MIT and Cornell teams [22]. It was caused by a misunderstanding of the interaction between the two vehicles combined with one of the vehicle to be unexpectedly stopped. Situations of the real world are more diverse and complex and would require, with such approaches, more complicated state machines.

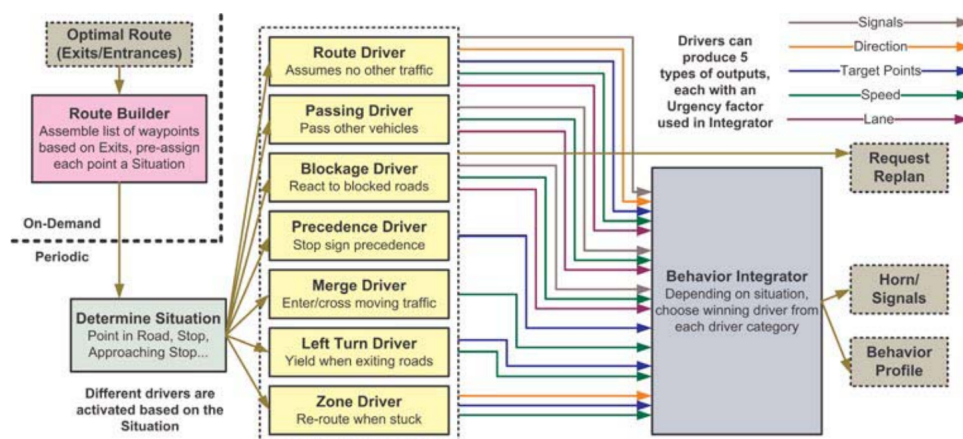
2 Related Work on Behaviour Estimation and Decision-Making for Road Intersection Crossing



(a) Road network definition file containing the map [75]



(b) Intersection view for Boss Team [72]



(c) Flow diagram of the Behavior-Based, Winner-Takes-All Driving Behaviors implementation. The behavior Integrator ensures there is one winner from each driver category, after [74]

Figure 2.18 Application of rule-based methods during the 2008 DARPA Urban Challenge

Liu et al. created state-machines for other situation with traffics light, pedestrian and included

the lane change manoeuvre that could be necessary to correctly position the vehicle [76]. In their system, while getting closer to the intersection entrance, different actions can be done. Each corresponds to a different deceleration, that for example reduces the velocity, stops the vehicle or performs an emergency stop. Transitions are controlled by conditions based on the vehicle distance to the intersection or the presence of other vehicles approaching the intersection.

The advantage of using state machines is that the behaviour of the subject vehicle is simple to verify as transition conditions are deterministic. However, measurement uncertainty could trigger an undesired transition resulting likely hazard situations. With the multiplication of state machines to handle more scenarios and with uncertain information, there is a possibility to select a wrong state-machine leading to incoherent decisions.

To allow for recovery and continuous analysis of the situation, a two stages rule-based decision-making system allows for more reactivity whilst approaching the intersection [77]. The first stage consists of a Petri net that filters the set of known manoeuvres to output only legal and feasible manoeuvres. The second stage uses multi-criteria decision-making to evaluate manoeuvres based on: safe distances, distance to the goal and to drive within lane boundaries. A fast update of the manoeuvre happens with an asynchronous event to react to changes of situation, but the sensibility to trigger these events must be carefully tuned to avoid false positive.

Developed originally for web crawling, ontologies have been recently applied to consider a larger variety of entities and situations. An ontology provides meaning and relationship between different elements. For autonomous vehicle, it is a database that includes the entities of the scene and their possible relationships [78, 79]. The decision is an attribute of the vehicle that is found by understanding the relation and interaction between attributes of the road and other entity attributes. However, the size of the database makes inference time an issue, also missing elements may cause reasoning errors. To enable real-time decision, a buffer of surrounding elements is used to obtain short response in [79]. These methods also assume that the system has a high confidence in the observation.

Overall, rule-based methods have the advantage to be deterministic, thus repeatable and simple to analyse. During the validation step, all scenarios could be analysed to gain confidence in the system performances. However, they require robust situation estimation to correctly change states. To use these methods with real measurement, large margins are often created in order to avoid entering in situations where uncertainties could degrade the decision quality. It results in conservative behaviour and sometimes unnecessary safety manoeuvres. These methods require substantial development effort if most of the driving situations need to be modelled.

2.4.2 Probabilistic Approach: Partially Observable Markov Decision Process

Probabilistic approach allows the uncertainty to be quantified as probability densities. Instead of using fixed values to represent an information, its range and distribution are also used. Consequently, decisions are chosen as actions that maximize an expected reward.

Model for probabilistic decision-making is named Markov Decision Process (MDP), firstly described Bellman in 1957 [80]. MDP considers uncertainty as stochastic transitions after an action, but it needs the current state of the system to be observable [81]. With this model, actions taken by the system will affect both the system and its environment. However, most of the time, the current state of the system is uncertain.

An extension of this model that considers partial observation of the situation is named Partially Observable Markov Decision Process (POMDP). In domains such as ecology, marketing and social studies this model helped humans to choose the best policy and to estimate its outcomes [82]. In robotic application, POMDP has been successfully applied on problems such as motion planning [83].

A POMDP models the decision process of an agent acting in an uncertain environment. Formally it is composed of $\{\mathcal{S}, \mathcal{A}, \mathcal{Z}, T, R, O\}$, with $\mathcal{S}$ the state-space, $\mathcal{A}$ a set of actions that the agent can take, $\mathcal{Z}$ a set of observations that can be obtained by the agent. T is the transition function $T(x, a) : \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ that describes how the system changes when the agent takes action a when in state x as the probability $P(x'|x, a)$. The reward function $R(x, a)$ indicates the value obtained after performing an action in a given state. And $O(x, a) : \mathcal{Z} \times \mathcal{A} \times \mathcal{S}$ estimates the probability $P(z|x, a)$ to obtain an observation $z \in \mathcal{Z}$ being in a state and taking an action. The agent does not know the real state after an observation, it reasons with a belief $b \in \mathcal{B}$ with $b : \mathcal{B} \rightarrow \mathbb{R}_{\geq 0}$ and $\int_{x \in \mathcal{S}} b(x) dx = 1$. Thus, the goal of the agent is to maximize the value $V : \mathcal{B} \rightarrow \mathbb{R}$ for an initial belief. In a POMDP there exists an optimal policy $\pi^* : \mathcal{B} \rightarrow \mathcal{A}$ that maximize V . The value of a policy can be estimated as the expected future sum of rewards for an initial belief $V^\pi(b_0) = \sum_{t=0}^{\infty} \gamma^t R(b_t, a_t) = \sum_{t=0}^{\infty} \gamma^t E(R(x_t, a_t) | b_0, \pi)$. In the context of AV navigation system, the agent is the decision-making system that decides the policy to be applied when approaching towards the intersection. The interactions between elements of the model are shown in Figure 2.19.

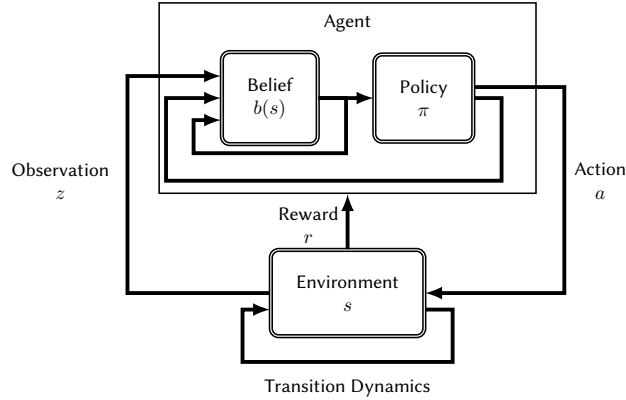


Figure 2.19 A POMDP model, and interactions between the agent (top layer) and the environment (bottom layer)

Moreover, elements of the POMDP model match the RCS node internal structure from [44] which allows the POMDP to be analogous to a planner unit of the generic architecture in vehicle centric system. Similarities between these two models are highlighted in Figure 2.20. Compare to other approach to decision-making that only include the behaviour generation or the value judgment, a POMDP includes in its formalism all the element necessary for decision-making.

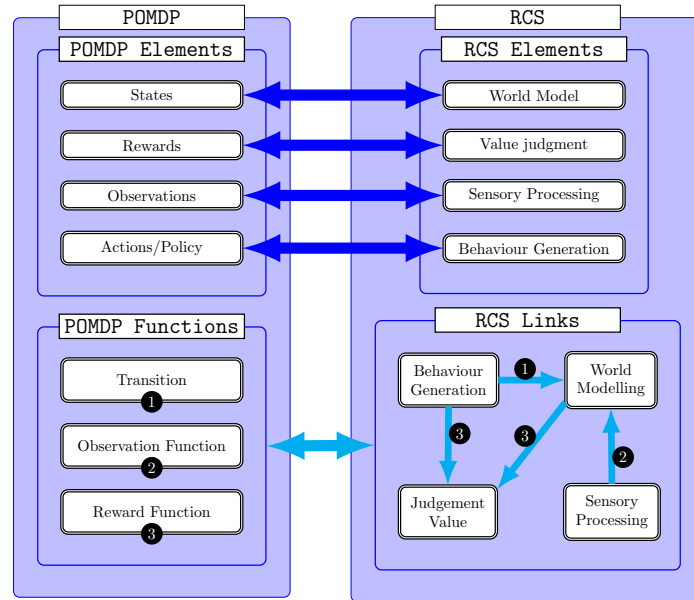


Figure 2.20 Similarities between POMDP and RCS node

Methods to find an optimal policy for a POMDP is in general complex and intractable. The complexity of the problem grows with the number of states, observations, actions and the decision horizon. Solver used to find this optimal policy can be categorized in two: offline and online. Offline methods explore all the model with all the actions to find the optimal after some time [84]. Whereas, online methods search the model for a given amount of time and return an approximate optimal policy.

Methods to solve POMDP more efficiently is an important research topic. Thrun et al. used a particle filter to represent the belief [85]. Then, the transition function is used only to model the transition of a single particle. This method is also known as point-based algorithm. Compacting the belief to obtain a simpler problem has been used to improve the value estimation [86]. In motion planning problems, there are some states that cannot be reached from a certain position, using this idea Kurniawati et al. build an efficient solver for robot motion that maintains the belief only on reachable states [87]. Instead of sampling from the belief, the solver samples states that are reachable by the robot. These improvements on offline solvers were insufficient to solve real-life problems, as the number of iterations required to find the optimal policy was too big.

Online solver has been found to adapt to many types of problems. An example of online solver is the Partially Observable Monte-Carlo Planning (POMCP) [88]. This method has been combined with reinforcement learning to win against humans playing Go [89, 90]. The value of each action is estimated by simulating the transition of a particle sampled from the belief for a time horizon. The action selection is made using an UCB1. During the tree exploration, the estimated value gets more accurate. Even if each node of the tree can be initialized without knowledge, previously obtained results can be used as prior estimation of the value. However, the search is only guided by the next action, whereas some actions should be performed in a certain order.

These models and recent solvers enable the application of POMDP in applications such as road intersection crossing. The following presents the elements of POMDP as applied at crossroad.

The description of the state-space is often similar for vehicle navigation. It includes the physical state of the vehicle and hidden variables to represent their behaviour. For example, drivers behaviour was separated between stopping, hesitating, normal and aggressive in [91]. These were inferred from a previously learned context. For the physical state, it is possible to work in continuous space [92]. However, their solution used a learned discretization to optimally divide the space into discrete values afterward. Drivers manoeuvre that could be enforced by the regulation in place is not considered in referred works.

Reward functions promote states and actions that make the crossing manoeuvre safe. Thus,

the reward function gives a high value to states after the intersection and negative reward to collision states [92, 93]. Acceleration changes or deviations from a reference speed may also be penalized to obtain better policies [93, 94]. These are not sufficient for the decision-making while interacting with humans.

Observation models are often similar, with Gaussian noises used to represent observation uncertainties [93, 94]. Reasoning with occluded obstacles has been achieved, but assumed a fix number of occluded obstacles in the scene [92].

Online solver allowed to scale simple problems, with one or two vehicles [94], into models with multiple vehicles driving at multi-lane intersection [93]. It also allows the implementation of POMDP for real-time application and on test vehicles [94]. Otherwise, simulated environments are used to test these systems. Another approach, with offline solver, is to decompose the problem into multiple simple ones [95]. Applied to road intersection, a single model is built each entity and solved offline.

The strong advantage of POMDP is the possibility to reason with different type of uncertainties (behaviour, measurement, occlusion). The model is also explicit in term of interaction between variables and judgment values. However, the crafting of these models requires many parameters to be tuned. Contrary to rule-based methods, errors in the models does not necessary result in dangerous situations but rather makes the evaluation of policies more complex.

2.4.3 Machine Learning

Machine learning uses data generated by vehicles to train models which are used in the navigation system. The recent availability of parallel computing hardware and annotated datasets have brought significant improvements on computer vision tasks with neural network. These have also been applied to the navigation and decision-making problems with success. These models have already been demonstrated in the 90's with a vehicle driving on the highway, solely based on cameras [5]. There exists three approaches to machine learning that have been used for navigation purposes: End-to-End, Inverse Reinforcement Learning and Reinforcement Learning.

End-to-end learning learns the relationship between sensors inputs and vehicle actions. The training dataset requires images labelled with drivers' actions. Bojarski et al. used a 72 hours dataset to train a Convolutional neural network (C-NN) to output a steering command [32]. They drove on the highway and unpaved road under various weather conditions. A C-NN uses

filters of different sizes to analyse the image, their outputs is processed in hidden layers and the last layer outputs the most likely action associated with the image, the structure of the network is shown in Figure 2.21.

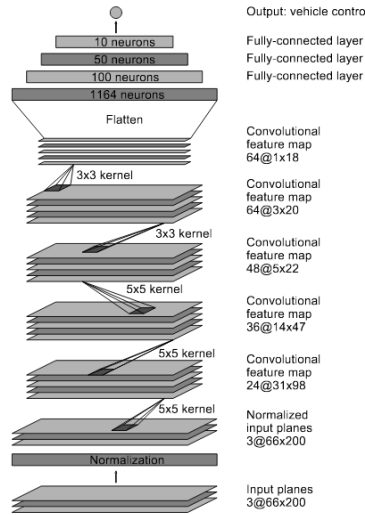


Figure 2.21 End-to-End neural networks structured that link inputs at the bottom to output at the top, after [32]

Their approach uses a single image to infer the next steering command. Thus, the output of the C-NN is a reactive action instead of long-term manoeuvre plans. The validation of such a black-box system is complex, as weights and features used in the hidden layers are difficult to be understood by humans.

Reinforcement learning (RL) and inverse reinforcement learning (IRL) are based on MDP but rely on machine learning to build the reward, transition and observation functions.

IRL aims to produce a human-like behaviour from demonstration made by experts. For autonomous driving, it requires driving demonstrations and the model learns the reward functions expected to explain what motivated the driver. IRL applied to ADAS improves the assistance by learning personalized driving style [96]. It has been used to reproduce lane change manoeuvre [97] and to understand interaction between drivers on the highway [98]. Reward models are easier to be learnt in these situations because contexts (lane markings, road shape) have many resemblances across the road network. At road intersections, learning these models is more difficult, as there exists various layouts, interactions and regulations.

Reinforcement learning analyses the consequence of a policy to improve its models and continues this process until the value of policies stops to improve. This requires the system to fail numerous times in order to improve, thus it is difficult to train in the real world.

For autonomous driving, it has been applied to highway driving [99] and intersection crossing [100] both in a simulated environment. MDP is used to model an intersection and learn the value function with a Deep Q-network [100]. Their agent managed to learn policies to move slowly and to avoid occluded vehicles. The model had to try 10 000 times for each of the four scenarios (turn right/left, multi-lane, straight) to achieve a 98% of successful crossing. Their method could be difficult to transfer into real life conditions. Their simulator (SUMO) uses simple kinematic models for vehicles and driver behaviour that does not accurately represent the real world.

In recent application of end-to-end and reinforcement learning, it has been demonstrated that simulation with some degree of validity can be used for training [101, 102]. They successfully drove on rural and urban road with an end-to-end network trained in simulation. Perceived images are coded into a latent vector which is decoded by a network trained to generate images as they could have been issued by the simulator. The decision process, that has only been trained in the simulated world, is able to use these fake images to generate action that are relevant to the real situation. Figure 2.22 shows the process flow of such a system and images used to decide.

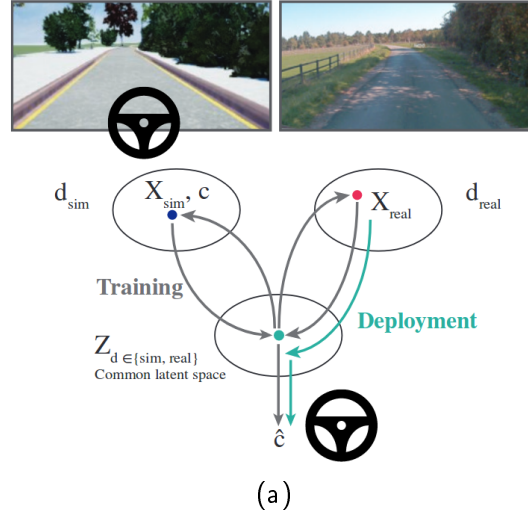


Figure 2.22 (a) Process flow of an end-to-end system for training and deployment, (b) images used by the network to decide with real perceived images (left) and the image transferred into the simulation space, after [101]

When these models are trained with measurement from real sensors or real situations, thus containing uncertainties, they can be robust against similar uncertainties but might under-perform in case the context has changed. The main reason is that while training the model looks for important features and compress the input representation. Thus, noises are filtered in the first layers of models as they are meaningless for the process. These machine learning algorithms are also more generalizable as the model searches for important features of the situation that are present across all the situation of the dataset. However, they are also prone to over-fitting if the dataset is not balance. For example, at road intersection if the model has been trained using data at peak traffic, thus containing many stop manoeuvres, the model may predict only stop manoeuvres even with low traffic condition.

Even if, machine learning models shows great performances on testing data left out the training dataset, their performance on new data or slightly different data (change country, weather) is difficult to predict. Simulation is regarded as an important way to generate training data with a controlled variance.

Hidden layers make the analysis and understanding of models a difficult task. The explainability of these models will be an enabler for their adoption for safety critical application such as autonomous driving [103].

2.4.4 Discussion

When compared in terms of performances, generalization, reasoning with uncertainty and explainability, none of the presented approach outperform the others. Table 2.1 shows strengths and weaknesses of each approach.

Table 2.1 Strengths and weaknesses of decision-making approaches

Approach	Uncertainties	Performances	Explainability	Generalization
Rule-based	- -	-	++	-
Probabilistic	++	+	+	+
Machine Learning	+	+	- -	++/?

- Rule-based method provides simple and understandable models which actions selection process can be understood and modified. However, the reasoning about uncertainties and interactions is done using pessimistic hypothesis. If an encountered situation is not within the predefined scope of the system design, rule-based methods cannot generate safe policies.
- Probabilistic approach uses uncertainty and interaction models to find policies that are the best trade-off between performance and risk. It requires human knowledge to build models, but unexpected situations are still considered as low probability events. Methods applied to solve probabilistic decision-making problems online approximate the optimal policy, there could be variance in approximate policies. Thus, each scenario must be tested multiple times to evaluation system performance and stability.
- Machine learning models improve on generalization and are robust to uncertainties. However, the training dataset must be balanced and cover the various traffic situations, these two objectives are difficult to achieve alongside. Most of the successful approaches use deep

learning at some stage, thus contain some hidden parameters that cannot be understood by humans. A posteriori parameter tuning is complex. It also affects their validation as conditions and reasons for their failure to decide correctly are difficult to understand.

The metrics used to evaluate these decision-making methods are often different. Cumulated reward is used with POMDP and RL to show the performances of the solver [92]. It does not demonstrate that the chosen policy is appropriate and safe. The number of collisions neither indicates the quality of the driving. RL may achieve close to zero collision, as these states are highly penalized. However, unexpected behaviour can be observed if part of the environment has not been explored [104]. Systems that used their performances indicators in their value estimation are likely to over-fit, further analysis of such systems might prove them unreliable [105]. Travel time by itself is not a good indicator, as it values acceleration over safety.

At road intersection, there are multiples solution to cross an intersection. To penalize a system because it chooses a slow policy and avoid unnecessary risk is wrong. Criteria based on the scenario can provide a better understanding of the system actions and allows comparison. It targets problematic scenario, that developers should focus on.

2.5 Conclusion

The vehicle navigation system is at the core of any autonomous vehicle architectures. To drive without an overseeing driver or wireless communication is complicated, as the system must reason with measurement uncertainties and be safe in any situations. At road intersection, the complexity rises due to interactions between entities and priority rules. Under these conditions, the behavioural planner is the most challenged, because it depends on this information to evaluate policies. At this stage, object lists with attributes and HD-maps are used to infer the context for behaviour understanding and decision-making.

Understanding the behaviour of drivers requires to contextualize the position of its vehicle with regards to the intersection and to analyse its motion. Machine learning methods is preferred for this task, because prediction of future states is already part of the POMDP transition model. A dataset that contains trajectories of drivers approaching an intersection and annotated with lateral and longitudinal manoeuvre can be used to train such models. These can be obtained via recording of real vehicle states or within a simulator [102]. Outputs of the behaviour understanding system remain uncertain because of human erratic actions.

Decision-making process uses these inputs to reason with interactions between entities and risk associated with the situation. Conservative approaches based on rules have difficulties to use this information. Whereas machine learning relies on a train dataset to find models for interaction and uncertainties, probabilistic approaches use model design by humans. The latter was preferred as the road intersection scenario rules and dynamics can be described. Reason for policy selection remains clear for human observer in these conditions.

In this thesis, probabilistic methods have been chosen to model the decision-making problem at road intersections. The direct modelisation of transition, reward and observation functions was preferred against machine learning because some elements of the model (vehicle dynamics, traffic rules) are known and can be directly coded into the model. A state representation as Lefevre et al. was chosen to reason with behaviours and risks [53]. Interactions was implemented as relation between driver manoeuvres. To provide a better evaluation of the system performances in different configurations, key performance indicators that depend on the scenario are proposed.

The wide variety of approaches to evaluate decision-making and the complexity to reproduce experiment make comparison difficult or impossible. There is a need for metrics and methods that better shows the performances of each methods.

3 Manoeuvre Classification Applying Learned Functional Discretisation

3.1	Introduction	67
3.2	Functional Discretisation	68
3.2.1	Motion Patterns With Gaussian Processes	71
3.2.2	Functional Discretization Framework	74
3.2.3	Discretization Results	83
3.2.4	HD-Map Storage	85
3.3	Manoeuvre Classification	86
3.3.1	Approach	87
3.3.2	Random Forest Classifier	89
3.3.3	Dataset	91
3.4	Results	94
3.4.1	Random Forest Topology	96
3.4.2	Functional Discretization Compared to Rectangular Discretization	98
3.4.3	Comparison With Baseline Approach	99
3.4.4	Results With the Hybrid Dataset	100
3.4.5	Discussion	101
3.5	Conclusion	104

3.1 Introduction

When crossing a road intersection in the presence of other vehicles, understanding the behaviour of the drivers is fundamental for the decision-making process. The spatio-temporal relationships are considered by observing the relevant vehicle motions and associated context. The former

3 *Manoeuvre Classification Applying Learned Functional Discretisation*

motion is observed via the on-board vehicle sensors. Whilst the latter is deduced projecting these observations on top of HD-maps.

This chapter presents the method proposed to analyse driver trajectories when approaching a road intersection. It allows to classify the longitudinal and lateral manoeuvres. The chapter is partitioned into three parts: The discretisation framework to contextualize the motion of the driver, the approach developed to infer driver manoeuvre through functional discretisation and results obtained by the combination of these methods.

Initially, surfaces that the vehicles will cross are segmented into two type of zones. The 'conflict zones' where the path of the observed and subject vehicle could overlap and the 'approaching zone' where the driver motion changes because of the road intersection crossing. This representation is projected and stored in a HD-map. Then, Random Forest classifiers are trained with features extracted from vehicle trajectories within a zone to classify driver lateral and longitudinal manoeuvres. The training phase takes advantage of a hybrid dataset made of simulated and real trajectories. It reduces the need to gather naturalistic data at the crossroad.

To conclude, the proposed approach is applied and compared with baseline methods such as support vector machine and other types of discretisation. These results are discussed to demonstrate the feasibility of the approach and to analyse its performance.

3.2 Functional Discretisation

Intersections are areas where different roads converge, thus the path of different entities using the road network intersects at different points. These are the zones where most of the 5000 junction related fatalities occur every year in Europe [18]. A typical crossroads intersection would have 16 potential collision points and 8 merging points as shown in Figure 3.1a. The introduction of roundabout divided by 4 these collision points as shown in Figure 3.1b. Further, roundabout geometry and lower visibility urge drivers to slow down, thus reducing the severity of accidents [20].

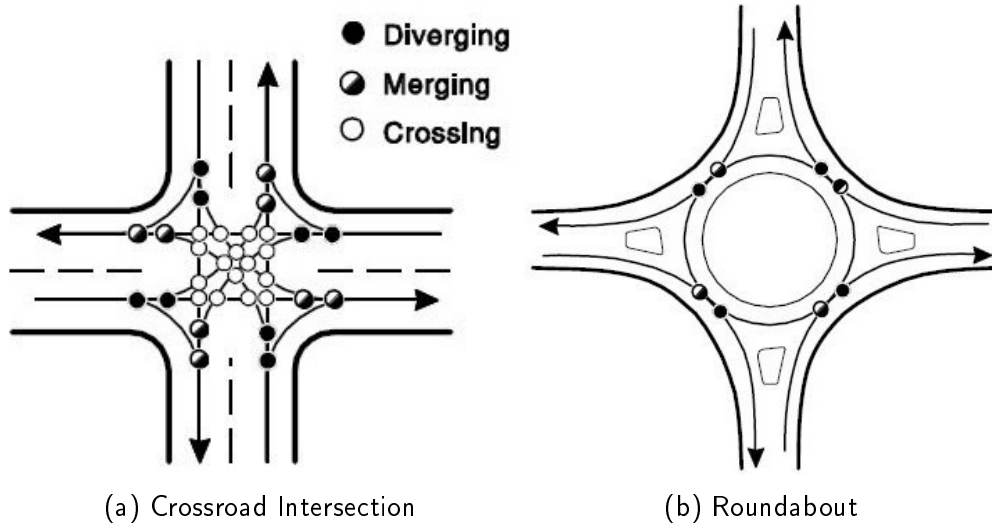


Figure 3.1 Vehicle conflict points comparison, after [21]

Two types of zones containing these points can be discerned: Crossing zones and Merging zones. The former is the result of vehicle paths crossing orthogonally as shown in Figure 3.2a. The latter is caused by the paths merging to reach the same exit branch as shown in Figure 3.2b.

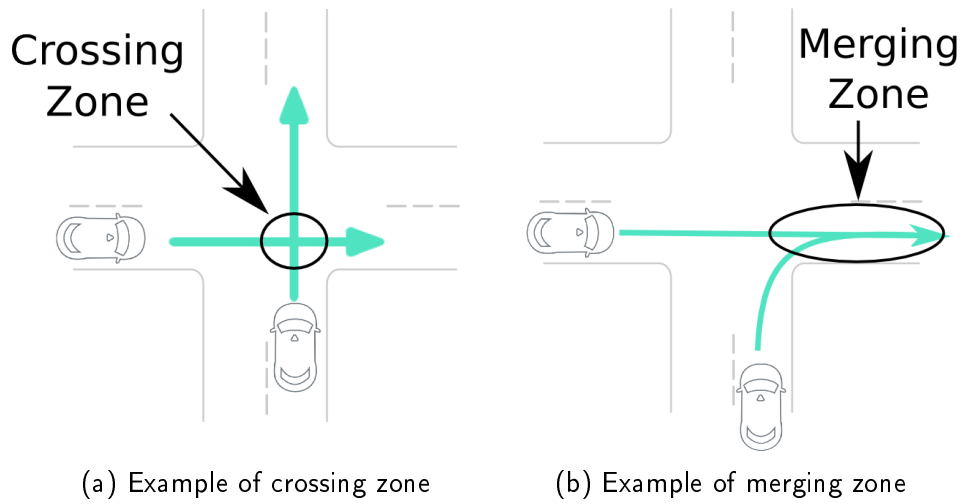


Figure 3.2 Examples of the two conflict zone types

Navigation maps are an important source of prior information to find them. They store knowledge

on driving context. They do not represent the manner humans drive, nor the approaching velocities as influenced by the presence of other vehicles. The estimation of conflict zones with lane centres is incorrect. For example, in Figure 3.3, orange and red polygons represent conflict areas estimated with lane centres represented in green. The path of a driver turning left at the intersection (in dashed violet stars) does not cross the conflict zones. This is caused by the area within the crossroad that does not have lane markings. That is, the freedom of motion of drivers is bigger.

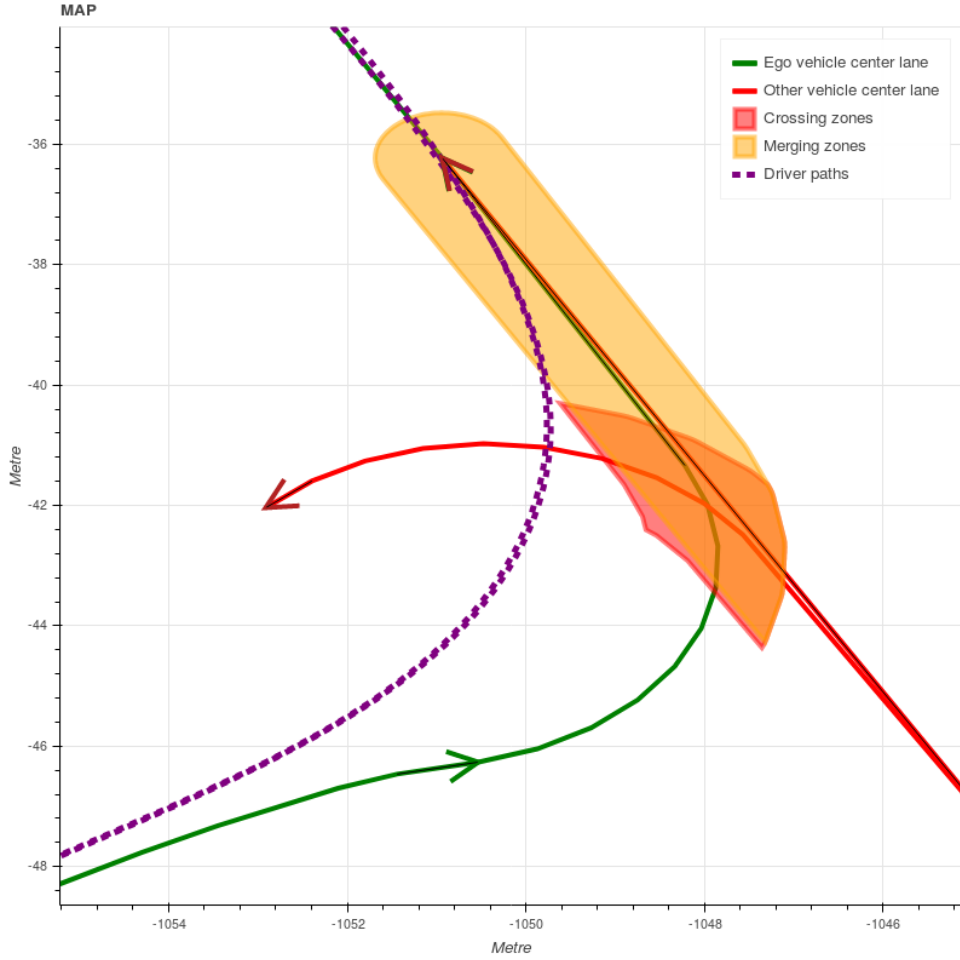


Figure 3.3 Comparison between the path of a human driver (purple), the path stored in the maps (green and red) at a T-shape intersection. This path is used to geometrically determines the crossing zone (red area) and the merging (orange area). The driver's path does not follow the lane center and avoid the crossing zone.

A road crossing an intersection can be partitioned into three areas: arrival (or entrance), crossing

3 *Manoeuvre Classification Applying Learned Functional Discretisation*

and exiting. A very important area is as the vehicle arrives, this is where the decision-making process engages the vehicle manoeuvre. Drivers need to be aware of their surroundings, any occlusion cause uncertainty and might force them to slowdown or to enter a hazardous situation. Drivers, as they arrive to intersections, must reduce their velocities as stated by the highway code even if they have the priority. The rational is to reduce collision risks. Drivers use the entrance area differently, some to slowdown, other to stop, or other to accelerate. This leads to different occupation rates of the entrance zone. These can be identified from driver trajectories and then used to infer behaviour. The crossing area is where drivers are not supposed to stay as the collision risk is high. Drivers decide to enter this area when they have the priority over other vehicles approaching and estimate that will have to stop there. Speed is low for vehicles that had to yield or to stop before crossing. The motion of driver is affected by the presence of other vehicles and the injunction not to remain in this crossing area. Thus, they are most likely to accelerate. Finally, crossing the exiting area ends the manoeuvre related to the intersection. However, this area remains important as pedestrian crossing or the next intersection might affect driver motion.

Representing driver motion at a road intersection is difficult. It does not only require to consider variations caused by different driving styles and manoeuvres. Non-parametric machine learning is suited to solve these issues [54, 106], however these approaches are computationally expensive as it requires an operation on the element of the dataset for inference. An alternative representation of these models could enable more practical usages.

HD-maps represent different types of information stored in a series of layers in a geographic information system (GIS) structure. Information includes road geometries, road attributes, as well as information from learning process. These maps can be generated using machine learning by observing the structure of the road [107]. Other machine learning applications used to understand drivers' motion can be analysed to create geometries. These by-products can be stored into the map. It allows to quickly access information about other driver behaviours based on a specific location.

3.2.1 **Motion Patterns With Gaussian Processes**

Vehicle trajectories result from a process that involves time. There are different ways to learn models that represent them [106, 108, 109]. Gaussian Processes are non-parametric machine learning methods that embed the dependency between time and vehicle poses whilst provid-

3 Manoeuvre Classification Applying Learned Functional Discretisation

ing a probabilistic approach for machine learning. A motion pattern, modelled with Gaussian Processes, is used to infer the likely vehicle poses at the crossroads for a given duration.

A Gaussian Process (GP) is a collection of random variables, where any finite number of which have a joint Gaussian distribution [110]. GP is a non-parametric model, thus the prediction step requires data points. It aims to retrieve the functional dependency $f(u_t) = v_t + \epsilon_t$ from a dataset $\mathcal{D} = \{(u_i, v_i) \mid i = 1, \dots, n\}$. For simplification purposes, the dataset is represented by two matrices. $\mathbf{U}$ is a $D \times n$ matrix that contains training inputs $\{u_i\}_{i=1}^n$ where D is the dimension of u_i and $\mathbf{v}$ a vector of size n containing observed values $\{v_i\}_{i=1}^n$. This dataset is used to represent the distribution of $P(v_* | u_*, D)$ where u_* an input and v_* corresponds to a prediction output. The GP is defined by its covariance $k(\cdot)$ (or kernel) and its mean function $\mu(\cdot)$. It can be written as $GP(k(\cdot), \mu(\cdot))$ with:

$$\mu(u_i) = \mathbb{E}[f(u_i)] \quad (3.1)$$

$$k(u_i, u_{i+t}) = \mathbb{E}[(f(u_i) - \mu(u_i))(f(u_{i+t}) - \mu(u_{i+t}))] \quad (3.2)$$

The mean estimation in equation 3.1 is the observed expectation of the process for an input. The covariance in equation 3.2 expresses how two inputs are related. Multiple kernel types can be used. The squared exponential covariance function is chosen as it makes close-by samples highly correlated, which is expected for motion patterns [54, 106]. This is written as:

$$k(u_i, u_{i+t}) = \sigma_n^2 \exp\left(-\frac{1}{2l^2}(u_i - u_{i+t})^2\right) \quad (3.3)$$

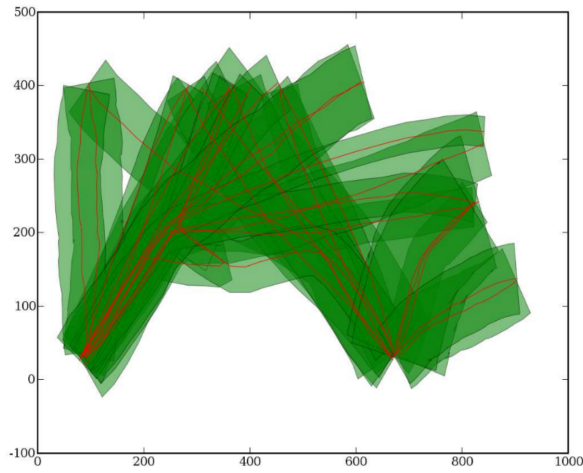
where σ_n the signal variance and l the length scale, they form a set of values $\Theta = \{\sigma_n, l\}$ called hyper-parameters. These are changed during the learning process to optimise log marginal likelihood.

A motion pattern can be defined as the probability of a moving object to occupy a position at certain time [108].

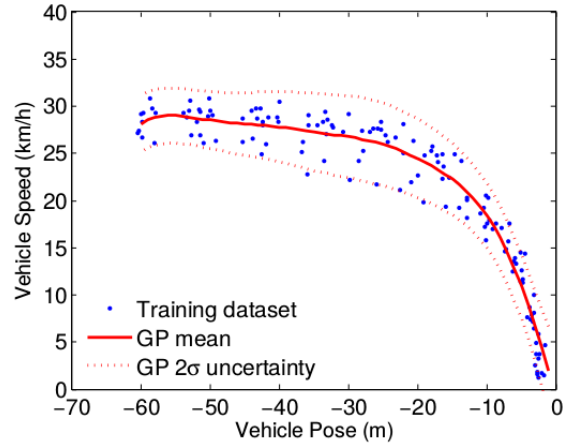
3 Manoeuvre Classification Applying Learned Functional Discretisation

This probability distribution for moving objects is not simple to estimate. It can be affected by goals, driver's interaction, driving style, etc. Thus, a learning based approach is often applied to retrieve this distribution using for example goals [111] or manoeuvres [106]. Trajectories are continuous functions, learning these motion patterns bares on regression problems. They are used in robotic applications to retrieve the dynamic models of unknown systems [112].

A Gaussian Process is a regression model that can be applied when changes are driven by Gaussian distributions. In the automotive domain, they have been used to represent driver paths [106], in Figure 3.4a, and to analyse drivers behaviour at stop intersections [54], in Figure 3.4b.



(a) Path estimations, after [106]



(b) Velocity profile regression at stop controlled intersection, after [54]

Figure 3.4 Application of Gaussian Processes on vehicle motion

We chose to train GPs to retrieve the motion pattern of vehicles. Compared to other regression models, like support vector regression, GP regression can represent the uncertainty around the predicted output. This allows GP to consider measurement uncertainty. The main drawback of GP is the complexity of the prediction step that grows with the number of training samples. To overcome this problem, training and predictions are done offline. Then, geometries encompassing the predicted motion pattern are extracted and stored in a map to be used at runtime.

3.2.2 Functional Discretization Framework

To find conflict and approaching zones, the proposed framework follows 5 steps, as shown in Figure 3.5 : step 1, the framework uses a dataset of trajectories grouped by direction to train GPs (step 2). Then, motion patterns (step 3) are used to determine relevant zones (step 4) that are then stored in a map (step 5). Next, details of each step are presented. We called this framework Functional Discretisation. The rational is that it divides the continuous intersection space into discrete areas by considering driver motion instead of a fixed discretisation step that uniformly divided the space. The resulting discretized zones are stored into the map. An example of the obtained zones is shown in Figure 3.6, with one branch segmented into 10 zones.

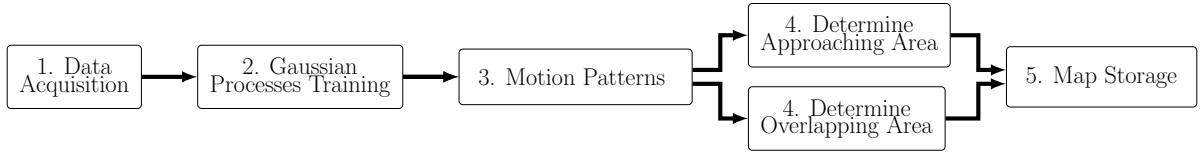


Figure 3.5 High level function of the framework with: 1) Acquisition of vehicle trajectories into a dataset. 2) Training by Gaussian Processes for each possible direction. 3) Generation of the motion patterns. 4) Application of patterns to determine the relevant zones. 5) Store results in a map

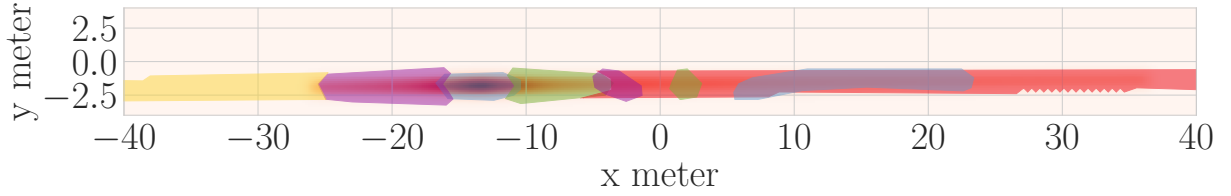


Figure 3.6 Zones resulting from the discretisation framework resulting of driver motion pattern approaching the intersection from left to right. The intersection center is located at $x=0$.

3.2.2.1 Dataset Acquisition

A dataset of trajectories is required to train GPs. To overcome constraints in obtaining a naturalistic driving dataset, a simulator was used to generate the dataset. SCANerTM is a simulator used in the automotive industry to test functions of vehicles (dynamics response, driver monitoring, HMI, etc.). Simulation models were built to include vehicle dynamics, traffic and intersection layouts. Dynamic models used in this simulator consider the size and weight

3 Manoeuvre Classification Applying Learned Functional Discretisation

of the vehicle. However, it is difficult to modify the driving style and to generate edge cases. The generation of driver behaviour considers the intersection layout and the traffic. Figure 3.7a shows a simulated four-way intersection, a yield manoeuvre is imposed to vehicles coming from the left.

In total 680 intersection crossing were simulated. Sampling rate was reduced to 10 Hz to match known perception systems and to avoid unnecessary data points¹. Samples of trajectories within 60 meters from the centre of the intersection are used. This distance corresponds to the distance where traffic signs should be present to warn drivers about the upcoming intersection. Vehicle motion starts to be influenced by the upcoming intersection from this distance.

Three vehicles were driven autonomously in the intersection to generate random situations. These vehicles had different dynamics due to different weights (two vehicles were small urban vehicle and one a heavy family vehicle). After a vehicle has crossed the intersection, it is removed from the simulation and reappears randomly in another branch. This generates randomness for each crossing, as sometimes vehicle with a higher priority is also approaching. The three vehicles are not always present at the same time at the intersection to vary the type of interactions. The velocity of each vehicle is controlled by Scanner traffic model. The lateral motion within the intersection is dynamically feasible. Samples of the recorded trajectories are shown in Figure 3.7. It can be observed that vehicle paths (Figure 3.7b) and velocity profiles (Figure 3.7c) are diverse. These differences are produced by models and manoeuvres that changed at each crossing. It makes the dataset suitable for training as it contains variance. The results obtained with these simulated data should be reproducible when GP is trained with real data as it will be trained with many different behaviours and Gaussian Processes consider measurement uncertainty added by real sensors.

¹Sampled at 100Hz, the position of vehicles driving at 50km/h would change by 13cm.

3 Manoeuvre Classification Applying Learned Functional Discretisation

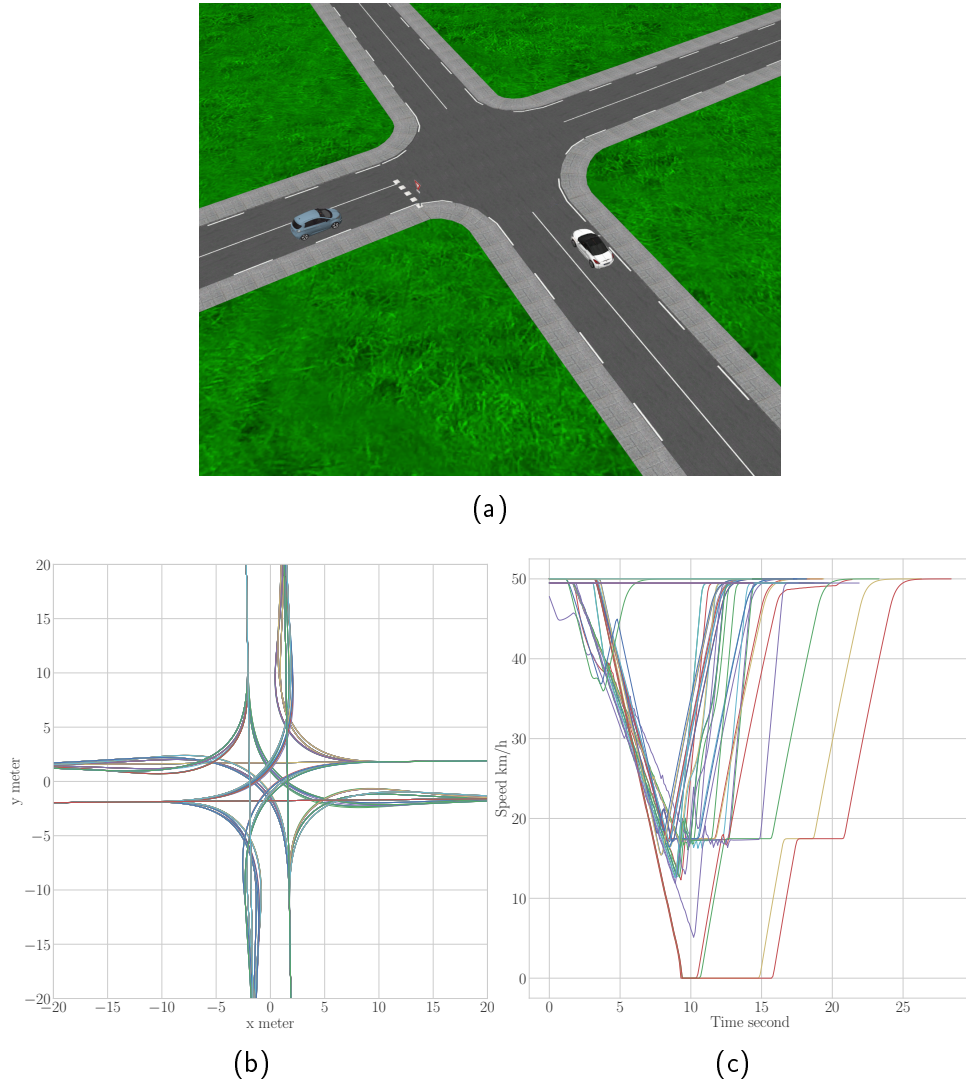


Figure 3.7 Sample of the dataset, (a) Simulated intersection used to generate the dataset, (b) Samples of paths in the dataset with pose variations due to the model of the vehicle, (c) velocity profiles of some vehicles driving in the intersection, with velocity variations due to the situation. (b) and (c) show different motions generated by the simulator.

Sampling time is generally used as input of GPs. However, each trajectory has a different duration. This corresponds to the behaviours that adapts the driver motion in response to the situation. Machine learning is more efficient when the input scale is the same for each trajectory [113]. For example, it takes less time for crossing manoeuvres when there is no other

3 Manoeuvre Classification Applying Learned Functional Discretisation

entities compared to yield manoeuvres, as the vehicle slows down differently. The distribution of the manoeuvre duration in the dataset shows that most manoeuvre durations are less than 20 seconds, and can range between 15 to 30 seconds, as shown in Figure 3.8.

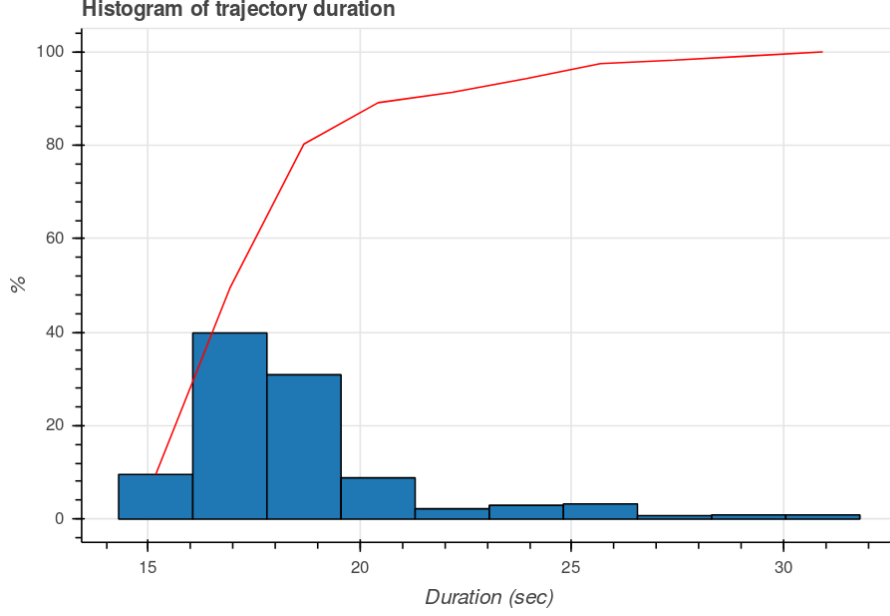


Figure 3.8 Histogram of trajectory durations in the dataset. 86% of trajectories duration is less than 20 seconds

In the literature, the distance to the intersection is often chosen to solve this problem [54]. However, this distance is irrelevant once the vehicle enters the intersection or turns left or right.

To consider different of trajectory durations, a normalization of the time vector is applied as follows:

$$T_i = \frac{t_i}{L} \quad (3.4)$$

with $i \in \mathbb{N}$ and $0 < i < M$, M is the number of samples, t_i the sampling time, L duration of the trajectory and T_i the normalized time.

Figure 3.9 shows positions and speed associated with two stop manoeuvres given time in Figure

3 Manoeuvre Classification Applying Learned Functional Discretisation

3.9a and normalized time in Figure 3.9b. It can be observed that the two trajectories look more similar using normalized time.

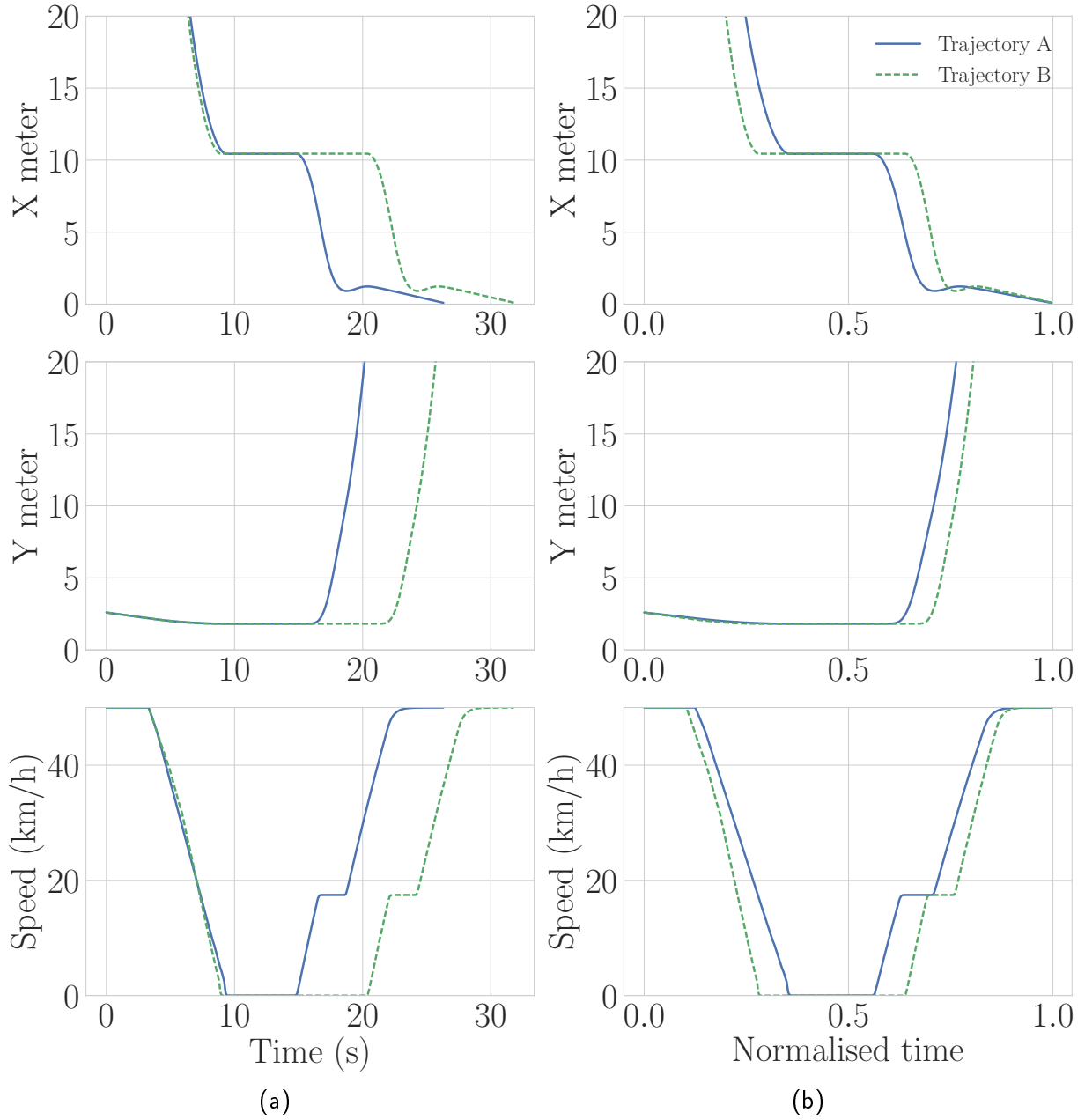


Figure 3.9 Measurements of the vehicle state against a) observation in time and b) normalized time. The top row shows the x position, middle row shows the y position and bottom row shows the speed profiles.

Inference of the motion pattern requires position and orientation to be used as feature vector, these are labelled:

$$Position : x_i, y_i \in \mathbb{R}^2 \quad (3.5)$$

$$yaw : \theta_i \in \{0, 2\pi\} \quad (3.6)$$

Vehicle positions are expressed in the intersection reference defined in Figure 3.10. Its origin is placed at the centre of the intersection with axes aligned along the access road axis. The vehicle reference frame is located in the center of the rear axle with the x-axis aligned with the direction of the motion of the vehicle and the y-axis points to the left. It defines the vehicle position. The position of the vehicle is by convention the centre of the rear axis.

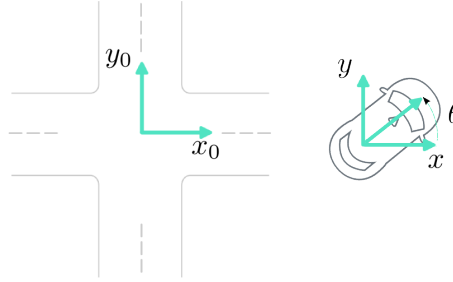


Figure 3.10 Intersection referential, the position of the vehicle is the centre of the rear axis

The final step on the dataset creation is to group trajectories into N clusters with respect to their origins and directions. For example, N is generally equal to 12 for a 4-way intersection with single lane entrances. This process uses the first and last sample of each trajectory to determine their group. Figure 3.11 shows these groups for the dataset.

3 Manoeuvre Classification Applying Learned Functional Discretisation

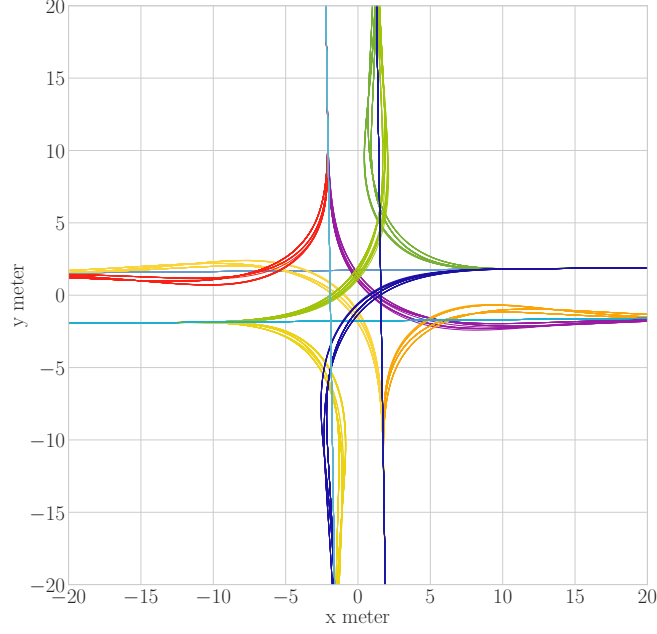


Figure 3.11 Clustered trajectories with respect to their origins and destinations, each colour corresponds to a different cluster

Let D denotes the full dataset to be used for training with:

$$\begin{aligned} \mathcal{D} &= \{(u_{i,n}, v_{i,n}) \mid i = 1, \dots, M; n = 1, \dots, N\} \\ \text{Where } u_{i,n} &= T_{i,n} \\ v_{n,i} &= (x_{i,n}, y_{i,n}, \theta_{i,n}) \end{aligned} \tag{3.7}$$

To simplify the notation, the vector of training input $u_n = \{u_{n,i}\}_{i=0}^M$, and the training output are in a vector $\mathbf{v}_n = \{v_{n,i}\}_{i=0}^M$

3.2.2.2 Gaussian Process Training

The learning process of a GP optimizes values for hyper-parameters $\Theta = \{\sigma, l\}$, that are the noise level and length scale to minimize the log marginal likelihood (in equation 3.8) of the posteriori distribution $P(\mathbf{v}_n | u_n, \Theta)$:

3 Manoeuvre Classification Applying Learned Functional Discretisation

$$\log P(\mathbf{v}_n | u_n, \Theta) = -\frac{1}{2} u_n^T K_v^{-1} \mathbf{v}_n - \frac{1}{2} \log |K_v| - \frac{M}{2} \log(2\pi) \quad (3.8)$$

where $K_v = k(u, u) + \sigma_n^2 I$ and $|\cdot|$ is the matrix determinant.

After this optimization process, the distribution of a testing input $\mathcal{N}(\mu_*, \Sigma_*)$ can be found by:

$$\mu^* = k(u, u_*) (k(u, u) + \sigma_n^2 I)^{-1} \mathbf{v}_n \quad (3.9)$$

$$\Sigma^* = k(u_*, u_*) - k(u, u_*) (k(u, u) + \sigma_n^2 I)^{-1} k(u, u_*)^T \quad (3.10)$$

To simplify the training process, each component of the feature vector is assumed to be independent. Thus, the motion pattern is the combination of 3 GPs in Equation 3.11, each trained with a single feature. It holds for trajectories as the position alongside the two axis is not dependent. The set of GP is used to represent the motion pattern as:

$$f(T) = [\mu(T), \sigma(T)] \quad (3.11)$$

$$traj_{pred,n}(T) = \{f_x(T), f_y(T), f_{yaw}(T)\} \quad (3.12)$$

with $T \in \mathbb{R}$, $0 < T < 1$, $n \in \mathbb{N}$, $0 < n < N$ and f_x, f_y, f_{yaw} the GP associated with each component of the motion.

Each GP is trained with a cluster of trajectories and optimized with the minimization method defined in Equation 3.8.

The motion pattern can be observed on a two-dimensional plane by marginalizing $traj_{pred,n}(T)$ with the equation 3.13.

$$\int_T traj_{pred}(T) dT \quad (3.13)$$

The motion pattern obtained from trajectories of vehicle approaching the intersection at the entrance that is controlled by the yield sign is shown in Figure 3.12.

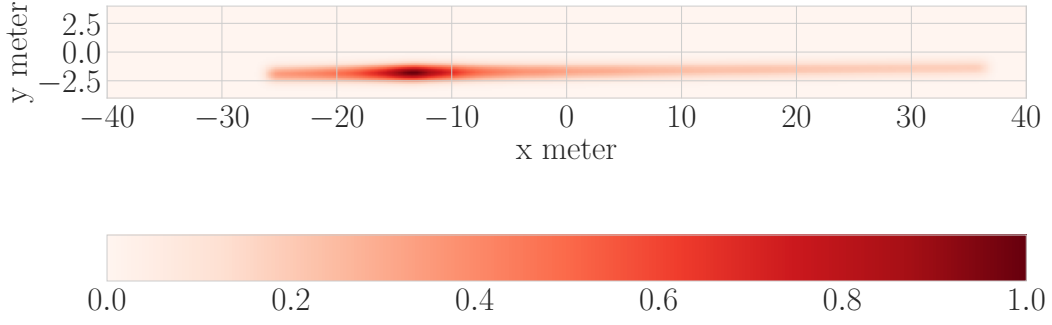


Figure 3.12 Motion pattern on a 2D plan after marginalization. The red colour intensity represents the probability associated to each position.

3.2.2.3 Discretization of a Merging and Overlapping Trajectory

Merging and crossing areas can be found where two motion patterns are likely to overlap. This is found by multiplying the marginalized motion pattern with each other marginalized motion pattern. The merging zones for the motion pattern n and a motion pattern j with $0 < j < N$ and $n \neq j$ is:

$$\int_T traj_{pred,n}(T)dT \sum_{j=0}^N \int_T traj_{pred,j}(T)dT > t_{cross} \quad (3.14)$$

To consider the vehicle size and shape, the resulting zone is geometrically grown by 0.75 metre. This value corresponds to half of the width of supermini vehicle.

3.2.2.4 Discretization of an Approaching Trajectory

Drivers go through different states while approaching an intersection [76]. This part of the framework identifies these changes by observing variation of the probable occupancy with respect to the previously trained GPs. Positions where the marginalize motion probabilities are similar are associated with a zone. Three heuristics are used t_{stop} , t_{slow1} and t_{slow2} . Each corresponds to a different likely occupation of the space. The zone where the vehicle is most likely to stop, is found where the marginalised motion pattern is higher than t_{stop} applying:

$$\int_T traj_{pred,n}(T) > t_{stop} \quad (3.15)$$

The same equation is used for the t_{slow1} and t_{slow2} thresholds. They allow for the identification of zones that are occupied while slowing down to adapt to the intersection speed or situation. Overlapping areas are assigned to the highest discretized zone.

These zones show a high mean value, as over the entire prediction time, they have been occupied longer. These zones are grown to consider vehicle size (as with the crossing zones).

3.2.3 Discretization Results

The motion of simulated drivers approaching a yield sign at an intersection is used to show the result obtained with the framework. The intersection is a typical crossroads with two streets crossing perpendicularly. It corresponds to the motion of a driver moving from left to right, as shown in Figure 3.7a. Trajectories from the lower branch are used to generate crossing and merging zones. Figure 3.13 shows the GP predictions after the learning process. The highlighted zone has a high probability of occupancy. It is the consequence of the yield sign that forces drivers to slow down or to stop in case another vehicle approaches the intersection.

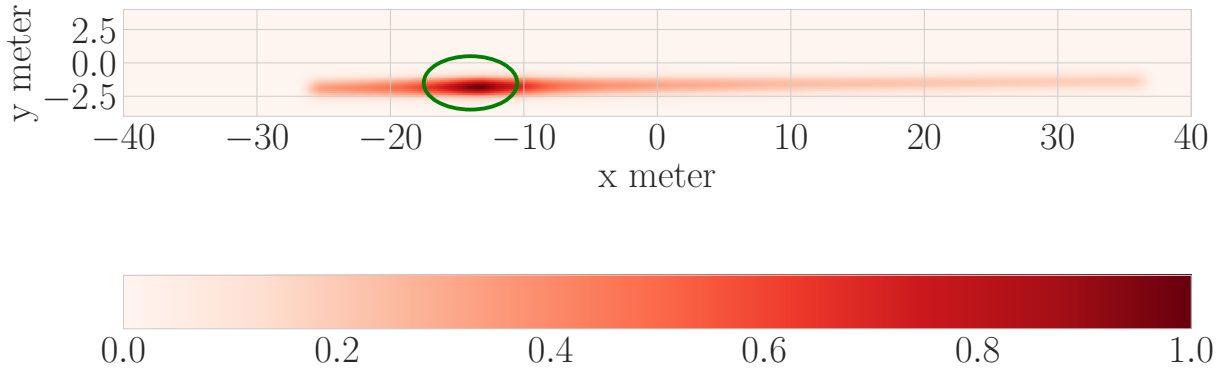


Figure 3.13 Map created with prediction from a set of GPs, the highlighted zone has a high mean probability due to the vehicle slows down.

The thresholds are empirically found and the values selected are shown in Table 3.1. These values hold for any clusters of trajectories of the dataset.

Table 3.1 Threshold values

t_{cross}	t_{stop}	t_{slow1}	t_{slow2}
0.3	0.8	0.4	0.1

3 Manoeuvre Classification Applying Learned Functional Discretisation

The results from Equation 3.14 are applied to define the three zones, shown in Figure 3.14. They correspond to merging (in red) and crossing (in blue) from a vehicle approaching from the lower part of the intersection zones.

The size and shape of zones depend on the uncertainty of the predicted motion pattern. Engaging the intersection while one of these zones is occupied may lead to a collision.

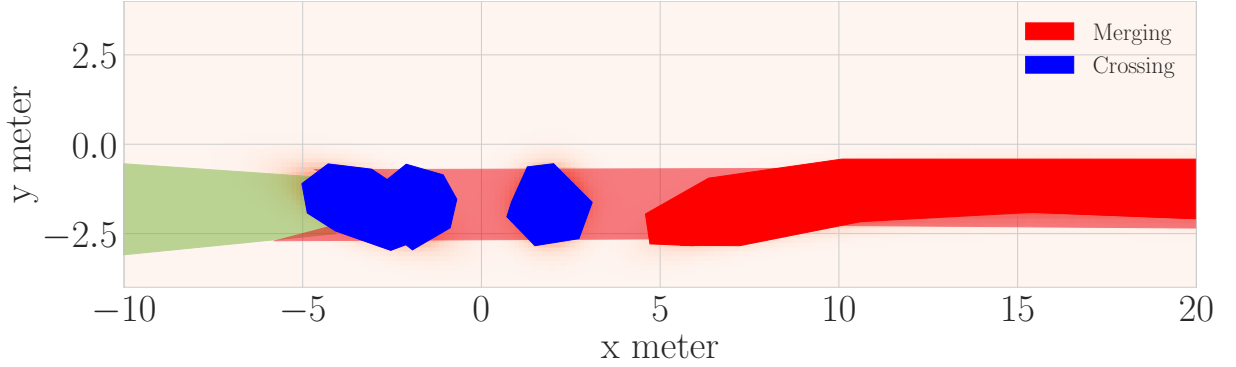


Figure 3.14 Blue zones are the crossing zones, zone in red corresponds to the merging zones

The Equation 3.15 applied to the motion patterns results in the discretized zones shown in Figure 3.15. The space to the left of the intersection is segmented into four zones (1,2,3,5). These zones correspond to the different states of a driver's behaviour while approaching an intersection. The fourth zone corresponds to the intersection and its exit. This zone has a low probability as drivers tend to quit it rapidly to avoid congesting the traffic.

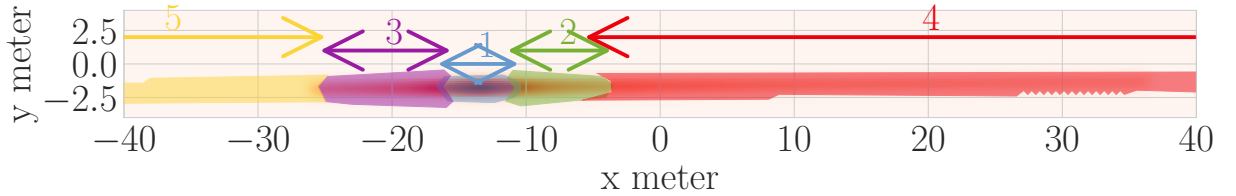


Figure 3.15 Discretization of approaching zones.

The combination of these steps generates 10 zones as shown in Figure 3.16. Application of these discretisations is made in the following section. It is compared with another type of discretisation for a manoeuvre classification task.

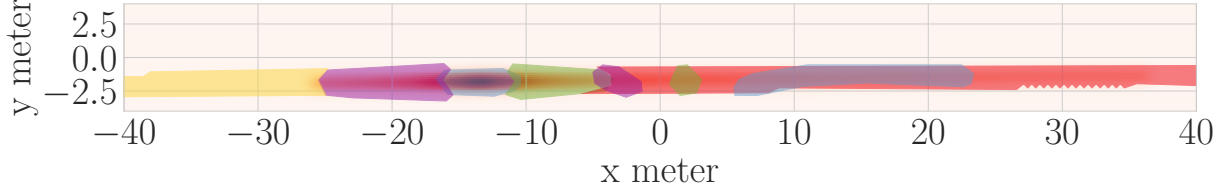


Figure 3.16 Final discretization for the proposed manoeuvre

3.2.4 HD-Map Storage

High-definitions maps are a core component of autonomous vehicle systems [114]. These are built from measurement made with dedicated platforms equipped with LiDARS and high-end localisation systems. These data are used to find the structure of the road and different elements of interest in the road network. As roads and static elements often change, maintaining these maps is costly, alternative solutions are sought in industry [102].

These maps are made of multiple layers as shown in Figure 3.17. The geometries obtained with the functional discretisation can be stored in the map priors layer. This layer is built through observation of a zone over time. It is generally used to store traffic light timing, probabilities of encountering a parked vehicle [114]. The decision-making process uses this information to adapt to local specificities, as two apparently similar intersections might need to be handled differently due to time of the day or occlusions, etc..

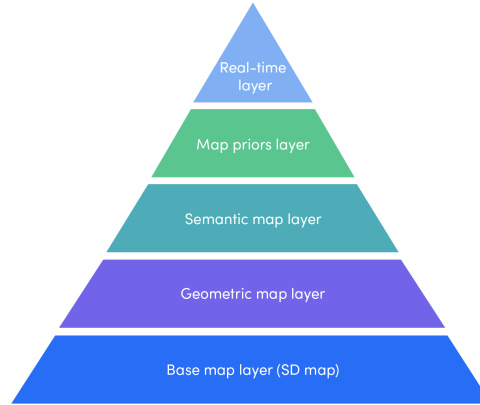


Figure 3.17 HD-map structure, after [114]

3 *Manoeuvre Classification Applying Learned Functional Discretisation*

Geographical Information System (GIS) provide a spatial database to store and structure geographic data. It is often used to build HD-maps. Data can be one or multiple points, lines or polygons in a referential to which information can be attached. For example, most HD-Maps include a list of lines associated that represents lane centres to which are associated speed limits and road connections.

To use these maps, spatial SQL queries can be made using the position of the vehicle to retrieve relevant geometries. GIS also defines a list of functionalities that must be implemented in maps. That include coordinate transformation, distance or relationship between geometries, further details are available at [115].

In our application, crossing and merging zones, that result from the discretisation, are stored as polygons associated to the ID of the roads that are crossing. Thus, instead of using the motion pattern, area can be accessed with a query to the data base. While driving, a query based upon the current road can be made to retrieve relevant crossing and merging areas. While planning AV motion, this zone can be monitored to decide whether or not to cross. Approaching areas are stored with their intersection entrance and exit. For the ego vehicle, they can be used to find the correct place to stop before crossing or to contextualize other vehicle motions within the other branch approaching zones.

3.3 **Manoeuvre Classification**

For a driver, to plan its action, the comprehension of another driver manoeuvre is very important. A driver's manoeuvre is influenced by context (e.g. traffic sign, road layout) and behaviour of other vehicles. An AV can interact with other road users, if it understands their intentions. Whereas, a human is able to interpret small clues of another human (i.e. head node, eye contact, etc.) as interaction acknowledgements, an AV can only observe vehicle motion. These make manoeuvre inference for machine a very difficult task.

Manoeuvring at a road intersection involves actions that change the vehicle speed and heading. These measurements associated with contextual information facilitate manoeuvre inference. Understanding interactions is difficult as access to every information on the situation is impossible. However, local context, inferred from information stored in the HD-map, provides an indication about elements that affect driver's manoeuvre and it is simpler to access.

Building models to analyse vehicle motion requires large naturalistic dataset [64]. These are costly, as their annotation requires human labour. The trained models might overfit on a specific

road intersection where it has been recorded, but also be influenced by local or driver population specification. To overcome these problems, simulated environment is often use to gather generic information but often lack of realistic human behaviour models [101].

3.3.1 Approach

For the purposes of this work, machine learning was applied to infer manoeuvre from drivers' motion and context. The dataset used to train models contains trajectories labels with driver manoeuvres. These can be divided into lateral (turn left, turn right, straight) and longitudinal (stop, yield, cross) manoeuvres, as defined in Section 2.3.1. Our classification approach aims to retrieve the dependency between motion features and these manoeuvres that act as labels.

The proposed classifier first infers the lateral manoeuvre, then the longitudinal manoeuvre. The rational is that the lateral manoeuvre has an impact on context understanding, thus constrains the motion associated with the longitudinal. For example, a driver turning left at a crossroad needs to slow to adapt its speeds to the curve, whereas going straight the speed is less reduced. Consequently, the same longitudinal manoeuvre has different motion features depending on the lateral manoeuvre. Furthermore, different priority rules are applied depending on the direction at the crossroad intersection.

The framework uses the previously described functional discretisation to get the context. For each area, multiple classifiers are trained with features from the trajectory dataset. Partition into zones means that the driver motion is meant to be simpler to analyse, thus performance should increase. This is demonstrated by comparing the performances of classifiers using these zones and classical rectangular ones. Another advantage of functional discretisation is to avoid the use of positions as features. Position is continuous and changes at each time step. It does not provide information about the manoeuvre but rather on the context by association with HD-maps. Its usage would add unnecessary information to the classifiers.

The proposed framework follows four steps, illustrated in Figure 3.18. The first step associates an extract of a trajectory with one of the zone, then a features vector is built from the extract. It is processed by a classifier specially trained to identify the lateral manoeuvre in this zone. Then, it is processed by another classifier trained to recognise the longitudinal manoeuvre. The output of the framework is an observation of the intended manoeuvre of a driver approaching an intersection.

3 Manoeuvre Classification Applying Learned Functional Discretisation

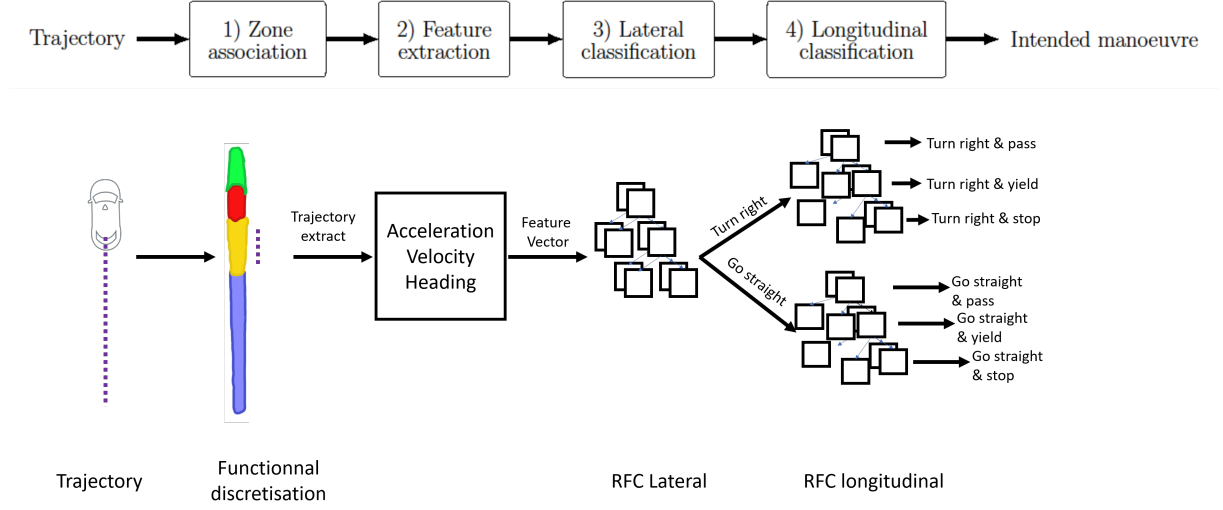


Figure 3.18 The classification framework structure used to infer manoeuvres at crossroad intersection

To generate the required amount of labelled data while keeping the cost low, the impact of using simulated data is studied. A hybrid dataset containing both simulated and real trajectories has been used to train classifiers. These are mixed in different proportions to study the impact of real and simulated data on the framework performance. It should be noted that performances are only studied by testing against real data.

The real dataset has been obtained by recording the vehicle response from a GNSS plus IMU from localisation system installed on a prototype vehicle. For the simulation data, the same setup as for functional discretisation is used.

An original contribution is the use of different classifiers for each zone of the functional discretisation instead of single one for the entire space. For each zone, a Random Forest Classifier (RFC) is trained to classify the lateral manoeuvre and two others for the longitudinal manoeuvre. This is equivalent to build larger forest, where each tree first tests the area where the sample was observed. The Figure 3.18 illustrates the framework structure.

This framework is applied to the classification of drivers approaching an intersection that is controlled by a yield sign on the approaching branch.

3.3.2 Random Forest Classifier

The machine learning algorithm retained for this work is the Random Forest Classifier (RFC) [116], which is meta-machine learning method that combines the output of multiple decision trees for classification or regression. The principal advantage of RFC is simplicity for training and testing, it requires simple operations. Thus, they are fast to train and to classify new samples. Training can be faster using the tree in parallel, as these are independent [117]. Compared to state-of-the-art deep neural networks, RFC models are understandable by humans. It allows for a clear understanding of classifier decisions and features used to build tests. It also requires less parametrization as only the maximum depth and the number of trees define the structure of the classifier. These are the reasons why RFC were chosen to classify drivers' manoeuvre.

A decision tree is composed of multiple nodes where simple tests are applied on the feature vector built from vehicle trajectories. These are learnt during training and used to guide new samples to a leaf node where it get assigned a probability that belongs to a manoeuvre class. The output of RFC is an average probability obtained as mean probability of each tree.

Formally, a tree j can be written as a function $f_j(x, \alpha_j) : \mathbb{X} \longrightarrow \mathbb{Y}$ with $\mathbb{X}$ the feature space and $\mathbb{Y}$ the class space. α_j captures stochastic elements of the tree (tests for each decision nodes) and x is the feature vector. Thus, the RFC can be written as $\mathcal{F} = \{f_1, \dots, f_N\}$ with N the number of trees and the probability of a class k given x defined as:

$$p(k|x) = \frac{1}{N} \sum_{t=1}^N p_n(k|x) \quad (3.16)$$

Two parameters are used to design a RFC: the number of trees T and the maximum depth of the decision tree. Increasing the number of trees should lead to better performances, however occasionally similar trees are added to the forest which do not improve the classification. The deeper a tree is, the more it generates pure nodes, though some branches are likely to overfit by creating many leaf nodes that classify only a single data point.

The training phase consists of finding the α_j for each tree. To avoid trees learning the same set of tests, each tree is fitted with a different subset of the data. This method, known as bootstrap aggregating or bagging, improves the stability and accuracy of the RFC.

Tests in a tree are found as follows: A group of features with different labels is presented at a node and multiple candidate tests are established. These are conditions on feature vectors (e.g.

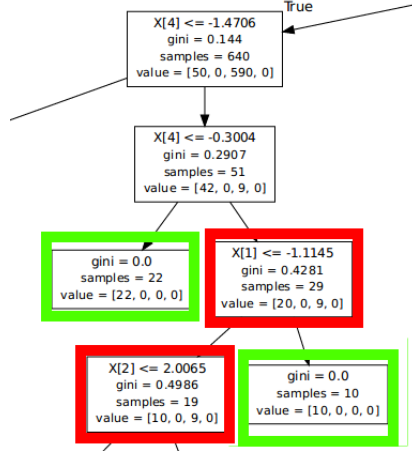


Figure 3.19 Example of tree classifier, focus is on a branch that generates pure nodes (in green) and impure nodes (in red). The first row of a node is the selected test, the second row is the Gini score of the test.

first feature is less than a value). Then, they are compared in terms of impurity that would result of splitting the group with the test. A pure node is obtained when, after a split, a group containing a single label is created. It also corresponds to a Gini score of 0. The Gini index is used to measure impurity. It is defined by:

$$Gini = \sum_{i \neq j} p(i)p(j) \quad (3.17)$$

with i the correct class and j another class. This index has a value of 0 when the test can differentiate completely a class.

The test with the lowest score is kept and used to split the data into two groups. Then, they are presented to two different nodes. The procedure is repeated until the maximum depth is reached or a pure node is created. Example of a tree branch generated from our dataset is given in Figure 3.19.

To infer the class of a new data sample, its feature vector is presented to every tree of the forest. In each tree, the feature vector follows the path given by the previously learned tests. When it reaches a pure node or a leaf node, a probability is assigned by the tree to the new sample. This single prediction is averaged by the result of all trees with equation 3.16.

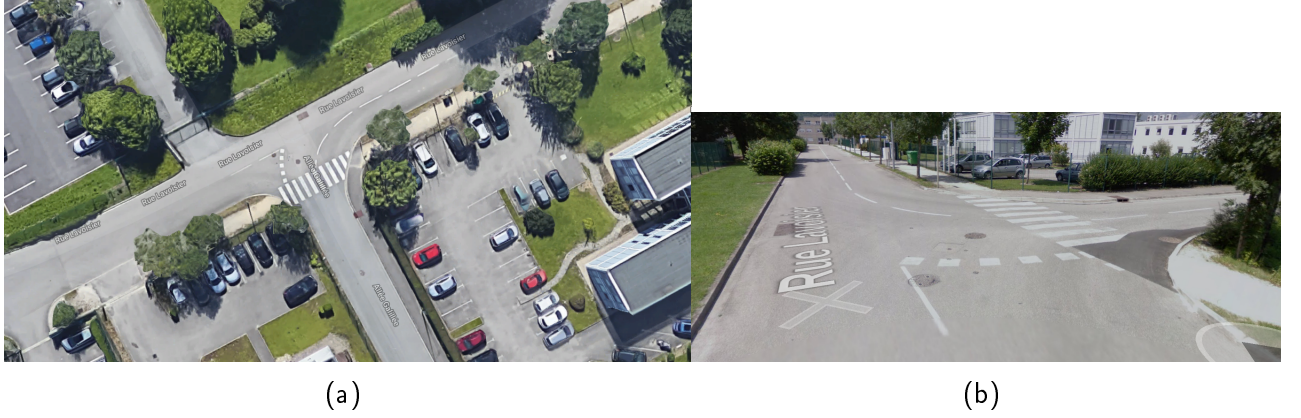


Figure 3.20 Satellite view (a) and ground view from the branch A (b) of the intersection. It can be found at $45^{\circ}13'02.2''\text{N } 5^{\circ}48'46.0''\text{E}$

3.3.3 Dataset

For the purpose of this research, two datasets were created. The first is constructed with real trajectories recorded from a vehicle at a three branches intersection. The second is made of simulated trajectories from a digital representation of the same intersection.

3.3.3.1 Real Environments

The experiment was conducted on a three-way intersection in an urban area (Grenoble, France). Figure 3.20 shows a satellite and ground view of the intersection used to acquire data.

This intersection was chosen because there were no other intersections in a 50 meter range, thus drivers are not accelerating approaching the intersection site. The speed was limited by the French law to 14 m/s (50 km/h). The traffic was relatively low (no more than two vehicles interacting together) and there were no pedestrians.

The vehicle state was measured by a high-end localisation systems, the X-sens (mti-g-710) that combines an inertial measurement unit with a GNSS. It measures the vehicle pose, speed, and acceleration at a rate of 100Hz. The measurement is used to represent the trajectory of the vehicle from which features are extracted.

Three different drivers controlled the vehicle for 30 minutes to have different driving styles in the dataset. In total 101 crossings were recorded, with 37 in the targeted branch (the one with

the yield sign). This is a relatively small dataset, further some data needs to be kept away for testing purpose. Therefore, the dataset was enhanced with simulated trajectories.

3.3.3.2 Simulated Environments

Simulated trajectories are obtained from the intersection Section 3.2.2.1. One branch was deleted to match the real intersection. Thus, zones obtained in the previous section can be re-used. In total, 100 crossings have been observed by driving on the branch with the yield sign.

3.3.3.3 Hybrid Dataset

The full dataset is made of 137 trajectories of driver crossing the yield sign. Each trajectory has been labelled with lateral manoeuvres (straight, turn right) and longitudinal manoeuvre (stop, yield, cross). The lateral manoeuvre is based on the final observation of a trajectory. The longitudinal manoeuvre labels depends on heuristics on the speed profile $S_{0,1,\dots,t}$:

$$\begin{aligned} \min(S_{0,1,\dots,t}) < 0.8m/s &\rightarrow T_{label} = Stop \\ 0.8m/s < \min(S_{0,1,\dots,t}) < 3.8m/s &\rightarrow T_{label} = Yield \\ \min(S_{0,1,\dots,t}) > 3.8m/s &\rightarrow T_{label} = Cross \end{aligned} \quad (3.18)$$

A trajectory was labelled as crossing when drivers kept a speed higher than 3.8 m/s. These drivers reduced their speed to obey the highway code that states that speed should be reduced in the surrounding of crossroads. When a lower minimum speed was recorded, another vehicle that has the priority was present thus the manoeuvre was labelled as yield. However, it was observed that drivers lowered their speed below 0.8 m/s to let multiple vehicles to cross the intersection. This manoeuvre is often named ‘slipped stop’. Thus, a lower bound was added to the yield manoeuvre to account for these exceptions and make them belong to the stop class. Examples of speed profiles and their classification are shown in Figure 3.21. It can be observed that there are resemblances between synthetic and real speed profiles. For example, deceleration and acceleration are similar. These classes could be enriched with perception system measurements in order to consider dynamic context and to create more classes. For example, at a yield controlled intersection, if another vehicle is observed in another branch of the intersection, the longitudinal manoeuvre can be labeled as a yield.

3 Manoeuvre Classification Applying Learned Functional Discretisation

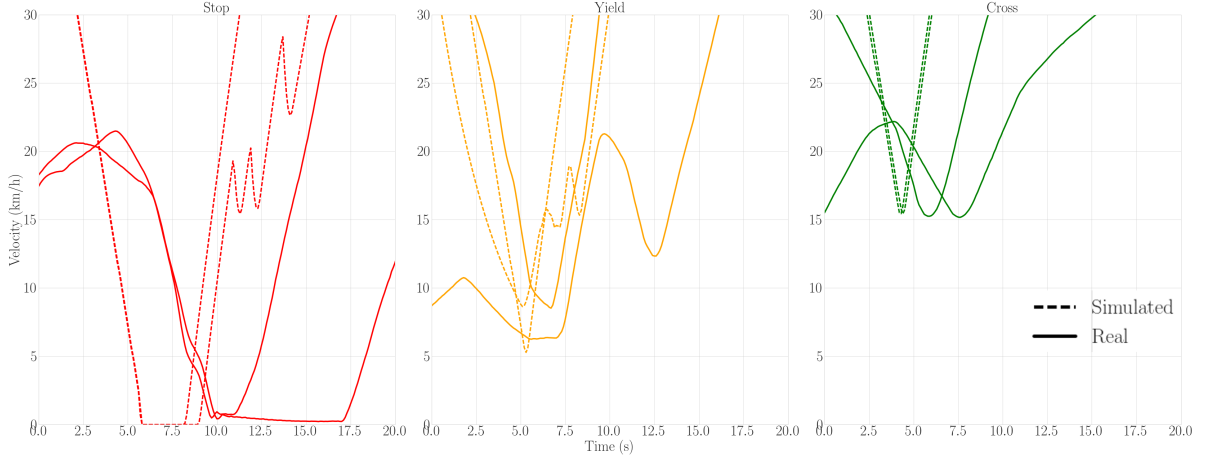


Figure 3.21 Example of trajectories with their given labels, dashed curves are synthetic speed profiles, plain curves are real speed profiles, horizontal lines correspond to each of the heuristics used to identify a class.

Table 3.2 Dataset composition

Label Dataset	Passages	Cross	Yield	Stop	Straight	Turn right
Simulated	100	43	37	20	39	61
Real	37	10	21	6	18	19

The final composition of the dataset is shown in Table 3.2. This dataset is unbalanced because classes are not present in the same proportion. A few numbers of stop manoeuvres were recorded in real life. Thus, simulation can be used to compensate for this weakness that would otherwise requires many observation hours. However, it makes the evaluation of the framework more complex. To address this problem, k-fold validation is used in Section 3.4, that is the classifier is evaluated k times with different parts of the data for training and testing. It helps to guarantee that classifier performances are not dependent on the training data.

3.3.3.4 Features

Features represent salient characteristics of trajectories that can be used to differentiate classes. Available measurements in our dataset are positions, speeds, heading angles and accelerations. Positions were left out of the feature vector, instead there are used to associate a sample to a zone. Thus, the feature vector was built from the group of measurements that belongs to the

3 Manoeuvre Classification Applying Learned Functional Discretisation

same area. To avoid sensibility to noise, the extrema of each measurement in a zone is used as features. Consequently, there are 6 features:

- S_{max}, S_{min} : Maximum and minimum speeds corresponding to extrema in the speed profile in m/s .
- A_{max}, A_{min} : Maximum and minimum accelerations corresponding to the extrema of the vehicle acceleration in m/s^2 .
- H_{max}, H_{min} : Maximum right and left deviation from the mean heading angle in radians.

To show the value of these features for the classification, Figure 3.22 shows the feature distribution along the diagonal and pairwise distributions. The training of the RFC consists of finding α_i , that are conditions on features in an automatic manner. It can be observed that acceleration features can be used to identify the cross manoeuvre easily, as there are two spikes in their distributions. However, there are some outliers that needs the contributions of other features to be correctly classified. For the other manoeuvres, overlaps can be observed in each feature distribution, but the combination of two features helps to determinate the class. For example, the combination of S_{min} and A_{max} can helps to classify a group of yield trajectory (highlighted by the purple circles in Figure 3.22) with two conditions on these features.

3.4 Results

To understand the importance of each implementation chosen, comparisons with other approaches are made. This is a three step process:

1. The optimized value for the RFC parameters are found by comparing multiple configurations.
2. Comparison between rectangular and functional discretisation is made, followed by a comparison between SVM and RFC.
3. Accuracies obtained with different dataset proportion are compared to understand the impact of simulated data during the training.

The implementation is made using python and Scikit Learn for training and testing [118].

3 Manoeuvre Classification Applying Learned Functional Discretisation

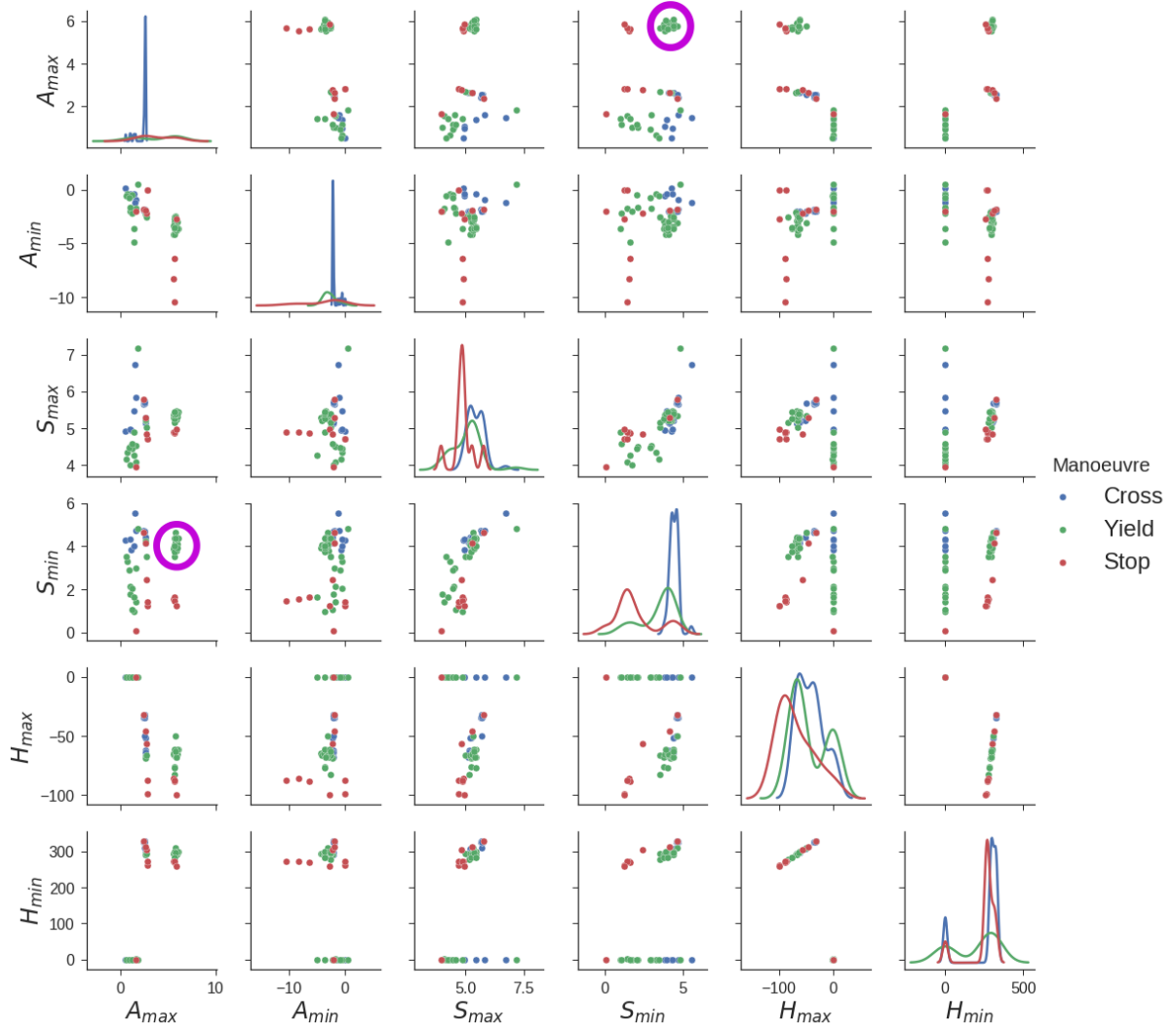


Figure 3.22 Feature pairwise relationships in zone 2 (blue zone of the Figure 3.24a). Areas highlighted in purple correspond to a group of yield manoeuvre that can be identified combining two features.

3.4.1 Random Forest Topology

The depth and number of trees are the two parameters used to control the creation of the RFC. Thus, it is important to find their optimized values. Out-of-bag error (OOBE) is commonly used to measure the quality of the learning phase of RFC. This metric represent the average error of each data point calculated with trees that did not use the data point for training. The lower the value of this metric is, the better the learning will be.

Multiple RFC with different configurations have been trained using a dataset containing 20% of real data. It favours a comparison of the different topologies to find where the OOBE is the lowest

Figure 3.23 shows the OOBE obtained with different sets of parameters. It can be observed that increasing the depth to over 10 nodes does not lead to significant improvements. For the number of trees, the OOBE decreased when 2-4 trees are used. It implies that a single tree would not be sufficient to learn behaviour patterns. This parameter has a lesser impact on the OOBE compared with the tree depth. It can be observed that performances stabilize below 0.2 with 10 trees and more.

A configuration having a maximum depth of 10 and 20 trees have been selected to construct RFC for the rest of the study. In this configuration an OOBE of 0.18 is obtained. It corresponds to the area highlighted in green in Figure 3.23. An OOBE score of 0.14 could have been obtained at the cost of increasing the depth over 25 which increases computation time and the likeliness of over-fitting.

Once a RFC has been trained, it is possible to count the number of times each feature has been used to split a node. It is used to determine each feature importance and how much a feature contribute to classification task. Table 3.3 includes results obtained with the selected configuration. For the longitudinal manoeuvre, features related to speeds and accelerations are the most important compared to the heading angle that was seldom used. V_{min} must have been used more often, as it was in the labelling process. It is noted that, the labelling process considers the global minimum of the speeds, whereas, the minimum in the feature vector is the minimum within a zone. The lateral classification has a more balanced usage of feature. This time, heading related features have more important than with the longitudinal classification.

The results show that each feature selected has an importance for the classification of the lateral and longitudinal manoeuvres.

3 Manoeuvre Classification Applying Learned Functional Discretisation

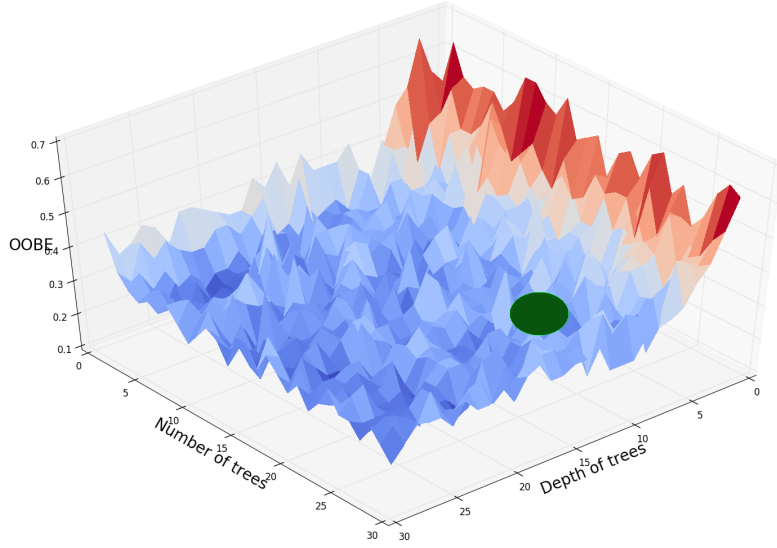


Figure 3.23 OOBE error with different parameters for the random forest classifiers. The green area represents RFC with 20 trees and a depth of 10 providing a 0.18 score (obtained from a dataset having 20% of real data)

Table 3.3 Features importance after training

Manoeuvre type \ Feature	S_{min}	S_{max}	A_{max}	A_{min}	H_{max}	H_{min}
Longitudinal	0.29	0.29	0.17	0.18	0.03	0.04
Lateral	0.21	0.20	0.15	0.18	0.13	0.13

3 Manoeuvre Classification Applying Learned Functional Discretisation

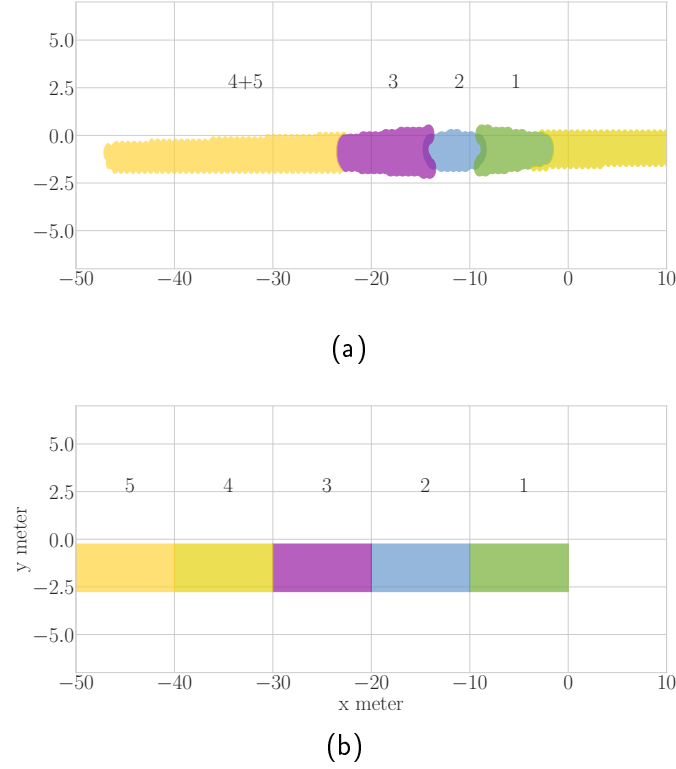


Figure 3.24 a) Discretization of the intersection space using the functional discretization.
b) Rectangular segmentation (the point 0,0 is the centre of the intersection)

3.4.2 Functional Discretization Compared to Rectangular Discretization

The performance of the classification framework should be improved thanks to the functional discretisation presented in Section 3.2. To understand its impact, a comparison is made with a rectangular discretisation. The latter is obtained by dividing the approaching branch into 5 zones that are 10 meters long and 2.5 meter wide. These dimensions are similar to the ones used by Gross et al. for a similar classification task [58]. Figure 3.24 shows a comparison between the two discretisation methods. The most left zone of the functional discretization is compared to zones 4 and 5 of the rectangular one because of its width of 20 meters.

Accuracy was computed as the result of the classification of real data, k-fold cross validation with $k=3$ and training made with a dataset containing 20% of real data. Table 3.4 shows the results obtained from each area (lateral and longitudinal manoeuvres). It can be observed that a global improvement of 5.4% for longitudinal manoeuvre and 5.8% for lateral detections was obtained.

3 Manoeuvre Classification Applying Learned Functional Discretisation

Table 3.4 Comparison of the classification accuracy using the functional discretization and a rectangle discretization

(a) Longitudinal manoeuvres classification accuracy

Discretisation \ Area	1	2	3	4	Mean
Functional	0.77	0.89	0.81	0.73	0.8
Rectangle	0.82	0.85	0.71	0.68	0.746
Improvement					+5.4%

(b) Lateral manoeuvres classification accuracy

Discretisation \ Area	1	2	3	4	5	Mean
Functional	0.84	0.93	0.77	0.71	na	0.81
Rectangle	0.84	0.89	0.67	0.67	0.70	0.752
Improvement						+5.8%

These results show that the functional discretization is more useful for the classification task compared to a simplified one. Reducing the size of the rectangular discretization could have led to better results but with more classifiers to be trained, thus a more complex system.

3.4.3 Comparison With Baseline Approach

RFC advantages are to be less sensible to noise and to variance in the data. This can be highlighted by making a comparison with baseline machine learning algorithm. Here, Support Vector Machine (SVM) was chosen. They have been applied with success for similar tasks [58, 64] and often served as baseline for many machine learning algorithms. Due to the multi-class description of the longitudinal manoeuvre detection, an “one-against-one” approach is used for SVM classification. That is a SVM is trained for each pair of classes. The result class is chosen by the maximal number of pairwise SVMs.

For this experiment, a hybrid dataset composed of 20% of real data and 80% from synthetic data was used with the functional discretization.

Table 3.5 shows the results obtained in each zone with K-fold cross-validation where k=3. For the longitudinal manoeuvre, the RFC performs as good as SVM in the area close to the intersection

3 Manoeuvre Classification Applying Learned Functional Discretisation

Table 3.5 Comparison between RFC classification and a SVM

	Discretization	1	2	3	4	Mean
Longitudinal manoeuvres	RFC	0.91	0.88	0.68	0.73	0.82
	SVM	0.91	0.88	0.62	0.69	0.80
	Improvement					+2.0%
Lateral manoeuvres	RFC	0.92	0.93	0.57	0.70	0.803
	SVM	0.81	0.70	0.60	0.68	0.712
	Improvement					+9%

centre. The RFC is better than SVM in the two zones furthest from the intersection. To determine the manoeuvre class the furthest from the intersection is important for decision-making. For the lateral manoeuvre, the RFC is over the SVM in most area.

These results show that RFC is more suitable to classify manoeuvres with our approach.

3.4.4 Results With the Hybrid Dataset

The final test is on whether or not the use of the hybrid dataset improves the training. The improvement of the classification framework is expected to improve the more real data are used. The functional discretization is used and K-fold validation with k=3 applied. The accuracy score is obtained with real data.

The evolution of the performances using different percentage of real data in the training set is shown in Figure 3.25. With the current size of the dataset, it was not possible to go over 20% as not enough real data would have been left for testing. These results indicate that the more real data are used the better the accuracy is. With no real data used for the training, the accuracy is little over 0.5 for the longitudinal manoeuvre. It means that the model train only with simulated data can recognize some manoeuvres. Thus, the two models (real and simulated) are not contradicting. For both manoeuvres, the accuracy increases when more real data are used. These shows that both the simulated and real data contribute to the classification performance. It should be noted that only a small proportion (here 10%) of real data is required to achieve good performance.

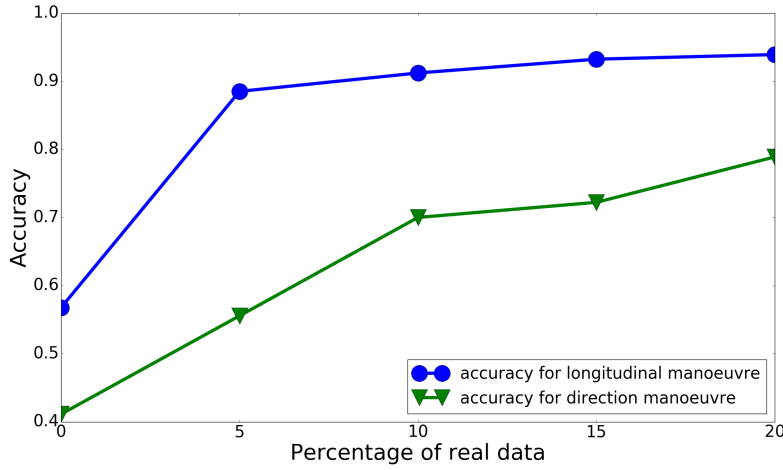


Figure 3.25 Accuracy of the classification using different composition of hybrid dataset, accuracy is obtained with real data testing

3.4.5 Discussion

Results showed that the proposed classification scheme using RFC, the functional discretization and the hybrid dataset should perform better than other combination of classifier, dataset composition and discretization. To assess the global qualities of a classifier, receiver operation curves (ROC) are used to compare performances. These are shown in Figure 3.26. The top left point (coordinate (0,1)) represents optimal performance with only true positives and no false positives. The closer the curve is to this point, the better will be the classification. Performances can be determined using the area below the curve. The steepness of the curve is also important, it shows how fast the true positive rate increases. The curve is obtained from average results of all the local classifiers. For the multi-class longitudinal manoeuvre classification, curves are an average of the ROC for every class. Results, in Figure 3.26 were achieved on real measurements. It experimentally shows that the chosen implementation is better than other approaches. It also shows, it is the combination of the two approaches that work best as the use of the functional discretisation does not improve the classification with SVMs. These results are obtained by comparing the classification of real trajectories left out during the training with their known labels.

The Figure 3.27 for different composition of dataset. The solution using only simulation data can still perform better than a random guess (dashed blue line), thus information from simulation is useful in the learning. The addition of a certain percentage of real data improves the classification. It shows that the information provided by data from simulation helps the classification. Only the

3 Manoeuvre Classification Applying Learned Functional Discretisation

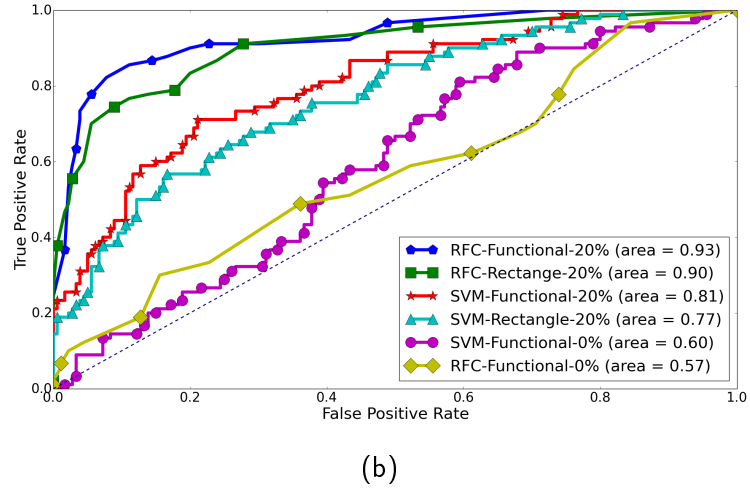
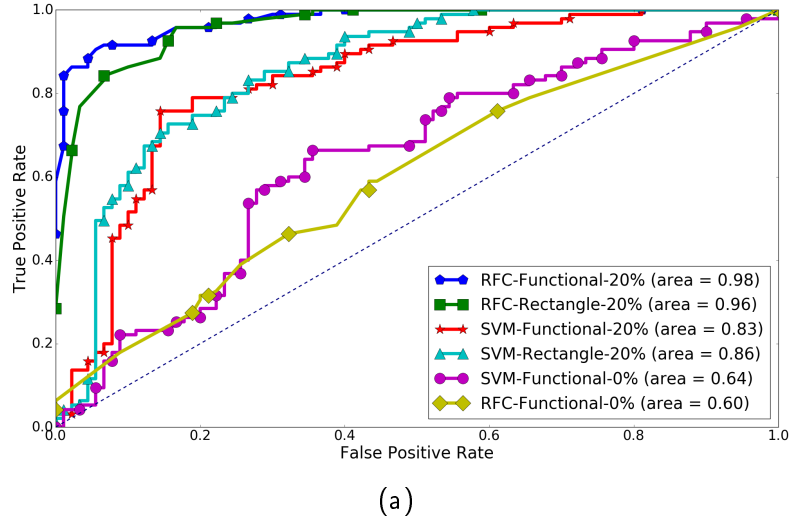


Figure 3.26 Receiver operation curves a) lateral manoeuvres; b) longitudinal manoeuvres. Dashed lines correspond to a random classification.

3 Manoeuvre Classification Applying Learned Functional Discretisation

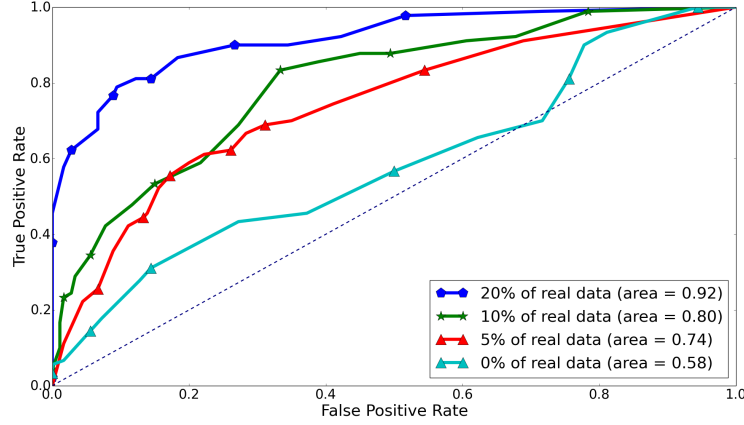


Figure 3.27 Receiver operation curves obtained from different compositions of the dataset for longitudinal classification. Testing is made using real data.

ratio of synthetic and real information has been discussed in this section. For our experimentation the size of datasets were relatively low, especially real data. If more simulation time is spent, the performance of classification could be increased. However, as pointed with these results, the use of even a small amount of real data increases performance.

The use of the functional discretization has improved classification. This discretization takes into account where drivers are most likely to adapt their trajectories to local context. Thus, the resulting classifiers are fitted to a more relevant feature range. For example, leaving zone 4, the driver is expected to slow down to adapt his speed to the intersection and then entering zone 3 slowing down, if a yield or a stop manoeuvre is required. Zone 3 for a rectangular discretization is between two zones of the functional discretization, the consequence is a lesser accuracy of the manoeuvre classification than using the functional discretization. It is caused by the driver being in a transitional state and features show more change in the third rectangle.

The use of RFC rather than another classification method shows a better steepness and a wider area below the ROC curves. Thus, outperforming the implementation with an SVM. RFC is known to be robust against over-fitting, it is possible that the SVM over-fits on the simulated part of the dataset whilst the RFC does not. The computation time is sufficient for real time implementation with sufficient margin to add more trees if required. A more advanced strategy could be used to train the RFC for on-line learning. For example, starting from a forest learnt with only synthetic data, some trees could be replaced by newly trained trees with real data. This would enhance the management of classifiers across its use.

3.5 Conclusion

This chapter presented a method for manoeuvre classification applying a functional discretisation. The latter uses simulated vehicle trajectories to train Gaussian Processes. These are used to represent the motion pattern of vehicles at a crossroad. Because of GPs complexity, these motion patterns are stored within an HD-map as polygon delimiting the intersection entrance and crossing zones. Their use is shown to improve classification results (Section 3.4.2)

The classification aims to identify the driver manoeuvre from its trajectory. A set of Random Forest Classifiers is trained for each of the zone of the functional discretisation. It creates classifiers that are fitted specific zone of the intersection. RFCs are trained with a hybrid dataset made of simulated and real trajectories. Both contribute to increase framework accuracy as shown with good results obtained with 20% of real data (Section 3.4.4). This contributes to reduce the cost involve of recording and maintaining a dataset containing only real data.

This framework enriches the perception data with the driver current lateral and longitudinal manoeuvres. These are part of the observation obtained by the decision-making system about the environment. Knowing the performance of the classifier simplifies its models into the observation function of the POMDP.

It is important to notice that this study uses trajectories directly recorded on the vehicle and not perceived trajectory. During the dataset recording in the real intersections, a LiDAR based perception from Rummelhard et al. was also recorded [119]. The classification results were unstable with rapid changes of classification. The likely cause is the difficulty to estimate the centre of another vehicle from its perceived bounding box. In real conditions, the bounding box only covers a small part of the other vehicle, thus the estimated centre is shifted and changes as different sides of the other vehicle are perceived. For example, we observed that the estimated vehicle centre lied in the wrong area, thus the wrong set of classifiers were used. Because of the complexity to test the decision-making system in the next chapter, simulation is preferred. In this condition, the simplified perception system uses the other vehicle centre as observation. it allows for the classification to be usable.

A pertinent perspective is to improve the classification by modifying the set of RFC whilst new real trajectories are recorded. Instead of retraining the classifiers, only a subset of trees could be trained with the new data. Their addition to the forest would be conditioned to an improvement of the results, especially on data that were previously misclassified. The functional discretisation would benefit for an automatic method to identity thresholds. Bayesian change points has been

3 Manoeuvre Classification Applying Learned Functional Discretisation

identified as a possible solution [120]. It estimates when an observed time related process has undergone changes in its model. This could be applied on the learned motion pattern or on trajectories.

The hybrid dataset could be improved by adding another recording modality: ask human drivers to drive in the simulated intersection. This solution has been used to compare trajectories generated a controller and humans in similar situation to decide which is the most valuable [121]. These trajectories would be close to real life trajectories without the burden to record them at the intersection. Dangerous situations could also be recorded to add more variance in the dataset

4 Probabilistic Decision-Making, Applied to Crossroad Intersections

4.1	Introduction	106
4.2	Context and Integration into Vehicle Centric Architectures	107
4.2.1	Road Intersection Crossing Scenario	107
4.2.2	Integration of the Probabilistic Decision-Making into the AV Architecture	110
4.3	Decision-Making Framework	112
4.3.1	State-Space	114
4.3.2	Action-Space	114
4.3.3	Transition Model	116
4.3.4	Observation-Space	121
4.3.5	Observation Model	121
4.3.6	Reward Function	123
4.4	Partially Observable Monte-Carlo Planning: An Online POMDP Solver	127
4.4.1	POMDP Solver	127
4.4.2	POMCP Elements	129
4.4.3	POMCP Algorithm	131
4.5	Conclusion	134

4.1 Introduction

Once the vehicle situation is understood with respect to context, it is possible to decide which vehicle action is the most appropriate in order to navigate safely and attain its destination. That

is, the vehicle decides what it is going to do next by considering the likely consequences that its action will have on the environment and whether or not it satisfies navigation requirements.

The decision-making process is difficult due to uncertainties associated with the vehicle perception system, situation understanding, other driver behaviour and interactions with other vehicles. Further, evaluation of the vehicle action requires to balance performances and risk, that are often divergent issues.

The objective of this chapter is to propose a probabilistic decision-making model applied to the road intersection crossing scenario. This requires understanding the context to which this system is applied and how it can be integrated within the vehicle architecture. To address behavioural uncertainty which is a major issue, the proposed model integrates observation of the behaviour understanding framework presented in Chapter 2.

For this purpose, Chapter 4 is divided into two sections. The first presents the scenario upon which the decision-making is applied by describing the intersection layout, actors involved and their expected behaviour. It includes, the manner into which the decision-making mechanism can be integrated into a typical AV system architecture. The second section presents the decision-making framework modelled as a Partially Observable Markov Decision Process (POMDP). Each element of the model is presented, emphasis is made on the specific design of the reward function that uses behavioural variables.

The result obtained with the proposed solution is presented in Chapter 5.

4.2 Context and Integration into Vehicle Centric Architectures

The decision-making process differs with respect to the application scenario and systems interacting with it. This section explains the specificities of the studied scenario and the integration of the associated decision-making into the vehicle functional architecture.

4.2.1 Road Intersection Crossing Scenario

The risk associated with the crossing of an intersection has been explained in Section 1.2 by looking into accidentology and generic decision problems. We restricted the focus of this study to the cross-cutting scenarios involving two vehicles, as shown in Figure 4.1. This scenario is

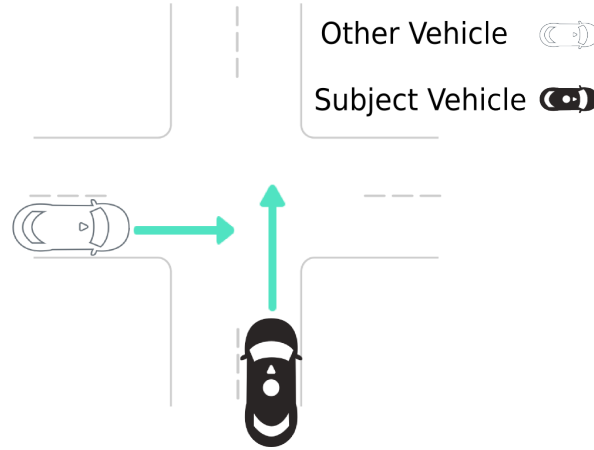


Figure 4.1 Crosscutting road intersection scenario with the two involved vehicles.

complex as the two vehicles are on a collision path. Drivers can obey or not traffic signals, there could be occlusion due to other vehicles stopped at such intersection, etc.

The layout of the intersection is like the one used in Chapter 3. The two roads cross perpendicularly without obstacles blocking the driver's sight. It is assumed that, there is no other intersection in a 50m radius to guarantee that driver behaviour is only affected by the studied intersection. Drivers are warned about the upcoming intersection and the priority that shall be applied by a traffic sign. Three types of traffic signs are considered: stop sign, give way sign or crossroad sign (indicates that the vehicle is driving on a major axis and has priority).

The scenario involves two vehicles: the Subject Vehicle (SV) and the Other Vehicle (OV). The SV is the autonomous vehicle, equipped vehicle centric systems including the proposed decision-making mechanism to plan its actions. The OV can be either an autonomous vehicle or human driven, however this information is unknown by the SV. The goal of both vehicles is to cross the intersection and to continue on their current roads.

In this scenario, the vehicles are on a collision path. This can be avoided if both respect the rule indicated by their respective traffic signs and take actions accordingly. However, there is much uncertainty to consider in this scenario. First, the SV gets observations of the situation that does not correspond to the true state of the environment. This is due to uncertainty inherently present in its perception system and vehicle state estimate. Second, the manoeuvre intended by the OV is not necessarily the one expected by the situation. Consequently, its motion is difficult to predict and to understand. The model presented in Section 3 aims to integrate these uncertainties in the decision-making reasoning process.

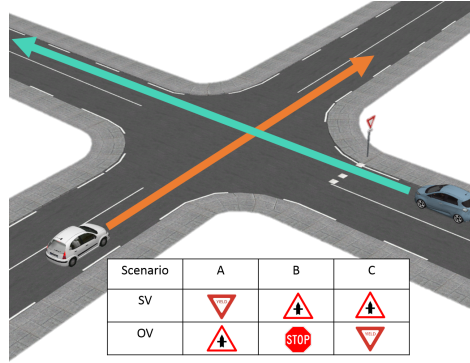


Figure 4.2 Different scenarios and associated traffic signs for the crosscutting road intersection scenario

Two groups of scenarios can be identified:

- Scenario A: the OV has priority over the SV. The case where the SV has to stop due to a stop sign is left out. Because the decision-making process is simpler as it is assumed that the SV shall always obey to the stop sign. The stop manoeuvre for the SV can be partitioned into two phases: slowing down to stop and waiting for the OV to cross. The decision-process does not need to understand the behaviour of the OV, only the observation that the vehicle has crossed is important. The other scenario, where the SV has to yield, requires to plan actions to slow down to let the other vehicle cross, or to stop. If the OV for an unknown reason has decided to slowdown. This requires understanding the OV behaviour and considering uncertainty associated with the estimation of the vehicle state.
- Scenario B and C: they consider the SV as having the highest priority. The OV can be expected to stop (scenario B) or to yield (scenario C). In these scenarios, the SV is expected to slow down as it approaches an intersection without stopping, as it has the priority. However, the OV might not be cooperating or interacting with the SV and could try to cross nevertheless. Thus, actions of the SV shall slow it down until it estimates that, it is safe now to cross the intersection. In some cases, the SV might need to stop because of an unruly driver in the OV. The difficulty for the SV is that it needs to balance all the three manoeuvres to consider behaviour uncertainty of the OV.

The proposed decision-making process needs to generate the adequate actions to enable the vehicle to cross the intersection in the best possible manner. The three studied scenarios have different solutions depending on the decision-making sensitivity to risk and uncertainties.

4.2.2 Integration of the Probabilistic Decision-Making into the AV Architecture

To solve the aforementioned scenarios, a Partially Observable Markov Decision Process (POMDP) is used (c.f. Sub-Section 2.4.2 for the definition). It uses available inputs to generate action sequences (or policy) and for their evaluation. Then, the solution with the highest value is to be fulfilled by another sub-system. These processes can be integrated following different structures into the vehicle functional architecture. The chosen structure is presented in this section together with the rationale for choosing it.

Three types of inputs are used by the decision-making process for the crosscutting scenario. These are as follows:

- From the perception and vehicle state estimation systems, it receives observations of the other vehicle state and its own. These are subject to measurement uncertainties (e.g. measurement noises, low accuracy.)
- HD-map allows for the contextualisation of these observations with respect to the road layout. The layout of the road is obtained using high-end technologies automatically processed and manually checked. Traffic signs and associated priority rules are carefully reviewed after the map creation. Thus, the uncertainty associated with the map is far lesser than the one associated with the perception system. In our application, no uncertainty is considered with the HD-map information.
- From the situation understanding, it gets observation of the OV behaviour. This is the result of applying the framework developed in Chapter 3. The uncertainty on this information is estimated from the classification performances as estimated in Section 3.4.

A POMDP uses these observations to reason with a probabilistic estimation of environment true state. The POMDP model described the states, relation and rewards that correspond to the scenario. This is used by a solver to generate policies π and to evaluate them. Policies are sequences of actions. In theory, it can find a single optimal policy π^* . However, because of the large number of state variables (c.f. Sub-Section 4.3.1) and the short period of time during which the estimation needs to happen, only an approximation of policy values can be obtained. This constrained the integration into the vehicle architecture as the output of the decision-making might vary in uncertain situations. To control the longitudinal motion of the SV, the policy is a sequence of accelerations (see sub-section 4.3.2 for more details on the action-space). Three solutions are considered to interface the POMDP with other navigation sub-systems, these are shown in Figure 4.3.

4 Probabilistic Decision-Making, Applied to Crossroad Intersections

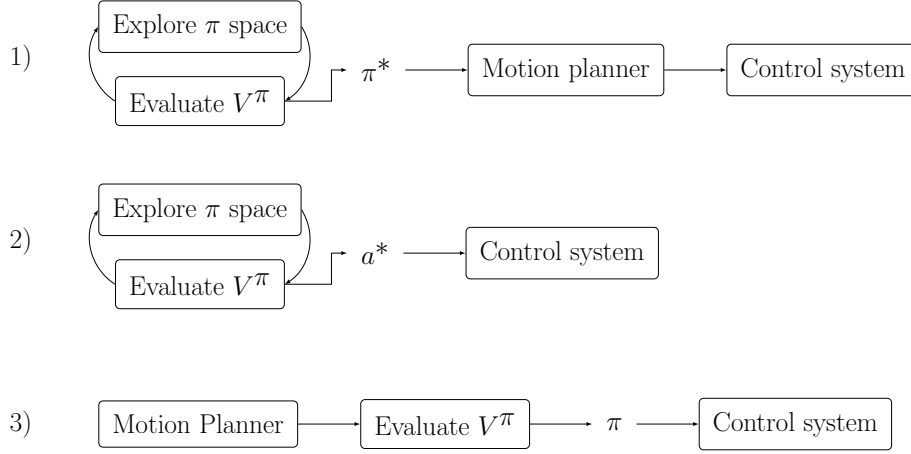


Figure 4.3 Different methods to integrate POMDP within a navigation system

1) The POMDP solver explores and evaluates policies to approximate the optimal one. The result is sent to the motion planner to construct a dynamically feasible trajectory for the control system. With few changes in the situation or if the situation progresses as predicted by the model, the policy holds for some time. However, the plan needs to be updated as the situation progresses and new observations are acquired. An update is triggered by unexpected observations, or by the lack of time to evaluate the policy due to dimensionality issues. In our scenario, this update might be frequent because of the large number of states and the uncertainty associated with the other driver behaviour.

2) The POMDP solver explores and evaluates policies, but this time only the best action for the current situation is used. This action is sent directly to the control system to be applied. The chosen action is the one with the best average return of all the policies resulting from this action. This method considers that the policy evaluation is difficult in the long term. Thus, using one action is more valuable than the whole policy. It is suitable for scenarios that do not require complex motion manoeuvres. In our scenario, the POMDP only controls the longitudinal motion of the vehicle and does not need to reason over a long-time horizon to evaluate a single action.

3) The motion planner generates policy sets that are evaluated with the POMDP. These can be guaranteed to be feasible and to avoid short term collisions. However, the optimal policy is not necessary in the set given by the motion planner. The resulting decision might be a sub-optimal solution of the decision-process, which could have found a better solution if unconstrained by the motion planner.

The second option is chosen for this work as the decision-making process only controls the

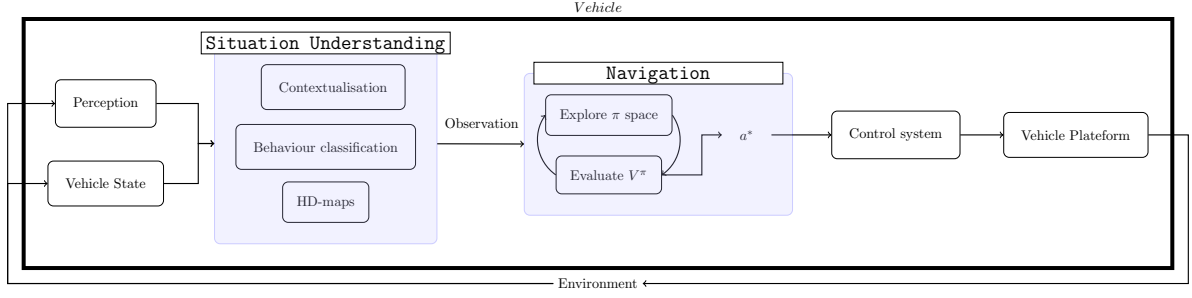


Figure 4.4 Incorporation of the proposed systems into a Vehicle centric architecture. Coloured blocks correspond to the proposed situation understanding mechanism and decision making as part of the decision-making and navigation functional block

longitudinal motion of the vehicle and its actions are acceleration commands. Furthermore, the reward function uses short term reward to deal with problem related to behaviours.

The vehicle centric architecture discussed in Section 2.2.1.1 is adapted to integrate the decision-making process. The situation understanding system includes the classification framework described in Chapter 3 and contextualisation that transforms perception measurements into observations used by the decision-making process. The navigation system includes the decision-making framework that explores POMDP policies and evaluates them using the model for road intersection crossing. The action, that is currently the best evaluated, is periodically sent to the control system to be executed. The overall architecture of the system is shown in Figure 4.4.

4.3 Decision-Making Framework

The POMDP framework needs a model of the environment for the solver to find the optimal policy. The model structure is represented by a Dynamic Bayesian Network, shown Figure 4.5. It shows variables describing the state both vehicles and interactions between their state variables. In this system, variables are discrete. This model is then used by a Partially Observable Monte-Carlo Planning (POMCP), that is an online POMDP solver to find the best action. A POMDP may include continuous variables but these increase the problem complexity. A continuous POMDP solvers adds a discretisation steps made in real-time [30, 92].

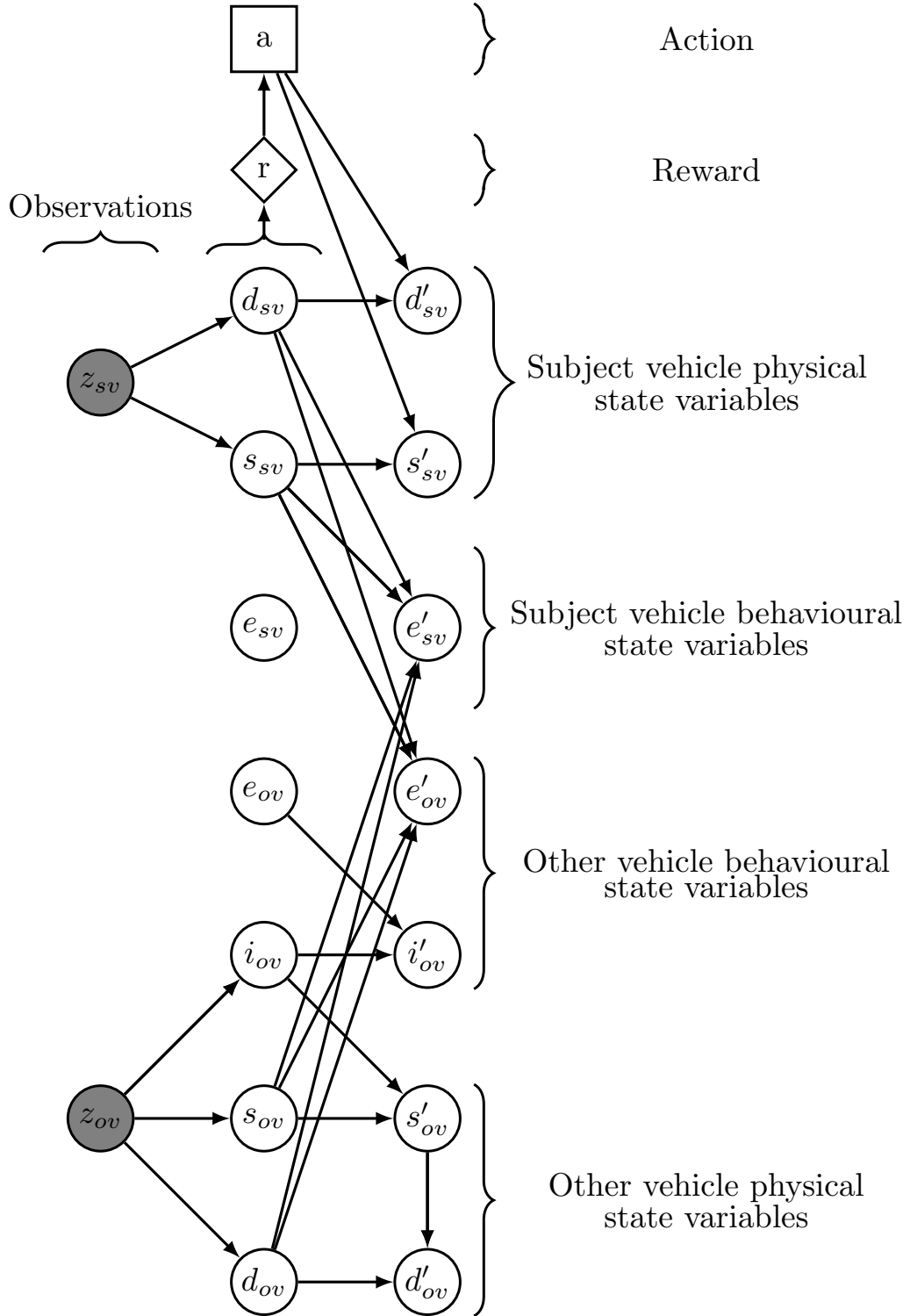


Figure 4.5 Structure of the POMDP, as a Dynamic Bayesian Network

4.3.1 State-Space

The state-space of a vehicle is divided in two: the physical state and the behavioural state. Vehicle physical state is represented by speed and distance variables. The vehicle speed s is discretized with a step of $1m/s$, it ranges between 0 to $14m/s$ ($50km/h$). The vehicle position is transformed into the relative distance of the vehicle to the intersection entrance, thus d represents the distance to the intersection entrance. It ranges between $0m$ to $50m$ with a step of $1m$. The functional discretisation is not used for this model as steps between zones are too long. It was not possible to design actions and transition to work with the functional zones.

The behavioural state variables include: the expected longitudinal manoeuvre e and the intended longitudinal manoeuvre i . These are defined over the same discrete space of manoeuvre, as defined in Equation (4.1), applied to road intersection crossing (c.f. Section 2.3.1).

$$\mathcal{M} = \{Stop, Yield, Cross\} \quad (4.1)$$

The intended manoeuvre represents what the driver is doing, whereas the expected manoeuvre represents what the situation requires [122]. This difference is important in situations that include a human driver that is not looking after cooperating with the SV. The cause may be intentional (e.g. unruly drivers) or unintentional (e.g. distracted drivers).

The behaviour of the SV is represented only by the expectation variable. A vehicle driven by an automated system has no reason to behave differently of what is expected by the situation.

Formally, the state of the environment is a vector containing these seven variables, Equation (4.2). These are shown in Figure 4.6. These variables indicate the large space that the environment-state vector can take.

$$x = [d_{sv}, s_{sv}, e_{sv}, d_{ov}, s_{ov}, e_{ov}, i_{ov}] \quad (4.2)$$

4.3.2 Action-Space

It describes the manner how the SV interacts with its immediate environment. In our application, the vehicle changes its speed. This is done by modifying the acceleration of the vehicle. Thus,

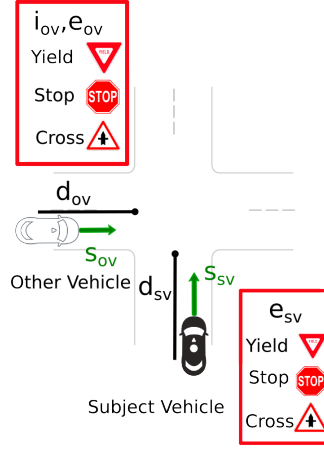


Figure 4.6 Variables defined for the model applied to road intersection crossing.

the action-space can be regarded as an ensemble of possible accelerations. These allow the SV to stop at the intersection starting from its current speed, and to accelerate, if it infers a favourable situation. The generation of strong decelerations to avoid collisions (i.e. safety brake) are not included in the model. The decision-making system aims to reduce the risk of collision by taking pre-emptive action. The control system obeys to the commands and ensures that the vehicle manoeuvres within the safe dynamic capabilities of the vehicle. A continuous action-space is possible, however it adds unnecessary complexity by discretising the action-space online. Thus most of the work use discrete action-spaces with 3 to 5 actions [91, 30]. However, the resulting command is not smooth, thus the control system needs to compensate this, model predictive control is one of the solution adapted to this problem [123].

In view of these observations, the action-space is defined by:

$$\mathcal{A} = \{-2, -1.5, -1, -0.5, 0, +1\} m/s^2$$

To approach a road intersection, a vehicle most likely has to decelerate, thus the action-space contains a wider collection of decelerations. A $0.5 m/s^2$ step should be sufficient to reduce uncomfortable acceleration changes. A single action makes the vehicle to accelerate. This is for situations where the vehicle restart after a stop, to slowly accelerate after the OV has crossed or to increase the speed a little when both vehicles are interacting correctly. These actions, sent to the vehicle control system, make the vehicle change its speeds. They are applied during $\Delta t = 0.5s$.

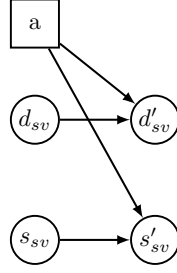


Figure 4.7 Transition of the physical state of the SV.

The size of the action-space is adapted to the capabilities of our solver that first estimates the value of each action (c.f. Section 4.4) and could under perform with more actions.

4.3.3 Transition Model

The transition model describes how an action of the SV changes the environment state. The transition function is given by:

$$P(x'|x, a) = P(s'_{sv}|a, s_{sv})P(d'_{sv}|a, d_{sv}, s_{sv})P(e'_{sv}|s_{sv}, s_{ov}, d_{sv}, d_{ov}) \quad (4.3)$$

$$P(e'_{ov}|s_{sv}, s_{ov}, d_{sv}, d_{ov})P(i'_{ov}|e_{ov}, i_{ov})P(s'_{ov}|s_{ov}, i_{ov})P(d'_{ov}|d_{ov}, s'_{ov})$$

The first part of the transition model represents the impact of action of the SV on its own physical state, as shown in Figure 4.7.

The coarse discretisation and the relatively long-time step (0.5 seconds) makes complex control and physical models of the vehicle unnecessary. Anyhow, an uncertainty on its capability to execute an action is considered. It is produced by instability in the control or external factor (e.g. wet road). These are modelled as Gaussian distributions over the change of speed and distance as defined in Equations (4.4) and (4.5):

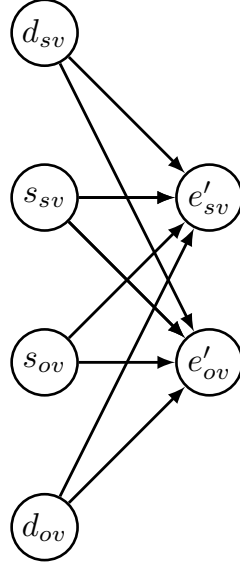


Figure 4.8 Expectation transitions for the SV and the OV. These depend on their physical states.

$$P(s'_{sv}|a, s_{sv}) = \mathcal{N}(\mu_s, \sigma_a) \quad (4.4)$$

$$\text{With } \mu_s = s_{sv} + a\Delta t$$

$$\sigma_a = 1$$

$$P(d'_{sv}|a, d_{sv}, s_{sv}) = \mathcal{N}(\mu_d, \sigma_a) \quad (4.5)$$

$$\text{With } \mu_d = d_{sv} - (s_{sv}\Delta t + a0.5\Delta t^2)$$

To infer the expectation transition, the influence of the priority rule in the vehicle expected manoeuvre is considered. This state variable could be fixed to be the same as the priority rule. However, when approaching an intersection, it is sometimes difficult to infer what a driver should do. After applying an action, there is a uncertainty about what might happen next, it is therefore important to model also the expectation transition as a probabilistic transition. Only the situation facing a stop sign is unambiguous as there is certainty on the expectation that the vehicle in nominal conditions will execute a stop manoeuvre.

The transition is the same for both vehicles and depends on their physical states, as shown in Figure 4.8.

We used a gap acceptance model from Pollatscheck et al. to estimate the probability that a vehicle stops for a given a time gap [124], as expressed in Equation (4.6). When the gap is small, the probability to stop is high, whilst for a large gap, the probability will be small. The gap acceptance model has been adapted to include the yield manoeuvre. Parameters δ and σ are respectively set to 6.1 and -4 as in Lefevre et al [53].

$$p_{stop} = 1 - \frac{1.05}{1 + \left(\frac{gap}{\delta}\right)^\sigma} \quad (4.6)$$

$$gap = \frac{d_{v1}}{s_{v1}} - \frac{d_{v2}}{s_{v2}}$$

$$p_{pass} = 1 - p_{stop} \quad (4.7)$$

Dependence on the priority rule implies three cases:

1. Facing a stop sign, the only legal and expected manoeuvre will be to stop, thus the transition probability will be given by Equation (4.8).

$$P_{stop}(e'_{v1}|s_{v1}, s_{v1}, d_{v2}, d_{v2}) \begin{cases} P(e'_{v1} = stop) = 1 \\ P(e'_{v1} = yield) = 0 \\ P(e'_{v1} = cross) = 0 \end{cases} \quad (4.8)$$

2. Facing a yield sign, a driver is expected either to stop if the gap is small or to slow down and yield. Thus, Equation (4.6) gives the probability for the vehicle to come to a stop with its complement representing the probability to yield. In this case the transition probability is as follows:

$$P_{yield}(e'_{v1}|s_{v1}, s_{v1}, d_{v2}, d_{v2}) \begin{cases} P(e'_{v1} = stop) = p_{stop} \\ P(e'_{v1} = yield) = p_{pass} \\ P(e'_{v1} = cross) = 0 \end{cases} \quad (4.9)$$

3. If the vehicle is given the priority, all the three manoeuvres are expected. The crossing manoeuvre is given a slightly higher probability to represent the fact that this vehicle has the priority. The driver is expected to yield or to stop, if the gap is too small. This transition probability is as follow:

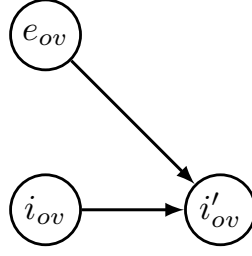


Figure 4.9 Intention transition for the OV. It depends on its behavioural state

$$P_{cross}(e'_{v1}|s_{v1}, s_{v1}, d_{v2}, d_{v2}) \begin{cases} P(e'_{v1} = stop) = p_{stop} \\ P(e'_{v1} = yield) = \frac{p_{pass}}{3} \\ P(e'_{v1} = cross) = 2 * \frac{p_{pass}}{3} \end{cases} \quad (4.10)$$

This model is applied to both e_{sv} and e_{ov} .

The intention transition probability depends on the previous intention and the expectation of the OV, as shown in Figure 4.9. This transition uses the notion of intention continuation [53]. That is, if the current driver's intention is the same as its expectation, the driver's intention is likely to remain the same. A driver, that has the correct intention, is likely to follow it in the future. Otherwise, if i_{ov} and e_{ov} are different, the next intention is unknown, therefore it will be uniformly distributed over the manoeuvre set. This implies that the driver has an erratic behaviour. This can be represented by the Equation (4.11)

$$P(i'_{ov}|e_{ov}, i_{ov}) \begin{cases} P(i'_{ov} = i_{ov}) = 0.9 & i_{ov} = e_{ov} \\ P(i'_{ov}) = \mathcal{U}(\mathcal{I}) & otherwise \end{cases} \quad (4.11)$$

The final part of the transition model concerns the physical states of the OV that depends on its previous state and behaviour, as shown in Figure 4.10.

As for the SV with Equations (4.4), these transitions use simple kinematic models to find the distribution of the next s'_{ov} and d'_{ov} . The action of the OV vehicle is unknown but mostly depends on its intention. For each manoeuvre, a different acceleration distribution is used to predict the future speed, Equation (4.14). The next distance is found with these speeds and previous distance, Equation (4.13).

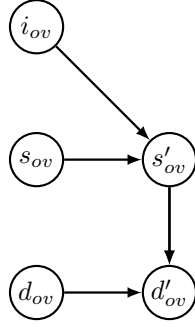


Figure 4.10 Physical transitions for the OV. These depend on the OV intention and previous physical state

$$P(s'_{ov}|s_{ov}, i_{ov}) = s_{ov} + \mathcal{N}(\mu_{acc}, \sigma_{acc})\Delta t \quad (4.12)$$

$$P(d'_{ov}|s'_{ov}, d_{ov}) = d_{ov} + s'_{ov} * \Delta t + 0.5 * \mathcal{N}(\mu_{acc}, \sigma_{acc})\Delta t^2 \quad (4.13)$$

$$With \quad \mu_{acc} = \begin{cases} 0 & i_{ov} = cross \\ -0.5 & i_{ov} = yield \\ -1 & i_{ov} = stop \end{cases} \quad (4.14)$$

$$\sigma_{acc} = 1 \quad (4.15)$$

These models are used to predict the probability of future environment states. There are several variables and uncertainty about the effect actions of the SV on the environment, thus it becomes difficult to keep meaningful distribution over each variable after few steps.

In this application, the transitions are based on expert knowledge and simple kinematic models. Recently, machine learning has been used to model the transition function [125]. It provides better models for the transitions, however, it lacks explainability introducing problems into already complex process. These models are learned using observations and not on the true state of the environment. Models for the OV motion (in Equation (4.12)) and the intention model (in Equation (4.11)) can be replaced by such methods to better cover human behaviours. The approach of Teawon et al. for example could be used to combine machine learning with gap acceptance models to maintain the understandability [126].

4.3.4 Observation-Space

The observation-space describes information obtained by the agent after an iteration. The belief over the true state of the environment is estimated from these observations. They are obtained from the exteroceptive and proprioceptive perception systems and situation understanding systems. The observation vector contains information about the SV and the OV:

$$z = [z_{sv}, z_{ov}] \quad (4.16)$$

with z_{sv} the SV vector of observation and z_{ov} the OV observation vector.

For the SV, its physical variables are observable. The observation of the distance to the intersection $\hat{d}_{sv}$ is found with a map matching between the current position of the SV and a lane level map. The position is projected onto the closest center lane that ends at an intersection entrance. The other measurement is the SV speed $\hat{s}_{sv}$. It is measured either with specific devices (IMU+GPS) or standard sensors embedded on any vehicles and shared over the CAN-bus. The vector representing the SV observation is written:

$$z_{sv} = [\hat{s}_{sv}, \hat{d}_{sv}] \quad (4.17)$$

These physical observations are also obtained for the OV, however with more uncertainty due to the perception system limits. The perception system returns the position of the OV that is used in a similar technique as the position of the SV to find the observed distance to its intersection entrance $\hat{d}_{ov}$. Its speed $\hat{s}_{ov}$ can be measured from radar sensor, or by tracking the vehicle across multiple consecutive images or lidar frames. With the framework of Chapter 3, the OV manoeuvre is observed. It gives an observation on the intended longitudinal manoeuvre $\hat{i}_{ov}$ that is used to estimate the belief on the true intention. The observation vector of the OV takes the form:

$$z_{ov} = [\hat{s}_{ov}, \hat{d}_{ov}, \hat{i}_{ov}] \quad (4.18)$$

4.3.5 Observation Model

This model characterizes the probabilistic relation between observations and the environment state. As each component of the observation-space has its counterpart in the state-space, it can

be described by:

$$P(z|x) = P(\hat{d}_{sv}|d_{sv})P(\hat{s}_{sv}|s_{sv})P(\hat{d}_{ov}|d_{ov})P(\hat{s}_{ov}|s_{ov})P(\hat{i}_{ov}|i_{ov}) \quad (4.19)$$

This part of the model is the most complex to build for this type of application. The uncertainty associated with the sensing chain is difficult to characterize. The perception system, that gives $\hat{d}_{ov}, \hat{s}_{ov}$, has different type of uncertainty for each sensor used (e.g. camera glare, lost lidar points). Then as data gets processed, environment sensing uncertainty changes. When multi-sensor data is fused, uncertainty is reduced through the fusion process, nevertheless, the distribution representing the environment sensing uncertainty remains complex to model. The same applies to observations of the SV states, they are affected by sensors uncertainty and local phenomena that affect GPS localisation. It affects how correctly the state of the SV is inferred. Communication between vehicles provides more accurate observations, however, it introduces other types of uncertainty (e.g. transmission delays). In a simulation, the physical state can be known without uncertainty by accessing the state of each agent.

To balance between complex observation model required for real-life implementation and low uncertainty achievable in simulated environment, the observation model assumes that each observation follows a Gaussian distribution. Observation of the SV physical state are modelled with a Gaussian noise including a small variance $\mathcal{N}(0, 0.5)$. Similarly, a Gaussian noise is considered for the OV observations, however, we have increased the variance $\mathcal{N}(0, 1)$. The increase of the variance represent the fact that the uncertainty from the perception system is higher than the vehicle state estimation system. These Gaussian noises are a good trade-off neither to overestimate nor to underestimate uncertainties.

The observation model of the intention uses performances of the classifier introduced in section 3.4 to know the probability of the classifier returning the correct estimated manoeuvre. The accuracy is estimated at 0.8. Consequently the true intention of the OV has a probability to be correct of 0.8 or to be uniformly wrong otherwise. This is described in Equation (4.20).

$$P(\hat{i}_{ov}|i_{ov}) \begin{cases} P(\hat{i}_{ov} = i_{ov}) = 0.8 \\ P(\hat{i}_{ov} = \overline{i_{ov}}) = 0.2 \end{cases} \quad (4.20)$$

4.3.6 Reward Function

The reward function applied to road intersection crossing use terminal reward models [30, 92]. They generate a large penalty for collisions and a high reward for crossing the intersection. Systems using these simple reward function display high performances in terms of crossing time and collision avoidance. However, they do not consider the behavioural interaction with other drivers while approaching the intersection. This interaction allows to solve the situation in a manner that both vehicles are satisfying their intention. Nevertheless, these must be used carefully as some human driver might not interact with the SV (see Section 1.2). The Reward function proposed in this thesis combines short term and behaviour related rewards. This is defined in Equation (4.21).

$$\begin{aligned}
 R(x, a) = & w_c(d_{sv})r_c(a) + w_r(d_{sv})r_r(s_{sv}, d_{sv}, s_{ov}, d_{ov}) \\
 & + w_e(d_{sv})r_e(e_{ov}, i_{ov}) + w_s(d_{sv})r_s(s_{sv}, d_{sv}, a) \\
 & + w_i(d_{sv})r_i(e_{sv}, i_{ov})
 \end{aligned} \tag{4.21}$$

Each reward component is weighted by a linear function $w(d_{sv}) = k_1 d_{sv} + k_2$. It depends on the SV distance to the intersection entrance to balance differently each reward component as the SV approaches. For example, comfort and interaction are important far away from the intersection whilst considering collisions risk at the intersection is not of concern. These concerns change when the SV is close to the intersection entrance, risk must be prevented even if passenger comfort is compromised. With this approach, the range of each modality must be the same. Otherwise, the weight has few or no effect on the global reward. For this purpose, we introduce a variable R_{max} such that $-R_{max} < r(s, a) < R_{max}$. These weights are found by iterating over possible configurations to find the appropriate one. It is expected that many configurations are viable and will allow for personalisation of the behaviour.

The first part of the reward function uses physical variables and the SV actions to compute three types of reward.

The comfort reward $r_c(a)$ penalises actions of the agent with strong deceleration. These are penalised to avoid passenger discomfort in the vehicle [127, 128]. Therefore, in our setup, the agent is only penalised, if it chooses the strongest deceleration. Otherwise, the system gets no reward.

$$r_c(a) \begin{cases} -R_{max} & a = -2m/s^2 \\ 0 & \text{Otherwise} \end{cases} \quad (4.22)$$

The speed reward $r_s(s_{sv}, d_{sv}, a)$ will reward the agent if its actions bring its speed closer to the reference speed, where the reference speeds S_{ref} are obtained by observing the speed of a single simulated agent approaching the intersections. An example of approaching a yield sign is shown in Figure 4.11a. It represents what the agent should do if there is no other vehicle (this reward does not use variables from the OV). An agent whose speed is within $\pm 2m/s$ is given the maximum reward due to its similarity with the reference speeds. If the agent chooses actions that reduce the difference with the reference speed, it will get half the maximum reward. For other situations, there are no penalties as drivers can choose their speeds at their convenience within the highway code framework. An example of this reward function is given in Figure 4.11b. this is defined by:

$$r_s(s_{sv}, d_{sv}, a) \begin{cases} R_{max} & |S_{ref}(d_{sv}) - s_{sv}| < 2 \\ R_{max}/2 & S_{ref}(d_{sv}) - s_{sv} > 2 \wedge a > 0 \\ R_{max}/2 & S_{ref}(d_{sv}) - s_{sv} < -2 \wedge a < 0 \\ 0 & \text{otherwise} \end{cases} \quad (4.23)$$

The risk reward $r_r(s_{sv}, d_{sv}, s_{ov}, d_{ov})$ penalises situations that might have two vehicles crossing the intersection at the same time. The risk is measured as the difference between the time it takes two vehicles to reach the intersection entrance assuming they do not decelerate or accelerate. This situation is considered as dangerous as collisions occur mostly when two vehicles enter the intersection at the same time. This measurement is similar to time to collision. To avoid risk and maintain passenger's trust, a gap of 4s is preferred [129] at a stop intersection, an extra 1 second is added for safety concerns. This value is used as a threshold over which the agent gets the maximum reward, otherwise, it gets the value of the time gap. The formulation of the risk reward is given in Equation (4.24).

$$r_r(s_{sv}, d_{sv}, s_{ov}, d_{ov}) \begin{cases} R_{max} & \left| \frac{d_{sv}}{s_{sv}} - \frac{d_{ov}}{s_{ov}} - 5s \right| > 0 \\ \left| \frac{d_{sv}}{s_{sv}} - \frac{d_{ov}}{s_{ov}} \right| & \text{otherwise} \end{cases} \quad (4.24)$$

The second part of the reward model is made of two components that use the behaviour variables

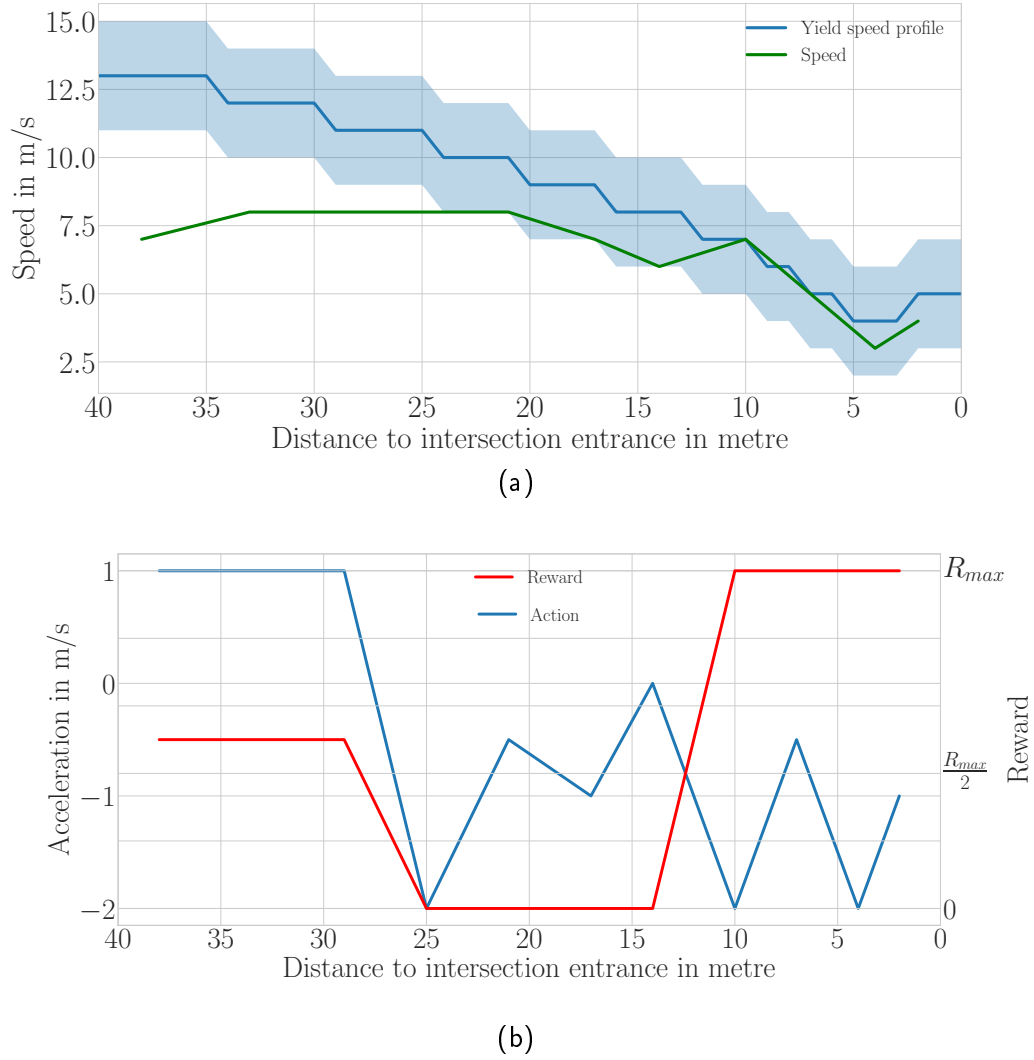


Figure 4.11 Example of reward return by the reference speed related reward function. The shaded area represents the range within which the agent can get the maximum reward. An example is included of the vehicle speed profile as it is brought into the maximum reward area. (b) Rewards obtained by the agent (in red) given the vehicle action (in blue) respected as vehicle accelerations.

Table 4.1 Rewards related to the behaviour of drivers with respect to traffic laws

$i_{ov} \backslash e_{sv}$	Stop	Yield	Cross
Stop	0	0	$R_{max}/2$
Yield	$R_{max}/2$	0	$R_{max}/2$
Cross	R_{max}	$R_{max}/2$	$-R_{max}$

The expectation reward $r_e(e_{ov}, i_{ov})$ follows the work by Lefevre et al., this is based on the notion of risk [53]. They estimate risk as the probability that the intended and the expected manoeuvre are different. When this probability is over 0.3, the situation is identified as dangerous. In our work this notion is embedded into the reward function (Equation (4.25)) that penalises states with a difference between e_{ov} and i_{ov} . Otherwise, the agent gets half of the maximum reward.

$$r_e(e_{ov}, i_{ov}) \begin{cases} R_{max}/2 & e_{ov} = i_{ov} \\ -R_{max} & otherwise \end{cases} \quad (4.25)$$

This reward incorporates the interaction between the SV and the OV. The e_{ov} depends on the physical state of the SV and the OV. Thus, the SV can choose actions that change its physical state, and thus the expected behaviour, to bring e_{ov} to be the same as i_{ov} . This is a one sided-interaction with a dangerous OV behaviours that is corrected by SV actions to reduce this risk by making the global situation less dangerous.

The intention reward $r_i(e_{sv}, i_{ov})$ recompenses states within the OV and the SV behaviour variables that comply with the highway code. The reward obtained for each variable combination is shown in Table 4.1.

The intention of the other vehicle is used because it highly impacts its motion (see Equation (4.12)). Situations that are not dangerous with combined the OV and the SV behaviour (stop/stop, stop/yield, yield/yield) but not optimal in terms of interaction are neither penalised nor rewarded. Half of the maximum reward is obtained for combination that reflects correct interaction though it becomes unsafe if the motion is carelessly executed (i.e. stop/yield, cross/yield). Maximum reward is attributed, if the OV intends to stop while the SV is expected to cross. It forces the agent to cross even if the risk is high. The situation with both vehicles trying to cross is highly penalised as there is no consensus, thus collisions might happen.

With this reward, the agent receives incentives to take actions so as to change its expected behaviour to interact with the OV.

The agent needs to use every component of the reward function to carefully approach the intersection, while choosing between performances (r_s, r_c) , risks, (r_e, r_r) and interactions (r_i) . This combined reward model is used by the solver to evaluate policies

4.4 Partially Observable Monte-Carlo Planning: An Online POMDP Solver

The model in Section 4.3 describes how the working environment functions. The subject vehicle shall find and evaluate policies for the confronted situation. These processes are difficult due to the large number of states, uncertainties associated with the observed variables and the manner the situation might evolve. The POMDP is solved using an online solver described below.

4.4.1 POMDP Solver

A POMDP solver evaluates policies which are sequences of actions $\pi = \{a_t, a_{t+1}, \dots, a_{t+n}\}$. This evaluation uses the model POMDP model, described in section 4.3, to find the policy value over a time horizon considering its consequences. There is always an optimal policy π^* that maximizes the value function. The solver finds this optimal policy or tries to approximate it.

The value of a policy V^π is the sum of the discounted expected rewards following the policy described in Equation (4.26).

$$V^\pi(b_0) = \sum_{t=0}^{\infty} \gamma^t R(b_t, a_t) = \sum_{t=0}^{\infty} \gamma^t E(R(x_t, a_t) | b_0, \pi) \quad (4.26)$$

where γ is the discount value, working as a stop condition for the evaluation. This value reduces the influence of future reward, some examples of its discounting effect are shown in Figure 4.12.

With $\gamma = 0$, only the next reward is considered, whilst for and with $\gamma = 1$ every reward has the same value and the evaluation stop when a final state is reached. Usually, this value is set between 0 and 1, to gradually decrease the reward importance. A long search horizon is more difficult

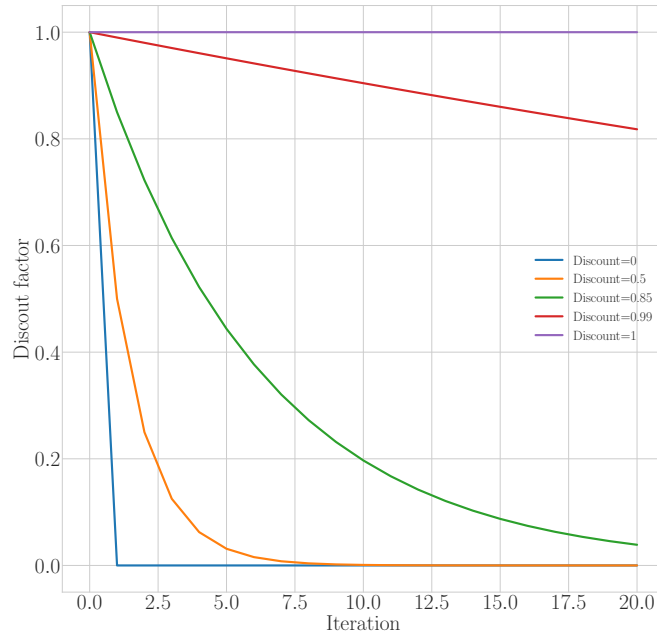


Figure 4.12 Discounting factor for different values of γ

as it requires more iterations. A practical use of the discount factor is to stop the evaluation, if discounting is too strong that new rewards are negligible. Thus, when the discounting factor is below ϵ (Equation (4.27)), the process is stopped.

$$\gamma^t < \epsilon \quad (4.27)$$

The time horizon H can be derived from Equations (4.28).

$$\begin{aligned} \ln(\gamma^{H/\Delta t}) &= \ln(\epsilon) \\ \frac{H}{\Delta t} \ln(\gamma) &= \ln(\epsilon) \\ H &= \Delta t \frac{\ln(\epsilon)}{\ln(\gamma)} \\ H &= \Delta t (\ln(\epsilon) - \ln(\gamma)) \end{aligned} \quad (4.28)$$

with ϵ a threshold below which the search is stopped.

With the reward function presented in section 4.3.6, there is no need for a long search in time as agent gets an immediate reward for its actions. Different values of this parameter will be tested to optimize performances in Section 5.4.1.

There exist two groups of solvers (see section 2.4.2). With the high number of states in our model and time constraints, Offline solver was not a practicable solution [80]. It requires to restart the entire solving process each time a parameter changes, furthermore this process is intractable in time. However, it does not require computation when the decision happen, and optimality can be guaranteed. An Online solver is more practical because, it only tries to solve the part of the model that corresponds to the current situation. This allows to solve large state problems and is more realistic for the proposed vehicle implementation. However, there is no guarantee that the policy estimation converges for the complete time horizon in the given time to choose an action, but the estimation on the short term horizon the evaluation shall converge. This is acceptable in our application as the evolution of the situation is uncertain for a time horizon of 5-10 seconds. Consequently, the short term motion is well evaluated while long term consequences are considered but with a lesser impact on the next action.

4.4.2 POMCP Elements

Out of the multiple algorithms existing [85, 130, 131, 132], POMCP developed by Silver et al. has been chosen for its simplicity and scalability [88]. The core idea of their approach is to use a Monte-Carlo tree search (MCTS) to track the belief combined with an Upper confidence bound (UCB-1) to direct the policy search. These are explained below.

4.4.2.1 Monte-Carlo Search Tree

Monte-Carlo tree search is heuristic search algorithm that allows to evaluate POMDP policies. It uses a generative function to update a particle state to estimate the belief after an action has taken place. This is built from the transition, reward and observation function of the POMDP model described in Section 4.3 with:

$$G(x, a) \sim (x', z', r') \quad (4.29)$$

The tree is composed of nodes $T(h) = \langle N(h), V(h), B(h) \rangle$ with $h = \{a_0, Z_o, a_1, Z_1, \dots, a_t, Z_t\}$ the history of actions and observations previously explored. $N(h)$ corresponds to the number of

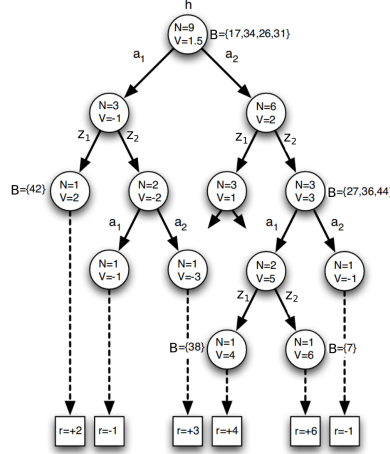


Figure 4.13 An example of a MCTS tree, with two actions, two observations and 50 states, after [88]

times this history has been explored, $V(h)$ is the value of h that is the mean return of cumulated reward from simulations and $B(h)$ the belief represented by particles that reached this node. During the search time, different actions are applied on state samples from the current belief. The generative model is used to update particles. Each time a particle reaches node, it increases by one $N(h)$, updates the value function with the cumulative reward and adds its state the belief function of the node (more details in 4.4.3). This procedure is repeated as many times as possible. The structure of a simplified the tree is shown in Figure 4.13.

4.4.2.2 Upper Confidence Bound

Random action sampling from the action-space during the search, provides a slow convergence. A guided search to explore the policy space provides better results. The application of the Upper confident bound method fasten the convergence by balancing exploration and exploitation [133]. Instead of choosing a random action to update particles, this method estimates which action could bring more information on the optimal policy.

For each action of the current node $T(h)$, an augmented value criterion $V^\oplus$ (in Equation (4.30)) is computed. The action that maximizes this criterion is applied on the particle.

$$V^\oplus(ha) = V(ha) + C \sqrt{\frac{\log(N(h))}{N(ha)}} \quad (4.30)$$

ha is the current history obtained by applying action a , C is a parameter that controls the trade-off between exploitation and exploration. If $C \rightarrow 0$, exploitation is favoured as only the value function impacts the UCB1, thus only the action that has the most value is tried. Increasing C , in an order of magnitude sufficient to outweigh the value function, makes the system exploring each action uniformly. Different values for this parameter will be tested to find the optimal configuration in Section 5.4.1.

Actions with the highest value are chosen more often, but as the estimation of the value function converges, new actions are tried to refine the estimation or to find avoid a local minimum of the value function.

4.4.3 POMCP Algorithm

The POMCP joins the UCB1 and MCTS to solve POMDPs in an online manner, that is a certain amount of time is given to estimate the value function, then the policy that maximise this estimation is applied by the agent. With this step by step approach, only the starting action of the policy is used as control inputs.

The POMCP algorithm is divided into three procedures: Search, Simulate and Rollout. The following explains these procedures.

The *search* procedure starts the algorithm. It samples particles x from the current belief, or from an initial distribution $\mathcal{I}$ if the history is empty. This particle is used by the stimulate procedure to update the Value function. Once Δt elapses in the real life, the sampling stops and the action that maximises the value function is returned to be executed. This repeated until the agent has finished its mission. The Algorithm is in 4.14.

The *simulate* procedure updates the state of the particle using the generative model and builds the MCTS tree. The name simulate is used because each particle represents one the possible future tried using the model that represent a simpler environment, before being applied in the real world. This is a recurrent procedure that is stopped when $\gamma^{depth} < \epsilon$.

The simulate procedure searches if the tree contains a node associated with the history of the particle. If not, it creates a new node for each action and applies the rollout procedure. When the history is already in the tree, the UCB1 is applied to choose an action. It allows to build upon previous evaluation to fasten the evaluation. This action is applied on the particle and the procedure restarts with the updated particle. Once the recurrence has stopped, nodes of

Algorithm 1 POMCP search procedure

```

1: procedure SEARCH( $h$ )
2:   repeat
3:     if  $h = \text{empty}$  then
4:        $s \sim \mathcal{I}$ 
5:     else
6:        $x \sim B(h)$ 
7:     end if
8:      $\text{SIMULATE}(x, h, 0)$ 
9:
10:  until  $\text{TIMEOUT}()$ 
11:  return  $\underset{b}{\operatorname{argmin}} V(hb)$ 
12: end procedure

```

Figure 4.14 Search Procedure

the tree reached by the simulate procedure are updated. The node value is updated with the accumulated discounted reward of future episode. The full procedure is in Algorithm 4.15.

The *rollout* procedure aims to quickly explore states and actions when a node is created. It applies a default policy that shall resemble a high value policy. The two common procedures are the random action selection or action continuation. The random action rollout policy randomly selects at each episode an action, whereas action continuation repeats the last action over and over. In our scenario, the optimal policy is more likely to contains sequence with the same action applied, to slowdown for example. Thus, the action continuation rollout policy is used. The discounted cumulated reward is used to initiate the value function of the corresponding history.

The search is stopped after Δt has passed. Then, the action that maximise $V(ha)$ is chosen and sent to the controller for the duration Δt . Nodes that do not contain the new history are pruned from the tree. The process is repeated until the vehicle crosses the intersection.

The more the search procedure is repeated the more accurate the estimation of the value functions gets. The complexity of the model, the time horizon and the policy exploration are parameters that affect the time it takes to complete the simulate procedure. There is also a trade to do between the evaluation performances and these parameters (see Section 5.4.1). The simplest implementation of the solver is used for in our system. A more advanced implementation can include imitation learning to initiate $V(ha)$ with knowledge obtained by looking at human drivers.

Algorithm 2 POMCP Simulate procedure

```

1: procedure SIMULATE( $x, h, depth$ )
2:   if  $\gamma^{depth} < \epsilon$  then
3:     return 0
4:   end if
5:   if  $h \notin T$  then
6:     for  $a \in \mathcal{A}$  do
7:        $T(ha) \leftarrow (N_{init}, V_{init}, \emptyset)$ 
8:     end for
9:     return ROLLOUT( $x, h, depth$ )
10:  end if
11:   $a \leftarrow \underset{b}{\operatorname{argmin}} V(hb) + C \sqrt{\frac{\log(N(h))}{N(hb)}}$ 
12:   $(x', z, r) \sim G(x, a)$ 
13:   $R \leftarrow r + \gamma \text{SIMULATE}(x', haz, depth + 1)$ 
14:   $B(h) \leftarrow B(h) \cup \{s\}$ 
15:   $N(h) \leftarrow N(h) + 1$ 
16:   $N(ha) \leftarrow N(ha) + 1$ 
17:   $V(ha) \leftarrow V(ha) + \frac{R - V(ha)}{N(ha)}$ 
18:  return  $R$ 
19: end procedure

```

Figure 4.15 Simulation procedure

Algorithm 3 POMCP Rollout procedure

```

1: procedure ROLLOUT( $x, h, depth$ )
2:   if  $\gamma^{depth} < \epsilon$  then
3:     return 0
4:   end if
5:    $a \sim \pi_{rollout}(h, \cdot)$ 
6:    $(x', z, r) \sim G(x, a)$ 
7:   return  $r + \gamma \text{ROLLOUT}(x', haz, depth + 1)$ 
8: end procedure

```

Figure 4.16 Rollout procedure

4.5 Conclusion

A framework for an autonomous vehicle to decide how to approach and to cross an intersection is presented in this Chapter. The approach is based on a POMDP to reason considering the uncertainties associated with the situation and driver behaviour. In order to solve the proposed POMDP, an online solver, POMCP, is used to approximate the policy value during execution. Chapter 5 presents the applications of the framework as well as the means developed for its evaluation.

The POMDP controls the longitudinal motion of the subject vehicle by choosing actions that are evaluated considering the behaviour of the other vehicle. The SV tries to interact with the OV by taking actions that allow interactions. It works by including behavioural variables into the POMDP reward model. The multi-objective reward function allows for the customization of the agent behaviour to favour certain rewards with respect to the intersection entrance distance.

In order to use this complex model an online POMDP solver, called POMCP, is applied. This estimates the value function of a policy range by balancing exploration and exploitation. To integrate the framework into the automated vehicle architecture, only the next best action is used and sent to vehicle control. It avoids planning vehicle motion for a long-time horizon whereas changes are likely to occur often. However, this results on the vehicle decision to be non-deterministic as much randomisation is involved in the POMCP. Consequently, given the same initial condition, the policy of SV might be different. This is to be expected for driving application as observation and interactions are uncertain.

The proposed models rely mainly on expert knowledge of the scenario, some parts of this model could be improved by using machine learning, however this is not trivial. To address measurement uncertainty, it is necessary to know models of the perception system. However, these are often unknown or too complicated for use during the exploration phase. Further, the solver starts from a blank state at each execution. A memory could be used to initialize the value function from previous exploration, accelerating the search for the optimal policy. This initialisation could be learned from observation of human behaviour.

The presented model considers only one other vehicle. To scale the model, two part of the framework need to be improved. First, the description of the model should include interaction mutual interaction that each vehicle has with each other. It could create interdependency between the behaviour of each vehicle [134]. The second problem is the solver performance. By adding more vehicle to the environment, the state complexity increases exponentially. A solution is to

break down the problem into a single POMDP for each agent and the subject vehicle to solve each problem in parallel, but then interactions are difficult to consider and the chosen action likely sub-optimal for most of the scene agent [135]. Another solution would be to use auto-encoder to model the environment [136], but this model lacks of understandability.

5 Implementation, Tests and Analysis

5.1	Introduction	136
5.2	Testing Framework	138
5.2.1	Testing Context	138
5.2.2	Testing Architecture	140
5.3	Key Performance Indicators to Evaluate Decision-Making System	145
5.3.1	Approach	146
5.3.2	Safety KPI	148
5.3.3	Trust KPI	148
5.3.4	Comfort KPI	150
5.3.5	Navigation KPI	150
5.3.6	Perspectives on Key Performance Indicators	151
5.4	Experiments and Results	152
5.4.1	Parameters Exploration	152
5.4.2	Behaviour Examples	154
5.4.3	Evaluation of the Decision-Making Performance.	158
5.4.4	Statistical Model Checking	160
5.5	Conclusion	164

5.1 Introduction

Navigation is a complex task due to the randomness in the behaviour of the different entities sharing the same road network. Multiple situations emerge that require the vehicle to adapt its behaviour. For this purpose, it is necessary to have a digital representation of the world through the acquisition of data from multiple sensors. This is then used to reason and to determine the vehicle behaviour before a given situation. Finally, the command signals are generated to

actuate the vehicle with the vehicle control system. These are automated cyber-physical systems (ACPS), and if they are to operate autonomously, need to be safe and to be accepted by society.

To evaluate the decision-making framework proposed in Chapter 4, a statistical argument needs to be built in different scenarios. This is part of the validation process [103]. This has proven to be more difficult than expected. Field experiments are resource intensive, there are multiple scenarios but fundamentally multiple hazardous situations would emerge. The use of simulation techniques is also a challenge, building models is difficult due to the system complexity. Moreover, metrics to assess such systems still need to be defined, there is no consensus. These problems are currently studied by academia and industry, as it is a necessary step for AV deployment [16, 103, 137, 138, 139, 140]. It is considered a strategic issue, as whoever is capable to demonstrate the operability and safety of their autonomous vehicles, will have a major deployment advantage.

The objective of this chapter is to provide an experimental framework to analyse the performance of the proposed decision-making applied to the crossing of a road intersection as per its design objective. This work has contributed also to the validation process developed within the context of the European project Enable-S3¹. This addresses the validation of Automated Cyber Physical Systems for automotive, health, farming, rails, etc. In Enable-S3, a clear separation between the System Under Test (SUT) and the Testing System (TS) is made. Two issues were highlighted: The need of a framework that excites the SUT for multiple scenarios (i.e. the testing space) and the metrics for the evaluation of such systems. Brainstorming in this project highlighted the need for a combined approach, namely physical trials and simulation. In the case of safety critical system, the testing space for physical simulations being limited.

The chapter is divided into four parts:

1. The first rationalises the testing framework by explaining the difficulties facing the testing of the system proposed. The used framework follows the architecture of ENABLE-S3. The testing framework relies on simulations for the experiments to expand the testing space, reduce hazardous situations, and optimise costs.
2. The second introduces Key Performance Indicators (KPI), these are metrics based on an analysis of the scenario and the expected vehicle behaviours. For this purposed, four categories are defined: safety, comfort, navigation and trust.
3. The third shows and discusses results obtained from the testing of the proposed decision-making process using the KPIs defined in the second part. It shows results with different

¹European Initiative to Enable Validation for Highly Automated Safe and Secure Systems

configurations in three scenarios. These can be used to what could be done to improve the system. An application of Statistical Model Checking (SMC) is included to extend the knowledge obtained on the system performance.

4. The Chapter concludes by providing perspectives with regards to the evaluated system and the testing framework.

5.2 Testing Framework

5.2.1 Testing Context

As identified by the NHSTA [141], it is a challenge to include into an autonomous vehicle probabilistic and non-deterministic algorithms as they lack repeatable system outputs. Further, uncertainties identified in the road intersection crossing scenario (c.f. Section 1.2.3) lead to choose probabilistic method for the decision-process. However, the complexity of the models made it difficult to solve. It requires partial exploration of the solution space to achieve real-time execution. Consequently, the system is non-deterministic as the same sequence of observations cannot be guaranteed to have the same associated policy.

This is amplified by the interaction between the SV and the OV. When the action of the SV changes, the reaction of the OV does, and modifies the sequences of observations. Consequently, even with the same initial condition, the scenario can unroll differently. The main causes of randomness are the belief sampling and the generative model (c.f. Section 4.4.3).

The time required to solve the problem offline or the computational power required to achieve a better convergence may not be achievable inside a test vehicle with restricted computation power. A more general issue related to non-determinism, is that the system is subject to many aleatory phenomena. Thus, in the same environmental conditions, the same state measured by the perception may be slightly different after being processed.

The aleatory behaviour of the system can be viewed as dangerous by instances in charge to grant the right to test a system on open. In an industrial and European minded context, the precaution principle prevails, and costly measures must be taken for a test to happen. This is compounded by the likely hazardous conditions that are to emerge as tests are performed. These can result in the destruction of prototypes or injuries. Within this context, it is administratively very difficult and costly to get authorisations for testing in close track and for operating in public

5 Implementation, Tests and Analysis

Test type Feature	Simulation	Test track	Field operation trials
Cost	small	medium	high
Danger	absent	medium	high
Repeatably	high	medium	low
Fidelity	medium	high	high

Table 5.1 Test procedures and associated advantages and drawbacks

roads. Simulation is regarded as a means to ensure the system can operate as expected. Further, it allows to test in different hazardous conditions beyond what could be possible physically.

Simulators are already deployed in the automotive industry to test different characteristics of a vehicle. For example, it is used to understand the interaction between a driver and the vehicle in 360 view angle simulator or to test ADAS functions [142].

There are currently three methods to test AV systems: simulation, closed test tracks or field operational tests in public roads. To build the statistical argument, several kilometres must be driven. It is estimated that billions of hours are necessary considering that some adversarial conditions are encountered [143]. This is costly, thus impossible for universities, SMEs or start-ups.

Even if simulators have a high initial investment cost, the cost of a test is insignificant while repeatably is high and physical hazard avoided. However, fidelity depends on the representativity of the model with respect to reality.

Test track experiments require trained drivers to operate vehicles and to react in case of danger. Tests are repeatable, but the associated logistic is cumbersome. It has the advantage to ensure sensor responses fidelity but lack of behaviour fidelity as human agents follow a previously agreed plan.

Deployment in field operational testing is difficult as the operational cost is high as well as the emergence of hazardous conditions. Their advantage is to gain representative kilometres of driving in a real environment. It allows to gather information about system performance in edge cases not accounted during previous study. However, it is impossible to reproduce a scenario, as they happen randomly. These advantages and draw backs for each method are compiled in table 5.1.

Platform should allow to consider the measurement of as many variables as possible, especially those measurement variables that represent the Key Performance Indicators (KPIs) of the SUT. The approach is based on the capability to cover as many different scenarios as possible by testing physical components to validate the models used for the simulation part. This is based on the premise that testing for edge-cases and hazardous conditions is virtually impossible to do on public roads. Simulation is considered effective though the modelling of sensors and disturbances could be complex.

5.2.2 Testing Architecture

The testing of the proposed approach is made using simulation techniques. This are based on the validation and verification architecture of the Enable-s3 project that brings together the needs of different domains (aviation, automotive, agricultural...). Its objective is to reduce cost associated with testing by using simulation and by choosing an adequate validation methodology. The author contributed to the formulation of this architecture and benefited from the results from an industry wide framework to address the validation of ACPS.

This architecture, shown in Figure 5.1, comprises into three stages:

- The test execution platform that includes the ACPS which includes of the SUT (in our application the decision-making process) and the environment that can be real test tracks or simulations.
- The test management system builds test cases to be performed by the execution platform. The test definition and control processes provide the means to achieve the tested scenario. These can be real means or simulated models with different configurations. This layer also includes systems to record measurements of test runs and to prepare this data for inspection.
- The verification and validation data management consist of systems applied to analyse results from the test framework (test management and the test execution platform) and to control the scenario exploration. The analysis includes a catalogue of KPIs built to examine the SUT in a specific group of scenarios. It includes systems to build scenario from real world traffic data base and systems to instantiate the ACPS with different configurations to be tested.

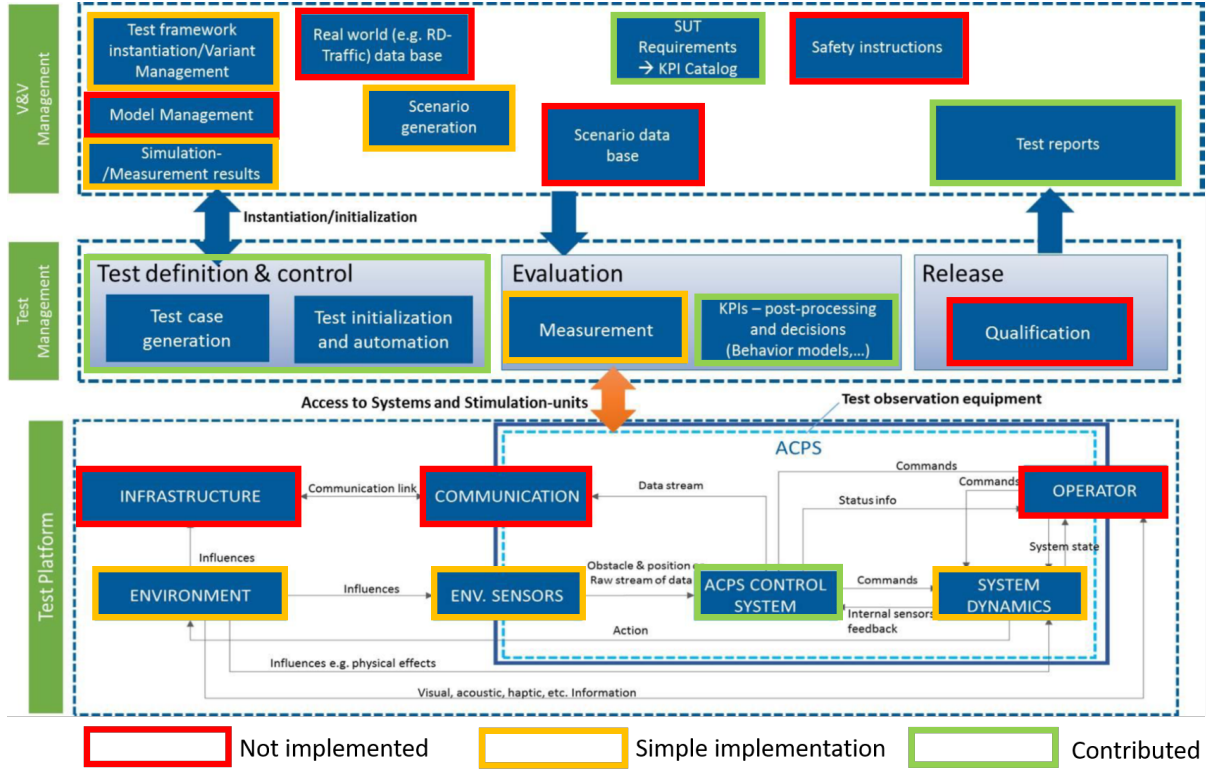


Figure 5.1 The three stages of the Enable-s3 validation and verification architecture, after [144]. The ACPS contains the system under test. Thesis contribution to the ENABLE-S3 architecture are highlighted

5 Implementation, Tests and Analysis

The work in this thesis contributed to the KPI catalogue and the formulation the testing framework that runs tests required by the validation management. Other systems were either implemented in a simple manner or ignored. As part of a joint work with the TAMIS team from Inria Rennes, a blueprint to analyse results obtained by applying Statistical Model Checking is done and presented in Section 5.4.4. Figure 5.1 highlights the different contribution of this thesis to the Enable-s3 architecture.

The testing platform excluded the infrastructure, communication, and operator, because the SUT is developed for a vehicle centric system. Whilst, it is imperative to test all vehicle functions as much as it is statistically credible, this is a major endeavour and still in a exploratory phase. To narrow the scope, other autonomous driving functions such as localisation, perception, control etc. are included in a functional level, that is, they operate nominally. Nevertheless, these simple models add Gaussian noise to environment observations to reproduce in a simple manner measurement uncertainties. A more advanced sensor model needs to include false detection and missed detection to accurately represent issues encountered in our field trials [145].

Simulation is performed using the Scanner² platform, a corporate tool at Renault used for the validation of ADAS applications and different AV functions. Scanner has the capability to represent road networks, generate traffic, simulate sensors and the SV dynamic model. For our test case generation, uses map representing one of Renault’s test sites that resembles the intersection studied in Chapter 3 is used. This was adapted from a lane level map of the test site. On this map, the traffic signal, the OV and the SV can be placed to build a template for the scenario generation system.

The test management system relies on Scanner simulation software functions to manage the different modules required by the test. It restarts the simulation after a given amount of time or after an event happened. The scenarios are executed in a sequential manner. After the execution is completed, measured data are processed by using python scripts to extract the KPIs relevant to the scenario.

Other functions of the Validation and Verification management were kept to a minimum. For example, the scenario generation randomly explores the scenario space. It uses a scenario template from the test case generation and changes its initial parameters. These are: the initial vehicle velocity, the initial distance to the intersection and the signalisation. Changing these values is sufficient to modify interactions happening while approaching the intersection.

²<https://www.avsimulation.fr/>

5 Implementation, Tests and Analysis

The KPI catalogue is constructed from the highway code and what could be considered as an exemplary behaviour for road intersection crossing (c.f. 5.3). The result analysis identifies executions where the ACPS failed to reach the performance required by a KPI. This is improved by applying SMC to vary the performance threshold to obtain a more detailed understanding of system performance.

The main advantage of the Enable-S3 architecture is that it can be applied to different types of ACPS. Thus, an SUT using a different approach, such as rule based or machine learning, can be tested in the same framework to compare results. For example, in Section 5.4, the decision-making process is compared with the one use in the simulator to generate driver behaviour, with the same testing architecture.

Figure 5.2 summarised the architecture of our system in interaction with a simulation environment. The environment simulation models the OV behaviour and the SV dynamics response to an action. This information is shared into the simulated world for the OV to interact with the SV. An interface shares the observed state of the simulated world with the vehicle intelligence. The vehicle intelligence main purpose is to include the SUT, that is the decision-making. It processes and contextualises observations from the environment by using the navigation map. The behaviour understanding system, proposed in Chapter 3, is used to enhance the observation with the OV intention. The SUT generates acceleration commands that are sent to the dynamic model of SV in the simulator via the bridge.

The SUT runs on a Linux machine under the Robotic Operating System (ROS) middle ware. It communicates via an UDP bridge with the simulation tools Scanner on a PC. This architecture needs for their time base (clock) to be synchronised. ROS can receive a clock signal and synchronise with it, thus the Scanner clock is used as the master clock for both systems. A detailed view of the different elements of both the simulation and the SUT is shown in Figure 5.2.

The simulation sends the list of all the entities in the scene. This is contextualised with the HD-map to filter non pertinent entities based on their distance from the subject vehicle. Then, the manoeuvre classification is performed to get the intention observation and observation associated with each vehicle are completed with variables corresponding to the observation-space defined in Section 4.3.5.

Then, the decision-making process uses these observations to return the likely most valuable action. It is transformed to correspond with the interface for the dynamic model of the vehicle part of the simulation to update the SV state for the next time step. Then the simulation world

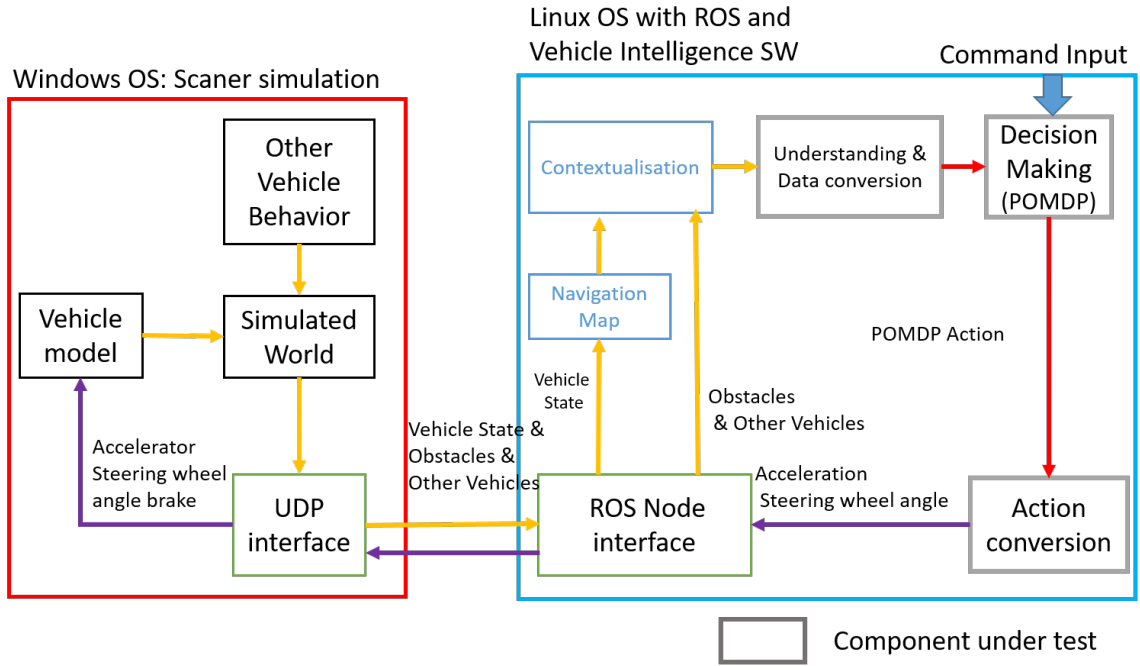


Figure 5.2 Software architecture of the developed system. The environment simulation, inside the red rectangle, is built with Scanner. It updates the environment with respect to the SV actions and returns the state of the environment as observations. Inside the blue rectangle, running under the ROS middle ware, observations are processed and used by the decision-making process to find the most valuable action. This is transformed to match the interface with the simulation.

is updated with the new state of OV and the process is repeated until the simulation run ends after 20 seconds.

From this testing framework, much data is generated and used to compute the KPIs

5.3 Key Performance Indicators to Evaluate Decision-Making System

The preferred metric to evaluate an Automated Vehicle is the Miles per Disengagement, which is the only metric required to be publicly communicated by law to the Department of Motor Vehicles in California (DMV). This reports the average distance vehicles operate without human intervention. It is only an indicator of the progress made. Testing AVs is very costly, the prototypes are expensive likely above the 150.000 Euros. The logistics required are very high. It requires skilled safety drivers, support engineers, safety officers, an infrastructure to store recording, and to analyse data, etc. Therefore, this metric is unfeasible for researchers. It also implies driving in public roads which is hazardous and administratively complex.

There are three types of metrics used to evaluate decision-making processes: to reflect the intrinsic system performance, for example POMDP are often evaluated in terms of the cumulated reward across an execution. To observe simple events that should not happen during operation, for example that no collision occurred during the test. Finally, metric made to compare various alternatives.

The evaluation of a probabilistic a decision-making system in an automotive context requires to go beyond these metrics to better understand the system. This is also required for machine learning approaches (c.f. Section 2.4.3), they are fundamentally based on POMDP. The main complexity factors are:

1. A decision-making process based on uncertain information can only be non-deterministic. In real conditions, it is unlikely that the same observation is obtained for the same state. For example, measurement uncertainty causes variation in the observation of the same scene. It affects the search for the optimal policy that might be initialise with the same belief. This is reinforced by the sampling and generative process that involve randomness (c.f. Section 4.4.1). Thus, the three types of metrics might vary even if the same scenario is tested.

2. These variations are amplified by interactions between the SV and the OV. The POMDP actions have an impact on the OV that might adapt its response. It creates a chain reaction that changes the resolution of the scenario, even with same initial condition. Further, the OV seeks to avoid collisions. These are rare events. Consequently, some events might not appear in simulation and like they are rare in real life (e.g. collisions). It will be difficult to evaluate a metric based on these events.
3. Even if an optimal policy exists, there are multiple local sub-optimal policies for the same scenario. These prevent the solver from finding the optimal policy, nevertheless these are acceptable in driving term. While driving, there are often multiple solutions for the same situation. For example, at a yield intersection both stopping and slowing down are acceptable solutions as it optimises safety for the former and travel time for the latter. Metrics based on the intern performance of the system might not show how correct is the behaviour of the vehicle, furthermore there might be high variance due to these multiple solutions. It also makes comparisons difficult.

5.3.1 Approach

The metrics proposed in this research seek to provide the means to understand and validate a decision-making process applied at road intersection crossing. Whereas, approaches like the Responsibility-Sensitive Safety aims to guarantee the safety of the vehicle at all costs [137], the criteria proposed aims to evaluate the system behaviour.

Key Performance Indicators (KPI) are defined as set of metrics used to evaluate the system in a specific scenario. Each KPI has a bound, used to identify when the system reached a correct, acceptable or failed performance levels. The acceptable level disqualifies a test if one KPI is below the correct performance level but remains below a threshold. It allows to identify, if the result is due to a specific situation that might require a special behaviour. For the road intersection crossing four KPI groups are identified: Safety, Trust, Comfort, Navigation.

Each group has at least one KPI, but the list is not exhaustive, and more could be added when scenarios become more complex.

Three scenarios are studied, these are shown in Figure 5.3. In Scenario A, the SV has to yield to the OV because of the traffic sign. This scenario requires the SV to slow down to let the OV crosses first. In scenario B and C, the SV has the priority and the OV needs to yield (Scenario

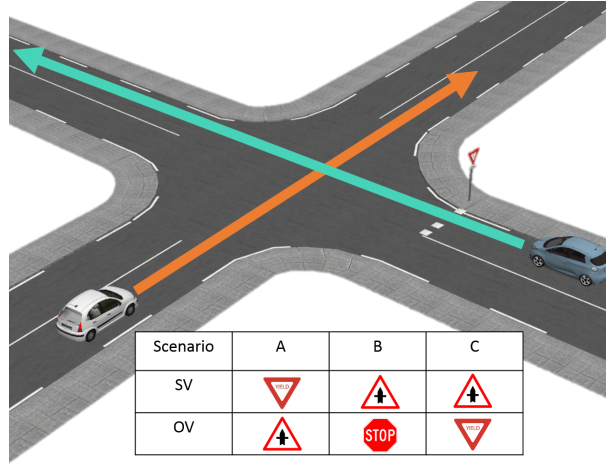


Figure 5.3 Studied scenarios, with the Subject Vehicle in blue and the Other Vehicle in white

B) or to stop (Scenario C). The difficulty is to interact with the OV that might try cross while the SV is approaching.

This work was decorrelated from the reward crafting done in Section 4.3.6. It can be envisaged to include KPIs into the reward function. This solution was not retained for two reasons. First, there is an organisational constraint, in an industrial context, the group charged to develop the navigation system and the group in charge to validate are different. This is to avoid developers to shape their system to pass the validation procedure instead of fulfilling the designated task. The second reason is that similar problems happen when the agent tries to optimise for the testing criteria [105]. In complicated environment, the agent can find hack to optimise its rewards gain by using unusual policies. In our case, the system might decide that stopping every time is the best strategy and would never cross the intersection. Economists face the same problem and formulated this idea as the Goodhart’s law stating that ‘when a metric is used as a target, it ceases to be a good metric’[105]. For a navigation task, an agent rewarded to reduce the travel time could try to accelerate, even if it is unsafe. The multi-objective reward function used in our system is one of the solutions proposed to avoid reward hacking [105], because it forces the agent make compromises.

The remaining of this section presents each group of KPI with examples that will be used to evaluate our decision-making process.

5.3.2 Safety KPI

The selected KPI aims to identify behaviours that are forbidden by the highway code. In every scenario related with road intersection crossing, to stop within the area at the crossroad is forbidden³. It includes the crossing areas, defined in Section 3.2, where most of collisions happen. Stopping there blocks the crossing of other vehicle and creates traffic jams. When an intersection exit is blocked, drivers are not supposed to cross the intersection. This is slightly enforced in France compared with other countries where checker patterns that signal the forbidden area are used, some examples are shown in Figure 5.4a and 5.4b. This unsafe area is shown in red in Figure 5.4c.

Def :Unsafe Stop KPI If the vehicle position is located within the unsafe area and has a speed of 0 km/h , the decision result is considered as failed.

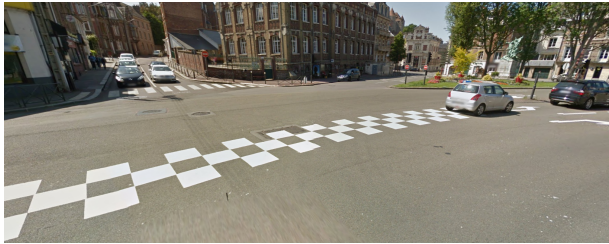
In the studied scenarios, it is not mandatory for the SV to stop before crossing. Thus, the metric associated with stop duration is in the navigation category. It would have been different if the stop manoeuvre would have been enforced by a stop sign.

5.3.3 Trust KPI

Driver trust is crucial to accept a new system, especially if vehicle's controls are shared [36, 146]. A driver/passenger that has lost trust in the system is likely not to use it again. At crossroads, the most stressful moment is at the time of the crossing, the gap with other vehicles must be enough to guarantee that no collision will happen and that there are no OV's in the crossing area. With V2X, gap may be reduced to a couple of seconds [147] and intersection capacity increased [37]. Such optimisation might appear fearsome for passengers that prefer to observe a certain time gap with other vehicles [129]. A criterion for the trust group is the time gap at the intersection entrance.

def :Gap KPI The time gap between the SV and the OV measured at the moment the SV enters the intersection. In the tested scenarios, maintaining the time gap over 4s, as recommended in [129], is mandatory for a successful execution, otherwise the test failed. The measure happens at the crossing event because the time gap changes during the approach. A small gap can be measured at the beginning of the approaching manoeuvre but reduces because of the decision-making actions.

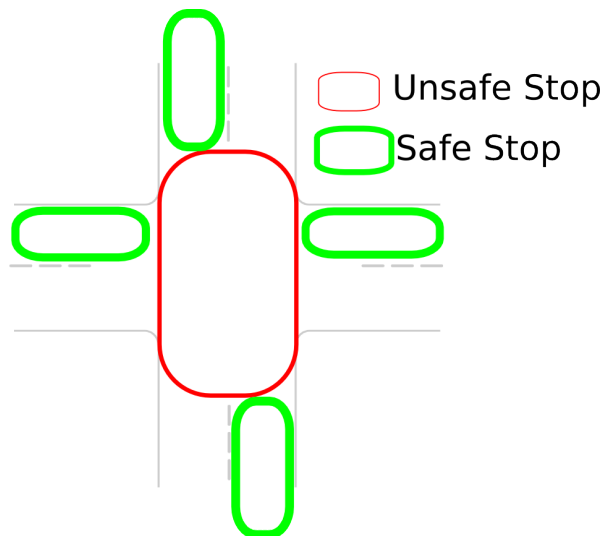
³Article R415-2 of the French highway code



(a) Unsafe area in France signalled with chequer-board marking



(b) Unsafe area in Ireland signalled with a yellow pattern



(c) View of the safe (green) and unsafe (red) areas to stop at a road intersection

Figure 5.4 Intersection layout with unsafe and safe area to stop with examples in two countries

5.3.4 Comfort KPI

In an autonomous vehicle, passenger tasks might be different, as for example office-type ones, thus ride comfort quality becomes imperative. Compared to trust, comfort maybe compromised in situations that require fast motion changes, hence the separation in two groups. Passenger's comfort is not only correlated with lateral and longitudinal accelerations, but also with sudden change of acceleration, also known as jerk [127, 128]. This is measured as the rate of change of acceleration with time. This makes jerk the first derivative of acceleration. In our scenario, longitudinal acceleration is controlled by the decision-making process. Consequently, changes happen to avoid collision and to adapt to the OV behaviour.

Def :Comfort KPI This is the maximum longitudinal jerk during the intersection approach. The jerk needs to remain between $\pm 3m/s^3$ to guarantee passenger comfort [128]. A stricter range of $\pm 2m/s^3$ is chosen for this criterion to identify successful runs.

This lower bound allows to differentiate system that takes pre-emptive actions to avoid collisions and systems that act only to avoid collisions. The former is less likely to get below the bound compared to the latter.

5.3.5 Navigation KPI

This KPI group determines the efficiency of the vehicle system to generate the motion that realises the manoeuvre. For the studied scenarios, two KPIs are identified: travel time and the time stopped before crossing.

Def :Travel Time KPI It evaluates, how fast the system is able to approach and to cross the intersection. Some scenarios (c.f. scenario A) requires the SV to stop as the OV has the priority, thus augmenting the manoeuvre duration. In our scenario, given the initial condition (see Section 5.4) and travel time in simulation (see 3.2.2.1 Figure 3.8), the vehicle should be able to cross in less than 20s even if it must yield. In the scenario B and C, the SV should cross first, thus a lower bound of 15s is chosen.

Def :Safe Stop KPI It considers time the vehicle is at a stop before crossing the intersection. In any of the three scenarios, it is compulsory to stop. However, the system can decide to

reduce its velocity to $0m/s$ when constrained by the OV manoeuvre. It is important that the SV does not stop for a long duration as it will block the traffic. However, stopping might be required to allow the OV to cross. This KPI is defined by the time the vehicle came to a stop before crossing the intersection. The duration of the stop is measured in the green area of Figure 5.4c. The KPI limits to determine the successful operation with this KPI are as follows. In scenario A, when the SV has the priority, it should not stop for more than 3 seconds. In scenario B and C, the yield sign makes the SV more likely to stop for a longer time, but it could also slow down. Therefore, the limit is higher at 5 seconds. In the highway code, there is no mention for a mandatory time to remain stopped at an intersection, even if there is a stop sign⁴. Thus, these bounds are empirically found from observation of the simulation and observation on the test environments.

5.3.6 Perspectives on Key Performance Indicators

The list of KPIs defined for the evaluation of the proposed decision-making framework is summarised in Table 5.2. These are formulated as conditions on measurements. A more formalised manner to define them is presented in Section 5.4.4, where the travel time, safe stop and unsafe stop KPIs are re-formulated by using Binary Linear Temporal logic to be analysed with statistical model checking.

The proposed list of KPIs is not exhaustive. More group and KPIs are required, if more complex scenarios are studied, for example the interaction with pedestrians or if other system components need to be analysed. Recently, the integration of AV as a social agent is studied and could create a new group of KPIs [146]. KPIs can also relate to the different manner that drivers behave across the world. This should lead to the adaptation of the KPIs to the testing context.

Limits for each KPI are found by using expert knowledge or the highway code. For some, it was more difficult to determine an appropriate bound (e.g. Safe Stop) as no rules exist. Observations from traffic may be used to find a value for these bounds. However, to copy human behaviours may not be an adequate target for validation of autonomous system, as humans understand the highway code differently, as solutions are still being developed, further machines perhaps need their own codes.

⁴Article R415-6 of the French highway code

Table 5.2 Condition of success using proposed KPIs

	Scenario A (the SV yields and the OV crosses)				
KPI Tag	Comfort	Trust	Safe stop	Unsafe stop	Travel time
Successful	Lower than $2ms^{-3}$	Over 4s	None	None	-
Acceptable	-	-	Lower than 3s	None	-
Failed	Over than $2ms^{-3}$	Lower than 4s	Over 3s	Over 0s	Over 20s
	Scenario B (the SV crosses and the OV stops) and C (the SV crosses and the OV yields)				
Successful	Lower than $2ms^{-3}$	Over 4s	None	None	-
Acceptable	-	-	Lower than 5s	None	-
Failed	Over than $2ms^{-3}$	Lower than 4s	Over 5s	Over 0s	Over 15s

5.4 Experiments and Results

This section presents the results obtained with different configurations of the proposed decision-making system, the evaluation is made applying the KPIs described in Section 5.3.

5.4.1 Parameters Exploration

The proposed decision-making framework has many parameters, these are divided in two categories: solver parameters and reward weights. The reward weights are made to customise how the system behaves during the approach, whereas the solver parameters affect its capacity to find the optimal solution. The latter were optimised to find solver parameters that are optimal for most of the weighting factors.

There are two solver parameters: the discount factor γ and the trade-off controller C (c.f. 4.4). The lower these values are, the less the POMCP has to search the policy space. It also increases the number of particles that can be sampled at each episode.

To optimise these parameters, 700 runs were made. γ values were sampled⁵ between 0 and 1 with a step 0.05 and C between 0 and 100 with a step of 10. The parameters for the reward weights were randomly selected. Figure 5.5 shows the percentage of successful runs for each combination

⁵The special case with $\gamma = 1$, that corresponds to no time penalty, is handled with a stop condition that stops the search after 40 iterations

5 Implementation, Tests and Analysis

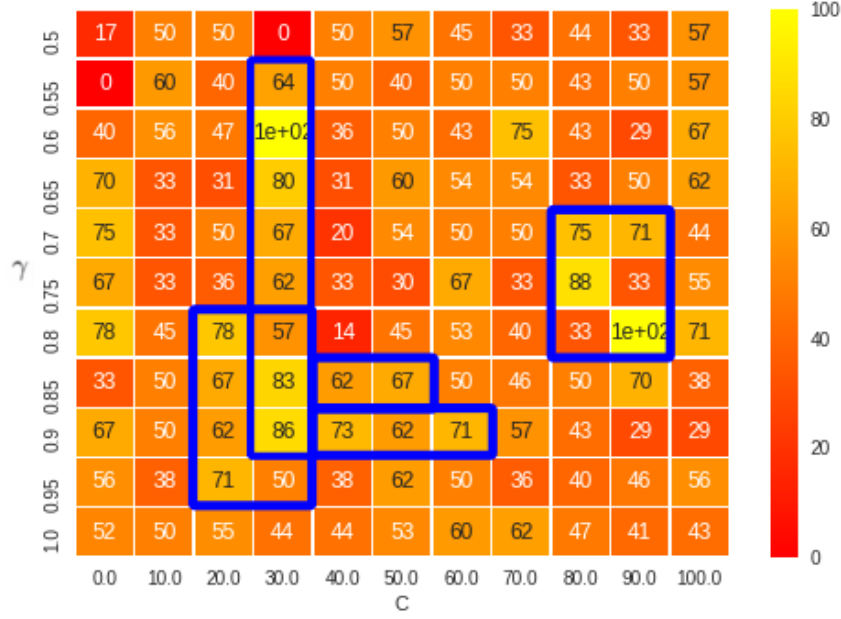


Figure 5.5 Result of the parameter exploration for γ and C , areas with the blue bounds show a high potential for successful crossing

of parameters. These were found by using the KPIs. Areas within the blue bounds display a high potential of success. The optimal configuration was not necessarily the one with the highest chance of success, but has two characteristics: neighbours configuration are also successful and the parameter values are not too big. The configuration with $\gamma = 0.85$ and $C = 30$ is chosen as it meets these characteristics. The discount value results in a search horizon of 12 seconds (see Equation 4.28). This value is appropriate as it means that the search horizon corresponds to most the manoeuvre duration.

With the solver parameters set, reward weight values are explored. Different weight combinations are tested and some configurations with a high potential kept to be tested more intensively. Many different configurations had sufficient performances to be used. For the remaining of the study, two configurations were selected to show and to discuss the advantage of the probabilistic approach and the validation method. They have been selected because of a high success rate, and contrast behaviour in different scenarios, this is presented in following sections. Their parameters are shown in Table 5.3 with the shape of the weighting function shown in Figure 5.6 with respect to the distance to the intersection. The resemblances and differences of these configuration can be observed. For both, none of the reward components is negated, it shows that all the component of the reward function elements are important to cross the intersection. The first configuration

5 Implementation, Tests and Analysis

Table 5.3 Parameters of each weighting function of the reward model

Configuration	1		2	
	k1	k2	k1	k2
w_c	0.5	0.5	0	0.6
w_r	0.1	0.3	0.7	0.9
w_i	-0.5	1	-0.5	1.4
w_e	0.5	0.7	0.5	1.5
w_s	-0.5	1.5	0	0.9
γ	0.85		0.85	
C	30		30	

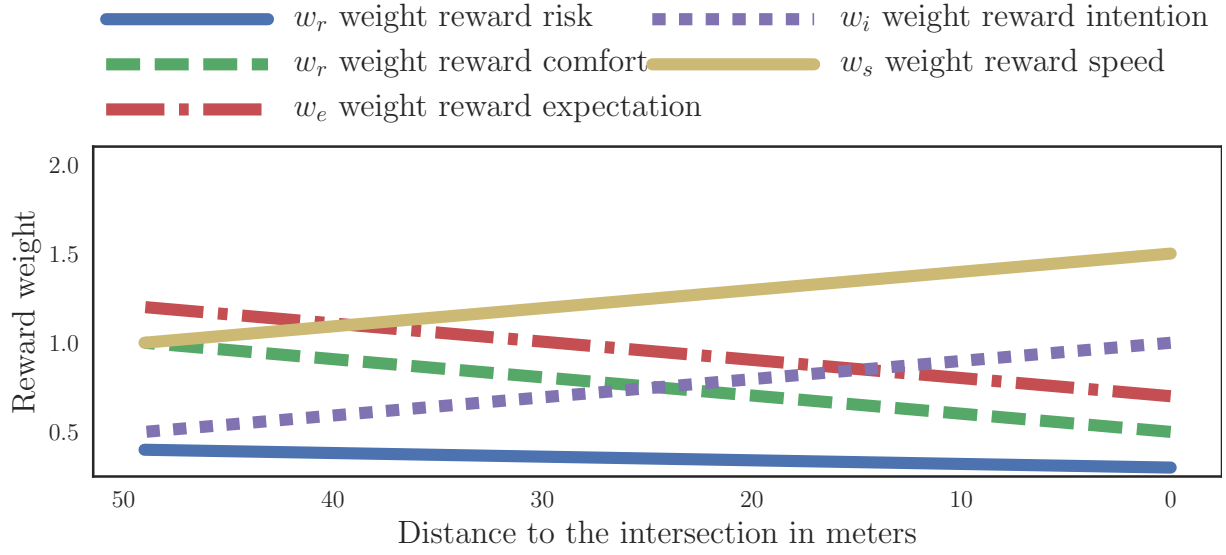
values more the speed reward compare to the second that values the behavioural rewards more.

The selection process of the configuration parameters may be improved by using reinforcement learning to automatically optimise the weight parameters. However, their numbers might not be practical for such approach, they could lead to a local minimal. Another issue with reinforcement learning methods is the large number of required runs. This was not feasible with the current testing platform as they need to run in parallel and with accelerated time.

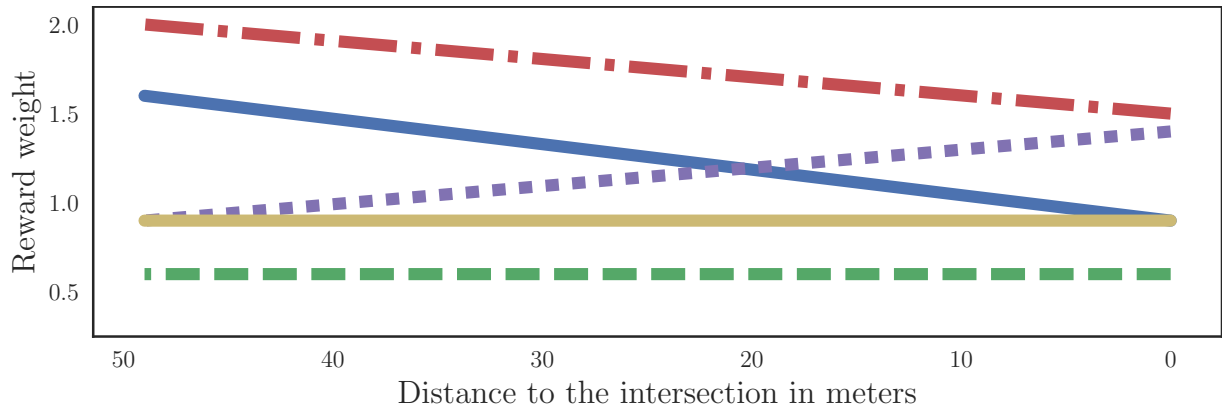
5.4.2 Behaviour Examples

To show how the system behaves approaching the intersection, two examples are discussed using the configuration 1. These results are obtained by applying the test framework presented in Section 5.2. Figure 5.7 shows the two vehicle speeds and the SV actions against time and intersection distance. It demonstrates how the system reacts in the scenarios.

In this scenario (A), the OV has the priority keeping its speed while approaching the intersection. The SV is able to accelerate to reach a higher speed as the OV is likely to cross first, thus risk is low. Strong decelerations ($-1m/s^2$ and $-2m/s^2$) are rarely chosen until the SV is close to the



(a) Configuration 1 weight functions



(b) Configuration 2 weight functions

Figure 5.6 Reward weights as function of the distance to the intersection for the two studied configurations

5 Implementation, Tests and Analysis

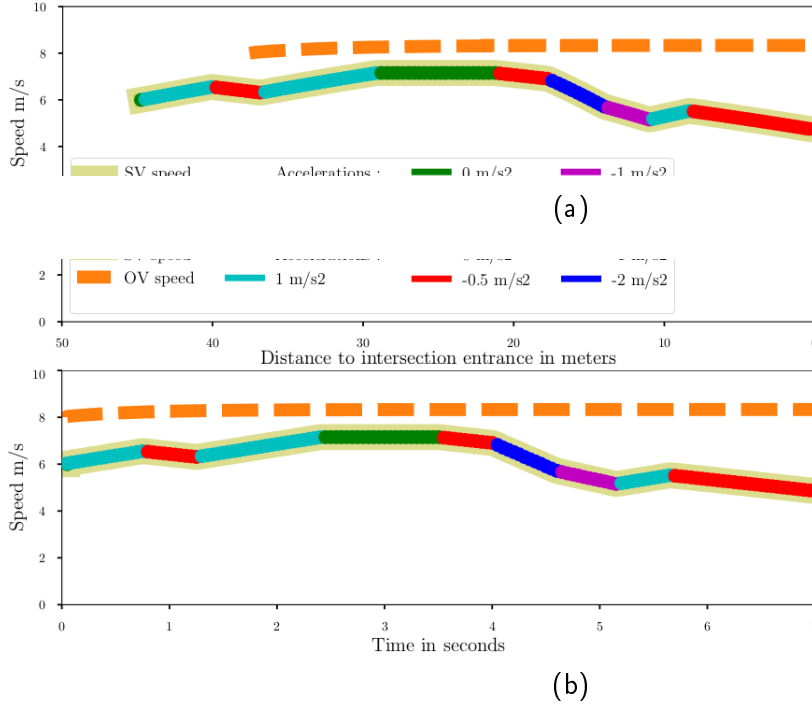
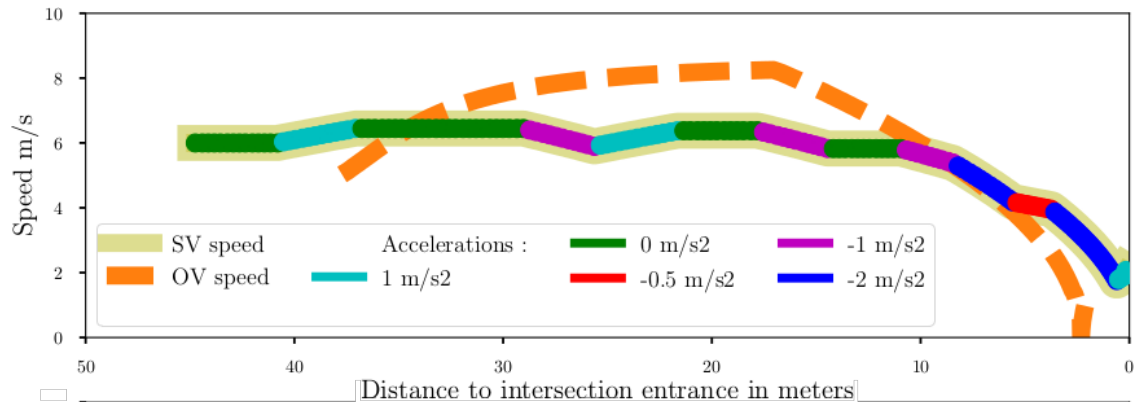


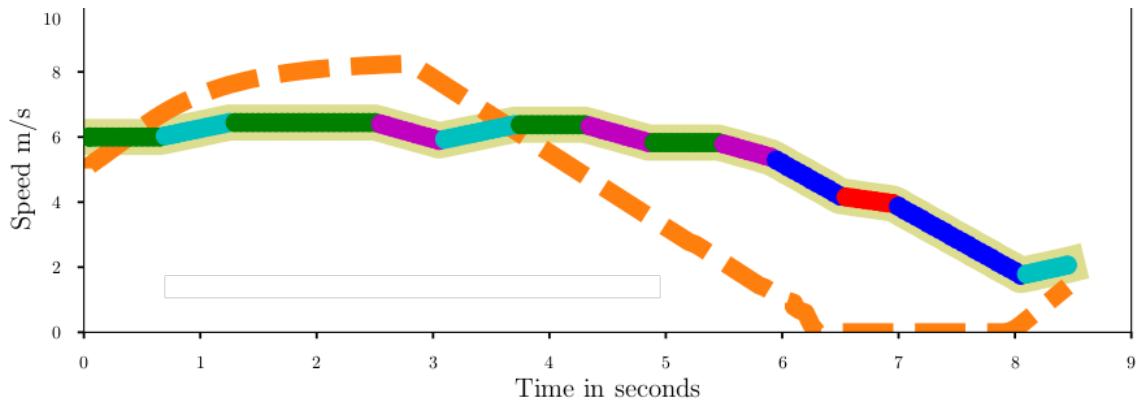
Figure 5.7 Scenario A: Evolution of the speed in function of distance (a) and time (b). Changes of acceleration are shown by the colour of each segment in scenario A with configuration 1

intersection. It corresponds with the reduction of the comfort weight and the importance given to speed close to the intersection entrance. The reason for deceleration after 4s is let the OV crosses with more certainty that expected behaviour of the SV is to yield. The SV reduces its speed as it arrives to the intersection, which given the scenario is what it will be as expected.

The same configuration is applied for Scenario B of our case study (the SV crosses and the OV stops), result is shown in Figure 5.8. The SV keeps its speed to around 6 m/s until it is closer to the intersection (approximately 10 m). At around 3 m to prevent collisions, the OV comes to a stop. Then, SV engages in the intersection by accelerating. As soon as the SV crosses, the OV re-accelerates. It can be observed that a sudden change of accelerations happened at 8 seconds at the intersection entrance, this is not comfortable for passengers.



(a)



(b)

Figure 5.8 Evolution of the speed in function of distance (above) and time (below) Changes of acceleration are shown by the colour of each segment, Scenario B with configuration 1

In these examples, it can be observed that the SV is more prudent than the model used in the simulation. The vehicle moves at slower speeds. The simulated model gets the state of the SV and uses without considering uncertainty, thus it can take more risk.

5.4.3 Evaluation of the Decision-Making Performance.

To understand the system behaviour, it is necessary to build statistics over many tests. For this purpose, the framework presented in Section 5.2 is used to test the decision-process. The configurations presented in Section 5.4.1 are used. The environment, OV and SV's dynamic are modeled using the Scanner simulator. The KPIs defined in Section (5.3) are used to evaluate the performance on the decision-making algorithm in the SV.

To compare the performance the developed system, the model implemented in Scanner is used. The SUT is replaced with this model, that is the same used by the OV. This model uses a perfect knowledge of the environment state. Its decisions are based on the gap between the two vehicles and the signalisation.

To generate statistical data, 800 scenarios are created to run the simulations. This number was chosen because it can cover the scenario space multiple time. These scenarios varied the initial speeds, distances to the intersection and different signalisations. Table 5.4 shows the results of the simulation runs. Percentages are provided for each KPI. It includes entries for the KPIs that showed failures. It is observed that multiple KPIs can be involved to ascertain performance failure. The table shows that each KPI can identify failures in the SUT behaviour.

Only few failures are the product of a long stop duration, but the acceptable limit for this KPI was passed in 91% of failed runs with the baseline model. This high frequency shows that this model is inadequate for yield scenario (A). The main reason for failure in scenario B and C relates to the Gap KPI. Thus, the simulated model is unlikely to be trusted by passenger or driver. This behaviour is unconventional (many stops and small gap). This is likely caused by an optimisation of the comfort and of the travel time.

The overall success rate of the POMDP is higher than the simulated model. In the best case, the proposed system is better by 75% for scenario A, by 2% for scenario B and by 44% for scenario C. Nevertheless, none of the configuration manage to reach a success rate higher than 86% (for scenario C with the configuration 1). The two main reasons of failure are the passenger comfort and the unsafe stop KPIs. The issue of comfort occurs because the POMDP is not penalised

5 Implementation, Tests and Analysis

Table 5.4 Percentage of successful tests and reason for failure out 800 experiments for each configuration from 5.3. The SV yields and the OV crosses for the Scenario A. The SV crosses and the OV stops in scenario B. The SV crosses and the OV yields in scenario C.

Metric	System under test	A	B	C
Success rate	Configuration 1	77%	77%	86%
	Configuration 2	81%	47%	56%
	Simulation model	2%	75%	42%
Safe stop (Acceptable)	1	0	21%	19%
	2	6%	19%	21%
	Simulation	91%	0	10%
Safe stop (Failed)	1	0	0	0
	2	0	1%	1%
	Simulation	0	0	0
Travel time	1	0	0	0
	2	0	46%	38%
	Simulation	0	0	0
Comfort	1	18%	73%	65%
	2	26%	19%	19%
	Simulation	18%	0	2%
Gap	1	81%	0	0
	2	60%	1%	0
	Simulation	6%	100%	89%
Unsafe Stop	1	10%	19%	23%
	2	13%	67%	61%
	Simulation	1%	0	0

to change its actions abruptly. Furthermore, after the SV crosses the intersection, it accelerates so as to get away from the unsafe area. If this action is preceded by a strong deceleration, the execution fails in term of comfort. This can be observed in the second example discussed in Section 5.4.2.

The configuration 1 performance are at least 77% of success for each scenario, whereas the configuration 2 is more adapted to scenario A with 81% chance of success. The main reasons of failures for the second configuration in scenario B and C is triggered by the unsafe stops and the travel time KPIs. It shows that this configuration is likely slower than the other. Upon investigation, some of the unsafe stops are due to the vehicle trying to crawl into the intersection. That is to stop just after the intersection entrance because the OV is trying to cross. The main advantage of the first configuration is that can cross every time the intersection (no failure related with travel time). This configuration is likely to be faster than the configuration 2, however this speed is likely to cause some failures in the scenario A, because it requires the SV to slow down.

Information obtained on the behaviour of each system allows for their enhancement. For example, the first must avoid stopping in an unsafe manner, whereas for the second configuration, the interaction between the SV and the OV should be improved in order to increase the time gap KPI.

5.4.4 Statistical Model Checking

Statistical Model Checking (SMC) [148, 149] provides an intermediate between test and exhaustive verification by relying on statistics. In order to perform SMC, one needs an executable model and a property to check. The executable model is expected to be stochastic, that is, to have some of its transitions governed by probabilistic choices. Most ACPS simulations are already modelled as stochastic processes, because variations in the scenario are defined by probability distributions. The property to check must be decidable on a finite trace. In our application, the properties are based on the KPI and the trace is the recording of the environment state during an execution.

The execution being stochastic, some traces will satisfy the property to check and some other will not. Therefore, the probability that a trace satisfies a property can be determined. The main goal of SMC is to evaluate that probability. This gives more information than a yes-or-no answer as it was realised in the previous sub-section. Indeed, if the model does not satisfy the formula, there is an evaluation of how well it performs.

5 Implementation, Tests and Analysis

In order to perform SMC, one needs to be able to

- Generate traces of the execution of the system to validate. These traces have to be generated according to the probabilities in the model. This is done as part of the test execution.
- Write the property to check as a formula that can be decided on a finite trace, and a procedure for deciding whether a trace satisfies the property.

In order to express such formulas, Bounded Linear Temporal Logic (BLTL) is used. This is a bounded version of Linear temporal logic [150]. In formal verification it allows to state conditions that will eventually be true. The syntax of BLTL is as follows: $\phi ::= p \mid \phi \vee \phi \mid \neg\phi \mid \phi U_{\leq t} \phi \mid X_{\leq t} \phi$. The symbol p represents a predicate expressed on the current state, for instance a comparison between a metric and a bound. The disjunction ($\vee$) and the negation ($\neg$) are defined as usual. The temporal operators until (U) and next (X) define properties about the time. To decide whether a property holds on a finite trace, these operators are parameterised by a time bound $t \in \mathbb{R}$. The formula $X_{\leq t} \phi$ is true if ϕ is true in the state reached after t units of time from the current state. The formula $\phi_1 U_{\leq t} \phi_2$ is true if 1) the formula ϕ_2 becomes true before t units of time from the current state and 2) the formula ϕ_1 remains true in every state before the one where ϕ_2 becomes true. For a formal definition of BLTL semantics, see [151].

In practice, we often use the *always* (G) and *eventually* (F) operators. Eventually is defined as $F_{\leq t} \phi = \mathbf{true} U_{\leq t} \phi$ and means that the formula ϕ should become true before t units of time happen. Always is defined as $G_{\leq t} \phi = \neg F_{\leq t} \neg \phi$ and means that ϕ must always hold for the next t units of time.

With this syntax, KPIs can be reformulated to be used by the SMC. The Table 5.5 shows the variable that are extracted from a trace to build the BLTL statement. To show what could be learned with this approach, three KPIs are reformulated and studied with the SMC. These KPIs are safe stop, unsafe stop and travel time. The method is applied to the traces of the first configuration obtained in the previous section.

The travel time can be stated as:

$$F \leq_d \text{crossed}$$

with d the maximum duration. This means that eventually, after certain duration, the intersection is crossed. The duration is changed to evaluate how fast the tested system can cross.

5 Implementation, Tests and Analysis

Name	Description	Unit
t	Time-stamp or time elapsed	s
s_stops	Number of stops in the safe area	
us_stops	Number of stops in the unsafe area	
t_s_stops	Duration of stops in the safe area	s
t_us_stops	Duration of stops in the unsafe area	s
crossed	True if intersection is crossed	

Table 5.5 List of metrics extracted from the test framework

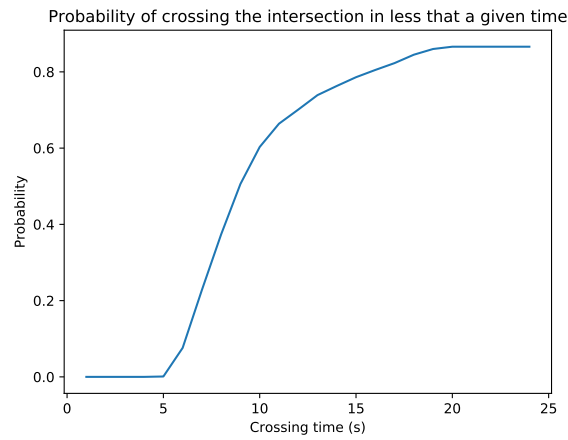
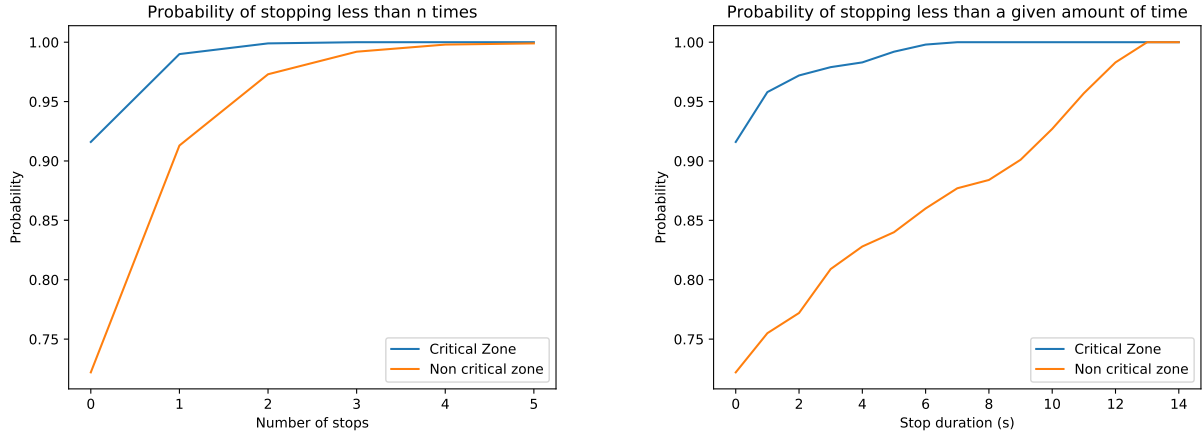


Figure 5.9 travel time KPI studied with SMC

The Figure 5.9 shows the probability for this property to be true depending on the time it takes to cross the intersection. It can be observed that the system is unlikely to cross in less than 5s. It most likely that the system crosses within 16 seconds. It reaches a plateau around after 20 seconds as the simulations stops. As this probability does not reach 1, the system is likely to fail to cross with a probability of 0.1.

The unsafe stop and safe stop are reformulated each into two BLTL to analyse the frequency of stop and their maximum duration. For these metrics, a smaller value indicate a better performance. Consequently, it checks whether metric is bounded by b . The general formula is $G \leq_t m \leq b$. It states that m , the metric, is always smaller than b . Thus, properties are written as:

5 Implementation, Tests and Analysis



(a) Probability to stop in less than a number of times (b) Probability to stop in less than given amount of time

Figure 5.10 Safe and unsafe stops analysis with SMC

$$Safe\ stop\ KPI \begin{cases} G \leq_t s_stops \leq b & frequency \\ G \leq_t t_s_stops \leq b & Duration \end{cases}$$

$$UnSafe\ stop\ KPI \begin{cases} G \leq_t us_stops \leq b & frequency \\ G \leq_t ut_s_stops \leq b & Duration \end{cases}$$

Results obtained by changing the bounds are shown in Figure 5.10a. It can be observed that there is a probability of 0.92 that the system doesn't stop in the critical area which is desirable. The system is also able to stop within the safe area as required by some scenario, however by varying the bound, it can be observed that the probability to stop more than one time is not null. TO stop on multiple occasion is not common and should be investigated. The duration of stops shows that there are often very small, especially in the critical area. It supports the hypothesis that this is due to the vehicle crawling into the intersection. Even if it is unlikely some stops in the safe area are often long, there is a probability of 0.15 that some stops duration are over 6 seconds.

This analysis tells more about the behaviour of the system and could be extended to other KPIs. The syntax associated also allows to build more complex KPI and to build combination of KPI that could be used to better understand the system.

5.5 Conclusion

In this chapter a framework to test an ACPS is presented and applied to a probabilistic decision-making process. It includes the integrate of the SUT within a testing environment that is simulated and to determinate KPI to evaluate the system. These KPIs are found by analysing the scenario and highway code. Four groups have been identified, these are: Safety, trust, comfort, and navigation.

The analysis and results show that the proposed decision-making process can cross an intersection while interacting with another vehicle. The comparison of two configurations shows different performances can be observed in different scenarios. KPIs that are not satisfied by the system can be the next target for the developers.

As the scenario can be known in advance with the map and traffic signs, the two configurations could cohabit and be used only in the scenario that is the most adapted.

The testing framework used for in this Chapter does not includes all the element of the general architecture. Particularly, the scenario exploration, that is simplified in our implementation. To do more test with the current framework would not add more information on the behaviour of the system wasting time and energy. This could be improved with SMC by allowing it to control the scenario generation. Instead of randomly search the scenario space, it looks for the limits between successful and failed execution. It does this search until it reaches a number of simulation sufficient to guarantee that the limits have been found with a known degree of error.

The proposed architecture for testing can be used for any kind of system. It could be used to test the other kind of decision-making process discussed in Chapter 2, especially for machine learning methods that can be formally validated, due to unidentifiable parameters.

6 Conclusion

6.1 Summary of Findings

Autonomous vehicles have been in development for many years. They are expected to reduce road fatalities and to change transportation. However, their deployment is stalling. It is unlikely that full autonomy everywhere is achievable without technological breakthrough in many domains. Nevertheless, autonomy in a predefined constrained operational domain can be possible. In urban areas, complexity comes from interaction with other entities sharing the immediate environment of the subject vehicle. To perceive and predict the motion of these entities is subject to much uncertainties thus decision-making process must consider these to plan the vehicle motion.

This research addressed the navigation of an autonomous vehicle in one of the most complex scenarios of the road network, that is the crossing of road intersections. For the Subject vehicle to decide what action to take next, the manoeuvre of other drivers must be understood. This is achieved by analysing the vehicle motion and its context. The approach taken is based on a learning-based approach, and a method to partition the space into contextually relevant areas. Results showed that it was possible to infer driver manoeuvre and that mixing synthetic and real trajectories helped to simplify the learning process. To contextualize the vehicle motion, a discretisation of motion patterns is realised. The decision process uses a probabilistic framework and the other driver manoeuvre to plan the subject vehicle motion, as it arrives to a crossroad intersection. It shows how uncertainties and the behaviour can be used to achieve interaction. This framework can adapt to different scenario by changing the parameters of its reward model. The performance is difficult to interpret using classical methods as the resulting system behaviour is non-deterministic. For this purpose, a framework for the testing and validation of the decision-making system was developed. It is implemented using a simulator that models the other vehicle behaviour and environments, whilst the decision-making algorithms (system under test) were coded as part of the AV software stack. The performance was evaluated applying a series of purposely defined key performance indicators.

6 Conclusion

The context was determined by analysing motion patterns learnt using Gaussian processes. These are trained with trajectories of vehicles crossing an intersection through simulation models. The resulting motion patterns are segmented to determine crossing and merging areas by finding overlaps between them. A single motion pattern is discretized in multiple zones, these divide the different phases of a driver motion arriving at an intersection. By combining them with the manoeuvre understanding, the manoeuvre classification is improved.

To classify driver manoeuvres, Random Forest Classifiers are applied to learn motion features associated with the three possible manoeuvres at an intersection. These are: Cross, Yield and Stop. A dataset made of synthetic and real trajectories is used to reduce the need for large naturalistic dataset. Results obtained by classifying the manoeuvre of a real driver showed that the proposed framework is better than other possible implementations. A classifier using only synthetic data can determine the manoeuvre, but with few real data added to the dataset, the performance increases rapidly. This reduces the need for large naturalistic dataset, that are complex and costly to obtain.

A Partially Observable Markov Decision Process uses the manoeuvre classification as an observation of other driver intended manoeuvre. It takes into account uncertainties of the interaction between the subject vehicle and the other driver. The subject vehicle is rewarded to interact, to maintain comfort, low risk, and to maintain an appropriate speed. These are balanced differently while the subject vehicle arrives at the intersection. The resulting model is solved by using an online POMDP solver, namely POMCP. It evaluates policies by sampling the current belief and by simulating the effect of the policy actions.

A generic testing framework is developed to interpret the decision-making process performance. It uses the architecture proposed in the Enable-s3 European project on Verification and Validation. Key Performance Indicators are formulated to analyse the behaviour of system under tests with respect to the road intersection crossing scenario. Four categories of KPI are found: comfort, safety, trust and navigation. It determines different problems of the tested system. The approach showed how the decision-making framework in two different configurations behave while approaching and crossing an intersection. Nonetheless, given the strict condition, 20% of executions do not satisfy at least one KPI. This analysis is extended by applying Statistical Model Checking to vary the bound of KPIs to improve the system performance interpretation.

6.2 Conclusions

The results obtained in this thesis lead to three majors conclusions which are described below:

1. **Interaction between vehicles is achieved by using their intended manoeuvre.** To cross safely an intersection, the proposed decision-making process reasons with the intended manoeuvre of the other vehicle. By comparing it with the expected manoeuvre, the system can determine how to interact with the other vehicle. Results from our simulation runs, showed that the system planned its motion as it arrives to the crossroad and anticipated actions of the other vehicle. To obtain the intended manoeuvre, the motion and context of another vehicle are used to classify its manoeuvre. Though, the resultant output includes some degree of uncertainty. The result of the decision-making process is an action in which interaction and risk are balanced.
2. **Uncertainties from perception and interaction should be included as part of the decision process.** The formalism of Partially Observable Markov Decision Process includes elements to consider perception uncertainties (the observation model) and how the situation changes (transition function). These are used to evaluate actions by considering its likely consequences. By considering uncertainties, a trade-off is made between unlikely hazardous events and likely low-risk evolution of the situation. The overall complexity and its application for a real-time system lead to choose an online solver. This solves the problem partially for time horizon. However, it results in variations of the subject vehicle behaviour given the same situation. Even if, the vehicle drives as if under human control, it is difficult to interpret the result of such a system
3. **Key Performances indicators facilitate a better interpretation of the system performance.** The complexity of a system to be deployed for autonomous vehicle makes standard methods for validation complex. Simulation and intensive testing are key elements of the validation processes. The proposed Key Performance Indicators allow to identify tests that are not conform with the expected behaviour at the road intersection. Four categories of KPIs are proposed to study different driving aspects. These are comfort, safety, trust and navigation. Combined with methods such as Statistical Model Checking, these can be used to determinate how complex and non-deterministic systems behave.

6.3 Perspectives

The results of this research work led to industrial interest as well as the need for further work in this area. The following section presents some general perspectives and discusses them in the context of the three previous conclusions mentioned above.

Despite demonstration of the approach validity using a simulation model, this should be validated by testing on physical prototypes. The framework shall be integrated into the architecture of Renault's prototypes. However, due to the tightly integration of the decision-making and navigation systems due to operational constraints, integration proved to be complex. Their system is not flexible in order to ensure safety and coherence between systems. Consequently, modifications are required in the navigation system to integrate the proposed decision system and interfaces need to be developed for the supervisory and control systems. If performances are confirmed with these experimentations, further simulations can be done to refine the results.

To deploy the solutions on real roads, the scalability of the approach needs to be demonstrated. In general, the scalability is related to the effort put into the system to address a larger problem. Most likely, it requires more data, more computational power and more robust implementation. The specific scalability limits of each contribution are discussed below.

In an industrial context, for a vehicle generalist manufacturer, where cost is a major concern. The results could be applied to an Advance Driver Assistance System (ADAS), with SAE Level 2 of autonomy, in addition to long term applications of Level 4 vehicles. [11]. The objective is to help humans during the driving task (e.g. lane keeping, emergency brake system). The differentiation between intention and expectation has already been used to warn driver about risks [122] or to predict the type of assistance required [152] at crossroad intersections. The proposed decision-making system can be used to provide two types of assistance while approaching the crossroad. First, an active assistance could increase or decrease the acceleration of the vehicle if driver's actions are too different compared to the system decisions. This type of assistance nudges the driver to take correct actions and reduces risks. This system could learn from drivers and adapt its reward model in function of the driver. The second is passive, it compares the value of driver's action with the value of the best action. This comparison could be used to score the driver's behaviour after crossing the intersection. This evaluates how well drivers interacted with other at a crossroad. This score can be displayed on an HMI to warn driver about their dangerous behaviours. These types of assistance can be tested with the architecture of the Enable-s3 project, that includes the operator (in our case a driver) in the testing framework. It is to note

that the proposed solution can be well aligned with the future safety vehicle evaluation systems under the Euroncap framework.

6.3.1 Behaviour Understanding

To scale the learning approach proposed in Chapter 3, recording of trajectories at different types of intersections are required. It would show how the approach generalize to new intersections. Likely, different models will be required and adapt to a specific intersection. However, synthetic manoeuvre has been showed to improve the classification. Thus, the time required to observe an intersection can be reduced.

Vehicles are not the only entities interacting with the subject vehicle, there are also pedestrians or cyclists. Similarly, these have an intention motivated by their goal, changing the manner they interact with a vehicle. At a crossroad, the presence of pedestrian crossing is likely, resulting in frequent interactions. There pedestrians can have different crossing intentions that affect the manner they cross [153, 154].

6.3.2 Probabilistic Decision-making

The main limitation to scale the decision-making process is the number of vehicles present in the scene. At the time, a single other vehicle is interacting with the subject vehicle. To increase the number of vehicles is required to address complex situation. The first consequence is a growth of the state-space and observation-space. The resulting model will be harder to solve, as the belief distribution will be harder to estimate. It could be compensated with a better solver or more computational power [87]. A more advanced strategy consists of sampling vehicle states differently based on a importance criterion. This could be for the example their distance to the intersection. Consequently, a vehicle far away from the intersection will have fewer particle describing its states than a vehicle close to the intersection. The second consequence is the complexity of interaction between vehicles. By increasing the number of vehicles, the behaviour variables will be dependent of each of the other vehicle. To predict in this condition is difficult because there is an interdependence between each vehicle.

Another challenge for decision-making process is reasoning with area of the environment that are not yet observed. This is highly difficult as there are many of them in urban area (e.g. area between park vehicles, in front of a bus or occluded intersection branches) from which vulnerable

6 Conclusion

entities might appear. Probabilistic approaches, similar to the one proposed in Chapter 4, are unlikely to scale correctly as they will require lot of computation power [135, 155]. The problem is caused by the worst event being the apparition in front of the vehicle of a fast-moving pedestrian, that needs to be highly penalised in the reward model, and that none of the longitudinal action helps to reduce this risk without resulting of slow-moving vehicle. A potential solution is to incorporate the decision of the lateral motion of the vehicle. By bringing the vehicle to the left side of its lane, hazard from the occluded areas are reduced and subject vehicle speeds maintain. Tough, this increases the action-space and reduces the chance to find the optimal policy evaluation.

The proposed model has many parameters that need to be tuned. This optimisation process could be improved by using genetic algorithm [156] or reinforcement learning [99]. These methods use trial and errors to self-improve in simulation, but there is a risk that the solution overfit on the simulation environment. Some new researches on transfer learning shows how, for simple environment, learned models can be transferred into a different space [157]. This has also some impact on the validation that also uses simulation. If the training and validating environment are too similar, because they are using the same simulator, the evaluation of the performances might not transfer to the real world. Deep learning can be part of the solution to model elements that are difficult to find analytically. However, this comes to a loss of explainability in a system that is already complex. This complexity increase must go jointly with the development of testing solution.

6.3.3 Validation

To validate the system and to build the statistical argument, there is a need to extend the capabilities of the testing platform. The number of simulations need to be increased. It can be improved by running multiple executions in parallel and by accelerating the time in the simulation. Note that the latter is not compatible with the proposed SUT, as it uses the period between two observations to evaluate the value function. Even if the number of tests is increased, to continue to randomly explore the scenario space would be a waste of time and energy. More complex scenarios have more variables, thus the scenario exploration complexity increases drastically the number of scenarios to tests. Further, simulators are often costly to develop and remains accessible to only a few people. In the recent years open-source simulators have been developed but these are more focused on machine learning application and not validation [158].

Consequently, the scenario exploration, part of the Verification and Validation management in

6 Conclusion

the Enable-s3 architecture, becomes an extremely valuable tool. It finds valuable test scenarios prior running the simulation. The complete Statistical Model Checking framework has a retroactive loop that can control the scenario exploration. Its objective is to choose which scenario configuration is the most valuable to increase the accuracy of the KPI probabilistic evaluation.

The validation problem remains a core challenge for the deployment of complex autonomous system. This is not only an effort problem from the industry to test a system but also a research problem to find the correct methods. These are likely to be different given the nature of the tested system.

Bibliography

- [1] P. Cramton, R. R. Geddes, A. Ockenfels, *et al.*, “Using technology to eliminate traffic congestion,” *Journal of Institutional and Theoretical Economics (JITE)*, vol. 175, no. 1, pp. 126–139, 2019. 13
- [2] P. Ross, “Volvo says goodbye to gasoline.” <http://www.scoop.developpement-durable.gouv.fr/en/general-presentation-a9.html>, 2016. last accessed 02/2018. 13
- [3] World Health Organization, “Road traffic injuries.” <http://www.who.int/mediacentre/factsheets/fs358/en/>, Jan. 2018. last accessed 02/2018. 13, 14
- [4] N. Trübswetter and K. Bengler, “Why should i use adas? advanced driver assistance systems and the elderly: knowledge, experience and usage barriers,” in *Proceedings of the 7th International Driving Symposium on Human Factors in Driver Assessment, Training, and Vehicle Design : driving assessment 2013*, University of Iowa, 2013. 13
- [5] E. D. Dickmanns, “Developing the sense of vision for autonomous road vehicles at unbwm,” *Computer*, vol. 50, pp. 24–31, December 2017. 14, 60
- [6] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, *et al.*, “Stanley: The robot that won the darpa grand challenge,” *Journal of field Robotics*, vol. 23, no. 9, pp. 661–692, 2006. 14
- [7] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, *et al.*, “Autonomous driving in urban environments: Boss and the urban challenge,” *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008. 14, 15, 32
- [8] M. Harris, “How google’s autonomous car passed the first u.s. state self-driving test.” <https://spectrum.ieee.org/transportation/advanced-cars/how-googles-autonomous-car-passed-the-first-us-state-selfdriving-test>, Sept. 2014. last accessed 12/2018. 14

BIBLIOGRAPHY

- [9] Google, “Self-driving car test: Steve mahan.” <https://www.youtube.com/watch?v=cdgQpa1pUUE>, march 2012. last accessed 12/2018. 15
- [10] Reuters, “Renault aims to launch paris car-sharing scheme in september.” <https://www.reuters.com/article/us-france-autos/renault-aims-to-launch-paris-car-sharing-scheme-in-september-idUSKBN1JU10J>, 2018. last accessed on 22/07/18. 15
- [11] SAE, “Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles.” <https://www.sae.org/standards/content/j3016201806/>, June 2018. 15, 168
- [12] NHTSB, “Preliminary report highway hwy1 8mh 010.” <https://www.nts.gov/investigations/AccidentReports/Reports/HWY18MH010-prelim.pdf>, 2018. last accessed on 22/05/18. 16
- [13] Waymo, “Waymo’s fully self-driving vehicles are here.” <https://medium.com/waymo/with-waymo-in-the-drivers-seat-fully-self-driving-vehicles-can-transform-the-way-we-get-around-75e9622e829a>, 2017. last accessed on 22/05/18. 16, 33
- [14] Waymo, “Waymo’s safety report: how we are building a safer driver.” <https://medium.com/waymo/waymos-safety-report-how-we-re-building-a-safer-driver-ce5f1b0d4c25>, 2017. last accessed on 22/05/18. 16
- [15] R. A. Daziano, M. Sarrias, and B. Leard, “Are consumers willing to pay to let cars drive for them? analyzing response to autonomous vehicles,” *Transportation Research Part C: Emerging Technologies*, vol. 78, pp. 150 – 164, 2017. 16
- [16] Enable-S3, “Validation and testing of complex automated systems.” <https://www.enable-s3.eu/>. last accessed 09/2019. 16, 137
- [17] E. Gonzalez, “Open autonomous safety.” <https://github.com/voyage/open-autonomous-safety>, 2018. last accessed on 22/05/18. 17
- [18] European Commission, “Traffic safety basic facts 2017: Junction,” 2017. last accessed 05/02/2018. 19, 20, 21, 68
- [19] European Commission, “About care.” https://ec.europa.eu/transport/road_safety/specialist/statistics_en, 2017. last accessed 05/02/2018. 20

BIBLIOGRAPHY

- [20] Washington State Departement of transportation, “Roundabout benefits.” <https://www.wsdot.wa.gov/Safety/roundabouts/benefits.htm>, 2018. last accessed on 22/05/18. 21, 68
- [21] L. A. Rodederdts, B. Nevers, B. Robinson, J. Ringert, P. Koonce, J. Bansen, T. Nguyen, J. McGill, D. Stewart, J. Suggett, *et al.*, “Signalized intersections: informational guide,” tech. rep., Federal Highway Administration Research and Technology, 2004. 21, 69
- [22] L. Fletcher, S. Teller, E. Olson, D. Moore, Y. Kuwata, J. How, J. J. Leonard, I. Miller, M. Campbell, D. Huttenlocher, *et al.*, “The MIT–Cornell collision and why it happened,” *Journal of Field Robotics*, vol. 25, p. 775–807, 2008. 22, 32, 54
- [23] Evan Ackerman, “Fatal tesla self-driving car crash reminds us that robots aren’t perfect.” <https://spectrum.ieee.org/cars-that-think/transportation/self-driving/fatal-tesla-autopilot-crash-reminds-us-that-robots-arent-perfect>, 2016. last accessed 05/06/2018. 22
- [24] F. M. Favaro, N. Nader, S. O. Eurich, M. Tripp, and N. Varadaraju, “Examining accident reports involving autonomous vehicles in california,” *PLOS ONE*, vol. 12, pp. 1–20, 09 2017. 22
- [25] S. M. LaValle and R. Sharma, “On motion planning in changing, partially predictable environments,” *The International Journal of Robotics Research*, vol. 16, no. 6, pp. 775–805, 1997. 24
- [26] M. Barbier, C. Laugier, O. Simonin, and J. Ibañez-Guzmán, “Functional discretization of space using gaussian processes for road intersection crossing,” in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 156–162, Nov 2016. 28
- [27] M. Barbier, C. Laugier, O. Simonin, and J. Ibañez-Guzmán, “Classification of drivers manoeuvre for road intersection crossing with synthethic and real data,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 224–230, June 2017. 28, 46
- [28] S. Ulbrich and M. Maurer, “Probabilistic online pomdp decision making for lane changes in fully automated driving,” in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, pp. 2063–2067, Oct 2013. 32

BIBLIOGRAPHY

- [29] H. Bai, S. Cai, N. Ye, D. Hsu, and W. S. Lee, “Intention-aware online pomdp planning for autonomous driving in a crowd,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 454–460, May 2015. 32
- [30] M. Bouton, A. Cosgun, and M. J. Kochenderfer, “Belief state planning for autonomously navigating urban intersections,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 825–830, June 2017. 32, 112, 115, 123
- [31] T. Gindele, S. Brechtel, and R. Dillmann, “Learning driver behavior models from traffic observations for decision making and planning,” *Intelligent Transportation Systems Magazine, IEEE*, vol. 7, pp. 69–79, Spring 2015. 32
- [32] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, “End to end learning for self-driving cars,” *arXiv preprint arXiv:1604.07316*, 2016. 32, 60, 61
- [33] J. Ibanez-Guzmán, C. Laugier, J.-D. Yoder, and S. Thrun, *Autonomous Driving: Context and State-of-the-Art*, pp. 1271–1310. London: Springer London, 2012. 33, 38, 39
- [34] L. Fridman, “Human-centered autonomous vehicle systems: Principles of effective shared autonomy,” *arXiv preprint arXiv:1810.01835*, 2018. 34, 35
- [35] E. Cha, A. D. Dragan, and S. S. Srinivasa, “Perceived robot capability,” *2015 24th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 541–548, 2015. 35
- [36] S. H. Huang, K. Bhatia, P. Abbeel, and A. D. Dragan, “Establishing appropriate trust via critical states,” *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018. 35, 148
- [37] K. Dresner and P. Stone, “Multiagent traffic management: A reservation-based intersection control mechanism,” in *The Third International Joint Conference on Autonomous Agents and Multiagent Systems*, pp. 530–537, July 2004. 36, 148
- [38] M. Tlig, O. Buffet, and O. Simonin, “Stop-free strategies for traffic networks: Decentralized on-line optimization,” in *Proceedings of the Twenty-first European Conference on Artificial Intelligence, ECAI’14*, pp. 1191–1196, IOS Press, 2014. 36
- [39] K. Katsaros and M. Dianati, *A Conceptual 5G Vehicular Networking Architecture*, pp. 595–623. Cham: Springer International Publishing, 2017. 36

BIBLIOGRAPHY

- [40] SCOOP, “Connected vehicle and roads.” <http://www.scoop.developpement-durable.gouv.fr/en/general-presentation-a9.html>, 2016. last accessed 05/02/2018. 36
- [41] SAFESPOT, “Cooperative vehicles and road infrastructure for road safety.” <http://www.safespot-eu.org/>, 2010. last accessed 05/02/2018. 36
- [42] L. Chen and C. Englund, “Cooperative intersection management: A survey,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, pp. 570–586, Feb 2016. 36
- [43] J. Ibañez-Guzmán, S. Lefevre, A. Mekkadem, and S. Rodhaim, “Vehicle to vehicle communications applied to road intersection safety, field results,” in *2010 13th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 192–197, Sept 2010. 37
- [44] J. S. A. Alexander M. Meystel, *Intelligent Systems: Architecture, Design, and Control*. John Wiley & Sons, Inc., 2001. 38, 39, 58
- [45] M. Simon, S. Milz, K. Amende, and H.-M. Gross, “Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds,” in *European Conference on Computer Vision*, pp. 197–209, Springer, 2018. 41, 47
- [46] L. Rummelhard, A. Nègre, M. Perrollaz, and C. Laugier, “Probabilistic Grid-based Collision Risk Prediction for Driving Application,” in *ISER*, June 2014. 41
- [47] J. Zhang and S. Singh, “Loam: Lidar odometry and mapping in real-time,” in *Robotics: Science and Systems*, vol. 2, p. 9, 2014. 42
- [48] F. Li, P. Bonnifait, J. Ibanez-Guzman, and C. Zinoune, “Lane-level map-matching with integrity on high-definition maps,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1176–1181, June 2017. 42
- [49] M. Nolte, S. Ernst, J. Richelmann, and M. Maurer, “Representing the unknown - impact of uncertainty on the interaction between decision making and trajectory generation,” *CoRR*, vol. abs/1804.08871, 2018. 43
- [50] S. Lefevre, D. Vasquez, and C. Laugier, “A survey on motion prediction and risk assessment for intelligent vehicles,” *ROBOMECH Journal*, vol. 1, no. 1, pp. 1–14, 2014. 44
- [51] J. Rasmussen, *Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*. New York, NY, USA: Elsevier Science Inc., 1986. 44

BIBLIOGRAPHY

- [52] N. A. Stanton and P. M. Salmon, “Human error taxonomies applied to driving: A generic driver error taxonomy and its implications for intelligent transport systems,” *Safety Science*, vol. 47, no. 2, pp. 227 – 237, 2009. 44
- [53] S. Lefevre, C. Laugier, and J. Ibañez-Guzmán, “Risk assessment at road intersections: Comparing intention and expectation,” in *2012 IEEE Intelligent Vehicles Symposium (IV)*, pp. 165–171, June 2012. 46, 47, 51, 52, 66, 118, 119, 126
- [54] A. Armand, D. Filliat, and J. Ibañez-Guzmán, “Modelling stop intersection approaches using gaussian processes,” in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, Oct 2013. 46, 47, 71, 72, 73, 77
- [55] W. Liu, S.-W. Kim, and M. H. Ang, “Probabilistic road context inference for autonomous vehicles,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1640–1647, May 2015. 46, 47
- [56] S. Sivaraman and M. M. Trivedi, “Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 1773–1795, Dec 2013. 47, 48
- [57] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005. 47
- [58] F. Gross, J. Jordan, F. Weninger, F. Klanner, and B. Schuller, “Route and stopping intent prediction at intersections from car fleet data,” *IEEE Transactions on Intelligent Vehicles*, vol. 1, pp. 177–186, June 2016. 49, 50, 53, 98, 99
- [59] M. G. Ortiz, J. Fritsch, F. Kummert, and A. Gepperth, “Behavior prediction at multiple time-scales in inner-city scenarios,” in *2011 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1068–1073, June 2011. 49, 50
- [60] D. J. Phillips, T. A. Wheeler, and M. J. Kochenderfer, “Generalizable intention prediction of human drivers at intersections,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1665–1670, June 2017. 49
- [61] A. Tawari, T. Misu, and K. Fujimura, “Predicting unexpected maneuver while approaching intersection,” in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2225–2229, Nov 2016. 49, 50

BIBLIOGRAPHY

- [62] S. B. Amsalu, A. Homaifar, F. Afghah, S. Ramyar, and A. Kurt, “Driver behavior modeling near intersections using support vector machines based on statistical feature extraction,” in *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1270–1275, June 2015. 49
- [63] A. Khosroshahi, E. Ohn-Bar, and M. M. Trivedi, “Surround vehicles trajectory analysis with recurrent neural networks,” in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2267–2272, Nov 2016. 49
- [64] G. Aoude, V. Desaraju, L. Stephens, and J. How, “Driver behavior classification at intersections and validation on large naturalistic data set,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, pp. 724–736, June 2012. 50, 51, 53, 86, 99
- [65] T. Streubel and K. H. Hoffmann, “Prediction of driver intended path at intersections,” in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pp. 134–139, June 2014. 50, 51, 53
- [66] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka, “Interaction-aware probabilistic behavior prediction in urban environments,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3999–4006, IEEE, 2018. 50, 51, 52
- [67] S. Lefevre, C. Laugier, and J. Ibañez-Guzmán, “Exploiting map information for driver intention estimation at road intersections,” in *2011 IEEE Intelligent Vehicles Symposium (IV)*, pp. 583–588, June 2011. 51
- [68] S. Lefevre, J. Ibañez-Guzmán, and C. Laugier, “Context-based estimation of driver intent at road intersections,” in *2011 IEEE Symposium on Computational Intelligence in Vehicles and Transportation Systems (CIVTS) Proceedings*, pp. 67–72, April 2011. 51, 53
- [69] S. Brechtel, T. Gindele, and R. Dillmann, “Solving continuous pomdps: Value iteration with incremental learning of an efficient space representation,” in *In Proc. Int. Conf. on Machine Learning*, 2013. 51
- [70] M. S. Shirazi and B. T. Morris, “Looking at intersections: A survey of intersection monitoring, behavior and safety analysis of recent studies,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, pp. 4–24, Jan 2017. 53
- [71] R. Ponziani, “Turn signal usage rate results: A comprehensive field study of 12,000 observed turning vehicles,” tech. rep., SAE Technical Paper, 2012. 53

BIBLIOGRAPHY

- [72] C. Baker and J. Dolan, “Traffic interaction in the urban challenge: Putting boss on its best behavior,” in *International Conference on Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ*, pp. 1752–1758, Sept 2008. 54, 55
- [73] J. K. Rosenblatt, “Damn: A distributed architecture for mobile navigation,” *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 9, no. 2-3, pp. 339–360, 1997. 54
- [74] A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. Reinholtz, D. Hong, A. Wicks, T. Alberi, D. Anderson, *et al.*, “Odin: Team victortango’s entry in the darpa urban challenge,” *Journal of Field Robotics*, vol. 25, no. 8, pp. 467–492, 2008. 54, 55
- [75] Darpa, “Urban Challenge Route Network Definition File (RNDF) and Mission Data File (MDF) Formats,” 2007. 55
- [76] Y. Liu and U. Ozguner, “Human driver model and driver decision making for intersection driving,” in *Intelligent Vehicles Symposium, 2007 IEEE*, pp. 642–647, June 2007. 56, 82
- [77] A. Furda and L. Vlacic, “Enabling safe autonomous driving in real-world city traffic using multiple criteria decision making,” *Intelligent Transportation Systems Magazine, IEEE*, vol. 3, pp. 4–17, Spring 2011. 56
- [78] R. Regele, “Using ontology-based traffic models for more efficient decision making of autonomous vehicles,” in *Fourth International Conference on Autonomic and Autonomous Systems (ICAS’08)*, pp. 94–99, March 2008. 56
- [79] L. Zhao, R. Ichise, Y. Sasaki, Z. Liu, and T. Yoshikawa, “Fast decision making using ontology-based knowledge base,” in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 173–178, June 2016. 56
- [80] R. Bellman, “A markovian decision process,” *Indiana Univ. Math. J.*, vol. 6, pp. 679–684, 1957. 57, 129
- [81] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. New York, NY, USA: John Wiley & Sons, Inc., 1st ed., 1994. 57
- [82] A. R. Cassandra, “A survey of pomdp applications,” in *Working notes of AAAI 1998 fall symposium on planning with partially observable Markov decision processes*, vol. 1724, 1998. 57

BIBLIOGRAPHY

- [83] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, “Planning and acting in partially observable stochastic domains,” *Artificial intelligence*, vol. 101, no. 1, pp. 99–134, 1998. 57
- [84] T. Smith and R. Simmons, “Heuristic search value iteration for pomdps,” in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pp. 520–527, AUAI Press, 2004. 59
- [85] S. Thrun, “Monte carlo POMDPs,” in *Advances in Neural Information Processing Systems 12*, pp. 1064–1070, MIT Press, 2000. 59, 129
- [86] N. Roy, G. Gordon, and S. Thrun, “Finding approximate POMDP solutions through belief compression,” *Journal of Artificial Intelligence Research*, 2004. 59
- [87] H. Kurniawati, D. Hsu, and W. S. Lee, “Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces,” in *In Proc. Robotics: Science and Systems*, 2008. 59, 169
- [88] D. Silver and J. Veness, “Monte-carlo planning in large pomdps,” in *Advances in Neural Information Processing Systems 23*, pp. 2164–2172, Curran Associates, Inc., 2010. 59, 129, 130
- [89] S. Gelly and D. Silver, “Combining online and offline knowledge in uct,” in *Proceedings of the 24th International Conference on Machine Learning*, ICML ’07, pp. 273–280, ACM, 2007. 59
- [90] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016. 59
- [91] L. Wei, *Planning under Uncertainties for Autonomous Driving on Urban Road*. PhD thesis, National university of Singapore, 2016. 59, 115
- [92] S. Brechtel, T. Gindele, and R. Dillmann, “Probabilistic decision-making under uncertainty for autonomous driving using continuous pomdps,” in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pp. 392–399, Oct 2014. 59, 60, 65, 112, 123
- [93] C. Hubmann, M. Becker, D. Althoff, D. Lenz, and C. Stiller, “Decision making for autonomous driving considering interaction and uncertain prediction of surrounding vehicles,” in *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pp. 1671–1678, IEEE, 2017. 60

BIBLIOGRAPHY

- [94] W. Liu, S. W. Kim, S. Pendleton, and M. H. Ang, “Situation-aware decision making for autonomous driving on urban road using online pomdp,” in *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1126–1133, June 2015. 60
- [95] K. H. Wray, S. J. Witwicki, and S. Zilberstein, “Online decision-making for scalable autonomous systems,” in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp. 4768–4774, 2017. 60
- [96] M. Hasenjager and H. Wersing, “Personalization in advanced driver assistance systems and autonomous vehicles: A review,” in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–7, Oct 2017. 61
- [97] M. Kuderer, S. Gulati, and W. Burgard, “Learning driving styles for autonomous vehicles from demonstration,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2641–2646, May 2015. 61
- [98] D. Sierra González, Ö. Erkent, V. Romero-Cano, J. Dibangoye, and C. Laugier, “Modeling Driver Behavior From Demonstrations in Dynamic Environments Using Spatiotemporal Lattices,” in *ICRA 2018 - Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, (Brisbane, Australia), May 2018. 61
- [99] L. Fridman, B. Jenik, and J. Terwilliger, “DeepTraffic: Driving Fast through Dense Traffic with Deep Reinforcement Learning,” *ArXiv e-prints*, Jan. 2018. 62, 170
- [100] D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, and K. Fujimura, “Navigating Occluded Intersections with Autonomous Vehicles using Deep Reinforcement Learning,” *ArXiv e-prints*, May 2017. 62
- [101] A. Bewley, J. Rigley, Y. Liu, J. Hawke, R. Shen, V.-D. Lam, and A. Kendall, “Learning to Drive from Simulation without Real World Labels,” *arXiv e-prints*, p. arXiv:1812.03823, Dec. 2018. 62, 63, 87
- [102] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah, “Learning to drive in a day,” *2019 International Conference on Robotics and Automation (ICRA)*, May 2019. 62, 65, 85
- [103] APTIV, “Safety first for automated driving.” <https://www.aptiv.com/docs/default-source/white-papers/safety-first-for-automated-driving-aptiv-white-paper.pdf>, 2019. Last accessed on 08/08/19. 64, 137

BIBLIOGRAPHY

- [104] M. Henaff, A. Canziani, and Y. LeCun, “Model-Predictive Policy Learning with Uncertainty Regularization for Driving in Dense Traffic,” *arXiv e-prints*, p. arXiv:1901.02705, Jan. 2019. 65
- [105] D. Amodei, C. Olah, J. Steinhardt, P. F. Christiano, J. Schulman, and D. Mané, “Concrete problems in AI safety,” *ArXiv e-prints*, 2016. 65, 147
- [106] C. Tay and C. Laugier, “Modelling smooth paths using gaussian processes,” in *Proc. of the Int. Conf. on Field and Service Robotics*, (Chamonix, France), 2007. 71, 72, 73
- [107] S. Basu, D. Bonafilia, J. Gill, D. Kirsanov, and D. Yang, “Mapping roads through deep learning and weakly supervised training.” <https://ai.facebook.com/blog/mapping-roads-through-deep-learning-and-weakly-supervised-training>, 2019. last accessed on 23/07/19. 71
- [108] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun, “Learning motion patterns of people for compliant robot motion,” *The International Journal of Robotics Research*, vol. 24, no. 1, pp. 31–48, 2005. 71, 72
- [109] Z. Ghahramani and S. Roweis, “Learning nonlinear stochastic dynamics using the generalized em algorithm,” *NIPS*, 1998. 71
- [110] C. E. Rasmussen, *Gaussian processes for machine learning*. MIT Press, 2006. 72
- [111] D. Vasquez, C. Laugier, and T. Fraichard, “Incremental learning of statistical motion patterns with growing hidden markov models,” in *Robotics Research*, pp. 75–86, Springer Berlin Heidelberg, 2011. 73
- [112] O. Sigaud, C. Salaün, and V. Padois, “On-line regression algorithms for learning mechanical models of robots: a survey,” *Robotics and Autonomous Systems*, vol. 59, pp. 1115–1129, July 2011. 73
- [113] C. M. Bishop *et al.*, *Neural networks for pattern recognition*. Oxford university press, 1995. 76
- [114] K. Chellapilla, “Rethinking maps for self-driving.” <https://medium.com/@LyftLevel5/https-medium-com-lyftlevel5-rethinking-maps-for-self-driving-a147c24758d6>, 2018. last accessed 05/02/2019. 85

BIBLIOGRAPHY

- [115] “Spatialite 5.0.0-beta01, sql functions reference list.” <http://www.gaia-gis.it/gaia-sins/spatialite-sql-5.0.0.html>, 2018. last accessed 05/02/2019. 86
- [116] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001. 89
- [117] T. Sharp, *Implementing Decision Trees and Forests on a GPU*, pp. 595–608. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008. 89
- [118] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011. 94
- [119] L. Rummelhard, A. Negre, and C. Laugier, “Conditional monte carlo dense occupancy tracker,” in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*, pp. 2485–2490, Sept 2015. 104
- [120] Y. Saatçi, R. D. Turner, and C. E. Rasmussen, “Gaussian process change point models,” in *ICML*, pp. 927–934, 2010. 105
- [121] P. de Beaucorps, T. Streubel, A. Verroust-Blondet, F. Nashashibi, B. Bradai, and P. Resende, “Decision-making for automated vehicles at intersections adapting human-like behavior,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 212–217, June 2017. 105
- [122] S. Lefevre, *Risk estimation at road intersections for connected vehicle safety applications*. Theses, Université de Grenoble, #Oct# 2012. 114, 168
- [123] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, “Constrained model predictive control: Stability and optimality,” *Automatica*, vol. 36, no. 6, pp. 789–814, 2000. 115
- [124] M. A. Pollatschek, A. Polus, and M. Livneh, “A decision model for gap acceptance and capacity at intersections,” *Transportation Research Part B: Methodological*, vol. 36, no. 7, pp. 649 – 663, 2002. 118
- [125] M. Bansal, A. Krizhevsky, and A. Ogale, “Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst,” 2018. 120
- [126] T. Han, J. Jing, and U. Ozguner, “Driving intention recognition and lane change prediction on the highway,” in *Intelligent Vehicles Symposium (IV), 2019 IEEE*, June 2019. 120

BIBLIOGRAPHY

- [127] K. Yi and J. Chung, “Nonlinear brake control for vehicle cw/ca systems,” *IEEE/ASME Transactions on Mechatronics*, vol. 6, pp. 17–25, Mar 2001. 123, 150
- [128] J. Martinez and C. Canudas-de-Wit, “A safe longitudinal control for adaptive cruise control and stop-and-go scenarios,” *IEEE Transactions on Control Systems Technology*, vol. 15, pp. 246–258, March 2007. 123, 150
- [129] K. Fitzpatrick, “Gaps accepted at stop-controlled intersections,” *Transportation Research Record*, vol. 1303, no. 11, pp. 103–112, 1991. 124, 148
- [130] A. Somani, N. Ye, D. Hsu, and W. S. Lee, “Despot: Online pomdp planning with regularization,” in *Advances in Neural Information Processing Systems 26*, pp. 1772–1780, Curran Associates, Inc., 2013. 129
- [131] H. Kurniawati and V. Yadav, “An online pomdp solver for uncertainty planning in dynamic environment,” in *ISRR*, 2013. 129
- [132] D. Klimenko, J. Song, and H. Kurniawati, “Tapir: A software toolkit for approximating and adapting pomdp solutions online,” in *Proceedings of the Australasian Conference on Robotics and Automation, Melbourne, Australia*, vol. 24, 2014. 129
- [133] P. Auer, N. Cesa-Bianchi, and P. Fischer, “Finite-time analysis of the multiarmed bandit problem,” *Machine learning*, vol. 47, no. 2-3, pp. 235–256, 2002. 130
- [134] C. Hubmann, J. Schulz, M. Becker, D. Althoff, and C. Stiller, “Automated driving in uncertain environments: Planning with interaction and uncertain maneuver prediction,” *IEEE Transactions on Intelligent Vehicles*, vol. 3, pp. 5–17, March 2018. 134
- [135] M. Schraner, M. Bouton, M. J. Kochenderfer, and D. Watzenig, “Pedestrian collision avoidance system for scenarios with occlusions,” *2019 IEEE Intelligent Vehicles Symposium (IV)*, Jun 2019. 135, 170
- [136] D. Ha and J. Schmidhuber, “World models,” *CoRR*, vol. abs/1803.10122, 2018. 135
- [137] S. Shalev-Shwartz, S. Shammah, and A. Shashua, “On a formal model of safe and scalable self-driving cars,” *arXiv preprint arXiv:1708.06374*, 2017. 137, 146
- [138] M. Barbier, A. Renzaglia, J. Quilbeuf, L. Rummelhard, A. Paigwar, C. Laugier, A. Legay, J. Ibañez-Guzmán, and O. Simonin, “Validation of Perception and Decision-Making Systems for Autonomous Driving via Statistical Model Checking,” in *IV 2019 - 30th IEEE Intelligent Vehicles Symposium*, (Paris, France), pp. 1–8, IEEE, June 2019. 137

BIBLIOGRAPHY

- [139] PEGASUS, “Pegasus.” <https://www.pegasusprojekt.de/en/>. last accessed 02/2018. 137
- [140] T. Bokc, M. Maurer, and G. Farber, “Validation of the vehicle in the loop (vil); a milestone for the simulation of driver assistance systems,” in *2007 IEEE Intelligent Vehicles Symposium*, pp. 612–617, June 2007. 137
- [141] NHSTA, “A framework for automated driving system testable cases and scenarios,” 2018. Last accessed on 08/08/19. 138
- [142] A. Broggi, M. Buzzoni, S. Debattisti, P. Grisleri, M. C. Laghi, P. Medici, and P. Versari, “Extensive tests of autonomous driving technologies,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, pp. 1403–1415, Sept 2013. 139
- [143] P. Koopman and M. Wagner, “Challenges in autonomous vehicle testing and validation,” *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, 2016. 139
- [144] Enable-S3, “Validation and testing of complex automated systems.” <https://www.enable-s3.eu/>. last accessed 09/2019. 141
- [145] P. Mitra, A. Choudhury, V. R. Aparow, G. Kulandaivelu, and J. Dauwels, “Towards modeling of perception errors in autonomous vehicles,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 3024–3029, Nov 2018. 142
- [146] E. Vinkhuyzen and M. Cefkin, “Developing socially acceptable autonomous vehicles,” *Ethnographic Praxis in Industry Conference Proceedings*, vol. 2016, no. 1, pp. 522–534, 2016. 148, 151
- [147] G. de Campos, P. Falcone, and J. Sjöberg, “Autonomous cooperative driving: A velocity-based negotiation approach for intersection crossing,” in *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, pp. 1456–1461, Oct 2013. 148
- [148] T. Héroult, R. Lassaigne, F. Magniette, and S. Peyronnet, “Approximate probabilistic model checking,” in *Proceedings of the 5th International Conference on Verification, Model Checking, and Abstract Implementations*, vol. 2937 of *Lecture Notes in Computer Science*, pp. 73–84, Springer Berlin Heidelberg, 2004. 160
- [149] K. Sen, M. Viswanathan, and G. Agha, “On statistical model checking of stochastic systems,” in *Proceedings of the 17th International Conference on Computer Aided Verification* (K. Etessami and S. K. Rajamani, eds.), vol. 3576 of *Lecture Notes in Computer Science*, pp. 266–280, Springer Berlin Heidelberg, 2005. 160

BIBLIOGRAPHY

- [150] A. Pnueli, “The temporal logic of programs,” in *Proc. of the 18th Annual Symposium on Foundations of Computer Science*, pp. 46–57, IEEE Computer Society, 1977. 161
- [151] P. Zuliani, A. Platzer, and E. M. Clarke, “Bayesian statistical model checking with application to stateflow/simulink verification,” *Formal Methods in System Design*, 2013. 161
- [152] A. Armand, D. Filliat, and J. Ibañez-Guzmán, “A bayesian framework for preventive assistance at road intersections,” in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1128–1134, June 2016. 168
- [153] P. Vasishta, D. Vaufreydaz, and A. Spalanzani, “Building prior knowledge: A markov based pedestrian prediction model using urban environmental data,” in *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, pp. 247–253, Nov 2018. 169
- [154] B. Volz, H. Mielenz, I. Gilitschenski, R. Siegwart, and J. Nieto, “Inferring pedestrian motions at urban crosswalks,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, pp. 544–555, Feb 2019. 169
- [155] C. Hubmann, N. Quetschlich, J. Schulz, J. Bernhard, D. Althoff, and C. Stiller, “A pomdp maneuver planner for occlusions in urban scenarios,” in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2172–2179, IEEE, 2019. 170
- [156] I. Blečić, A. Cecchini, and G. A. Trunfio, “A decision support tool coupling a causal model and a multi-objective genetic algorithm,” *Applied Intelligence*, vol. 26, pp. 125–137, Apr 2007. 170
- [157] S. J. Pan and Q. Yang, “A survey on transfer learning,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, pp. 1345–1359, Oct 2010. 170
- [158] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16, 2017. 170