Reliable and safe control Navigation for Autonomous Vehicles in Dynamic Urban Environments

Charles Philippe

UNIVERSITÉ CLERMONT AUVERGNE
ÉCOLE DOCTORALE SCIENCES POUR L'INGÉNIEUR
CLERMONT-FERRAND

Thèse

Présentée par

CHARLES PHILIPPE
Ingénieur CentraleSupélec

pour obtenir le grade de

Docteur d'Université

Spécialité: **Electronique et Systèmes**

RELIABLE AND SAFE CONTROL AND NAVIGATION FOR AUTONOMOUS VEHICLES IN DYNAMIC URBAN ENVIRONMENTS

Soutenue publiquement le 29 Juin 2020 devant le Jury composé de:

Philippe BONNIFAIT	Examinateur	Professeur Heudiasyc - UTC
Thierry-Marie GUERRA	Rapporteur	Professeur LAMIH-UPHF
Ivan PETRUNIN	Rapporteur	Professeur, Cranfield University (UK)
Lydie NOUVELIERE	Rapporteur	MCF-HDR, IBISC - UEVE
Lounis ADOUANE	Directeur de thèse	Professeur Heudiasyc - UTC, Associé à l'Institut Pascal/UCA
Antonios TSOURDOS	Directeur de thèse	Professeur, Cranfield University (UK)
Benoît THUILOT	Encadrant	MCF, Institut Pascal - UCA
Hyo-Sang SHIN	Encadrant	Professeur, Cranfield University (UK)

Abstract

In this thesis is presented an algorithmic architecture for systematic risk evaluation, mitigation and management intended for autonomous transportation vehicles. The methods presented span low level control, trajectory tracking and multi-vehicle coordination. A task separation between low level steering control and trajectory tracking has been implemented to spread the design effort across two functional blocks. A robust low level controller has been designed, and a comfortable and flexible Model Predictive Controller (MPC) has been implemented for trajectory tracking. This controller has been associated with a supervision mechanism that monitors its performance in real time to evaluate the probability to underperform. When such a risk is identified, the speed of the system is adapted. The multi-vehicle coordination block fulfils the planning task. It is a decentralized, probabilistic optimization algorithm that is naturally risk-adverse. It is based on the Probability Collectives (PC) algorithm and operates a multi-stage negociation between vehicles. It has been made compatible with mixed-traffic scenarios with human drivers on the road. Results show that risks are monitored and managed across the whole architecture. Furthermore, easy to understand risk metrics are outputted to make the algorithms decisions understandable by the users and engineers working on the system. The work in this thesis thus proposes systematic risk management techniques transposable to all autonomous vehicles systems. It has been tested in simulations and on the autonomous test vehicles available at the Institut Pascal.

Abstract

Dans cette thèse est présentée une architecture algorithmique pour l'évaluation, le management et la minimisation du risque pour les véhicules de transport autonomes. Les travaux présentés couvrent le contrôle bas niveau, le suivi de trajectoire et la coordination multi-véhicules. Un contrôleur de direction robuste a été conçu, fonctionnant en synergie avec un module de suivi de trajectoire MPC (Model Predictive Control). Cela permet d'assurer l'optimalité du suivi, des garanties de confort ainsi qu'une supervision de la performance de suivi permettant de prévenir des écarts trop importants à la trajectoire de référence. Un algorithme décentralisé de coordination multi-véhicules a été conçu afin de remplir la tâche de planification. Il est décentralisé, fonctionne sur des hypothèses probabilistes et a pour conséquence un comportement naturel de minimisation du risque des manoeuvres. L'algorithme présenté est basé sur le "Probability Collectives Algorithm" (PC Algorithm) et opère une négociation entre les véhicules. Cet algorithme a été rendu compatible avec du trafic mixte comprenant des véhicules autonomes et des conducteurs humains. Des indicateurs liés au niveau de risque de la solution sont également calculés par l'algorithme. Ils permettent de rendre les solutions calculées facilement compréhensible par les décideurs et développeurs qui travailleraient sur un tel système. Le travail proposé dans cette thèse propose donc un management du risque systématique sur toute l'architecture typique d'un véhicule autonome, du contrôle bas-niveau à la coordination multi-véhicules. Les solutions proposées ont été testées en simulations et sur des véhicules autonomes de test disponibles à l'Institut Pascal.

Contents

Table of Contents **2**

List of Figures **6**

List of Tables **10**

1 General Introduction and Thesis Overview **12**

1.1 Context of the thesis 14

1.2 Research gap 16

1.2.1 Autonomous vehicles background 16

1.2.2 Navigation of autonomous vehicles 17

1.2.3 Multi-Vehicle navigation 19

1.3 Objectives 23

1.3.1 Control and trajectory tracking 24

1.3.2 Multi-Vehicle Coordination (MVC) 25

1.3.3 Path to achieve the aims 27

1.4 Manuscript organization 29

1.5 List of published and submitted works 30

2 Risk and Comfort Management for Multi-Vehicle Navigation using a Flexible and Robust Cascade Control Architecture **41**

2.1 Introduction 44

2.2 Towards a Flexible And Robust Control Architecture 45

2.2.1 Related works 45

2.2.2 Architecture design 45

2.3 Proposed Cascade Control Architecture 46

2.3.1 Inner loop design 47

2.3.2 Outer loop design 49

2.3.3 Conclusion . . . 53
2.4 Simulations and Results . . . 53
2.4.1 Model Used . . . 53
2.4.2 Simulation scenarios . . . 53
2.4.3 Behaviour comparison . . . 56
2.4.4 Safety and comfort assessment . . . 57
2.5 Conclusion . . . 59

3 Safe and Online MPC for Managing Safety and Comfort of Autonomous Vehicles in Urban Environment 63
3.1 Introduction . . . 66
3.2 Proposed Architecture . . . 67
3.2.1 Architecture design . . . 67
3.2.2 MPC controller . . . 68
3.2.3 MPC behaviour monitoring . . . 74
3.3 Experiments . . . 76
3.3.1 MPC behaviour analysis . . . 76
3.3.2 Risk monitoring . . . 79
3.4 Conclusion . . . 81

4 Probability Collectives Algorithm applied to Decentralized Intersection Coordination for Connected Autonomous Vehicles 84
4.1 Introduction . . . 87
4.2 Proposed intersection coordination algorithm . . . 88
4.2.1 Probability Collectives algorithm . . . 88
4.2.2 Application to Intersection Coordination . . . 90
4.2.2.1 Problem formulation . . . 91
4.2.2.2 Objective Function . . . 93
4.2.3 Comparison with existing work using PC . . . 94
4.3 Experiments . . . 95
4.3.1 Proof of concept . . . 95
4.3.2 Analysis of consistency on fixed initial situation . . . 98
4.3.3 Influence of search space sampling . . . 99
4.4 Conclusion . . . 100

5 Robust and Decentralized Traffic Flow Management of Autonomous Vehicles at Intersections based on Probability Collectives Algorithm 104

5.1 Introduction . . . 107
5.1.1 Related works . . . 107
5.1.2 Organization of the chapter . . . 108
5.2 Proposed Intersection Coordination Algorithm . . . 109
5.3 Algorithm Characterization . . . 112
5.3.1 Effect of sample size for cost expectancy computation . . . 115
5.3.2 Scalability properties . . . 116
5.3.3 Intersection layout effect . . . 118
5.4 Towards Real-Life Implementation . . . 119
5.4.1 Ensuring smooth speed profiles for comfortable use . . . 119
5.4.2 Optimization with non-collaborative connected agents . . . 122
5.4.3 Adaptation to continuous flow at an intersection . . . 124
5.4.4 Demonstration of continuous traffic capabilities . . . 126
5.5 Conclusion . . . 127

6 Unified Probabilistic Multi-Vehicle Coordination (UP-MVC): Extending the P-MVC for Real-Life Application and Mixed-Traffic Scenarios 132
6.1 Introduction . . . 135
6.2 UP-MVC: Extension of the Algorithm to Mixed-Traffic Scenarios . . . 136
6.2.1 Computational aspects of dealing with non-collaborative, non-connected agents . . . 136
6.2.2 Possible action prediction for human-driven vehicles . . . 137
6.2.3 Criteria for comparing solutions . . . 139
6.2.4 Probability of disruption . . . 143
6.3 UP-MVC Demonstrations and Experiments . . . 144
6.3.1 UP-MVC for mixed traffic: proof of concept . . . 144
6.3.2 Study of probability distribution standard deviation . . . 148
6.3.3 Insertion of human-driven vehicles in online continuous flow simulations . . . 149
6.4 Conclusion . . . 155

7 General Conclusion and Perspectives 157
7.1 Conclusion . . . 159
7.2 Further work . . . 160
7.2.1 Vehicle Planning and Control . . . 160
7.2.2 Multi-Vehicle Coordination . . . 161

List of Figures

1.1 PAVIN test track in Clermont-Ferrand (France) 15
1.2 IPCar test vehicles 15
1.3 Schematic illustration of the targeted research gap 20
1.4 Different functional blocks addressed in the thesis 23
1.5 Example of a targeted use-case for the coordination algorithm 26
1.6 Illustration of the 4DV simulator environment 28
1.7 Validation workflow for control and trajectory tracking contributions . . . 28
1.8 Validation workflow for Multi-Vehicle Coordination (MVC) algorithm development 29

2.1 IPCar autonomous transport vehicles in a coordinated maneuver 44
2.2 Designed cascade architecture 46
2.3 Dynamic bicycle model conventions 47
2.4 Closed-loop step response envelope 49
2.5 Partial availability of a reference trajectory in leader/follower navigation . 50
2.6 Navigation errors definition 51
2.7 Example of path for the simulations (target and follower) 55
2.8 Tracking performance (1^{st} series) 56
2.9 Control signals (1^{st} series) 56
2.10 Comfort indicators (1^{st} series) 57
2.11 Tracking performance (2^{nd} series) 58
2.12 Control signals (1^{st} series) 58
2.13 Comfort indicators (2^{nd} series) 59

3.1 Planning and risk monitoring architecture 68
3.2 Model conventions 69
3.3 4D-Virtualiz simulation environment 77
3.4 Reference trajectory for MPC coefficients assessment 78

3.5 Effect of $R_{d,11}$ coefficient, smooth reference trajectory 78
3.6 Effect of $R_{d,11}$ coefficient, noisy reference trajectory 79
3.7 Lateral tracking risk prediction and speed replanning 80

4.1 Example of intersection configuration and paths of the vehicles 91
4.2 Search space representation with two-step optimization 92
4.3 Snapshot of the application of the best solution when $t = 4s$. The purple vehicle is entering from the right lane and will exit at the bottom. It slowed down just enough to yield for the yellow vehicle (yellow vehicle came from the right, exits at the top). The red vehicle (coming from the left, exits to the right) accelerated up to its maximal allowed speed so that the blue one has to wait the minimal amount of time before entering the intersection (blue enters from the bottom). Full video available at `https://youtu.be/XTgY-4RUFz0` 97
4.4 Results of 100 runs of the PC optimization for the same initial conditions. Mode 1 in blue and Mode 2 in orange. 99
4.5 Effect of the size of the search space on the performance of the coordination algorithm. 100

5.1 Comparison of the layout of a traditional crossroad and a roundabout. Matched upstream roads and initial position of vehicles. 114
5.2 Search space representation with two-step optimization 115
5.3 Effect of sample set size for cost expectancy computation 116
5.4 Scalability study of the PC algorithm with variable number of vehicles involved in the optimization 117
5.5 Results of Monte-Carlo simulation with 8 vehicles in a 4-way intersection versus a roundabout (matched initial positions) 118
5.6 Generated smooth speed profiles 122
5.7 Snapshot of the collaborative solution accounting for non-collaborative connected agent in blue, taken at $4.7s$. Link to video at `https://youtu.be/bpxXq9G4RTk` 123
5.8 Illustration of intersection layout for continuous traffic coordination . . . 125
5.9 Snapshot of the continuous traffic simulation. Link to video at `https://youtu.be/9k_J8E0x_-M` 128

6.1 Example of strategies hypotheses and discrete probability distribution . . 138
6.2 Possible strategies for V_1 and associated probabilities 145

6.3 Snapshot of the application of the best solution when $t = 4.3s$ 146
6.4 Snapshot of the application of the best solution when $t = 5s$ (high certainty) 147
6.5 Relation between standard deviation of behaviour estimation and space given to human drivers. 149
6.6 Snapshot of a continuous traffic flow simulation with 20% of human drivers 152
6.7 Speed profiles of some of the autonomous vehicles 153
6.8 Repartition of intersection crossing times 154

List of Tables

2.1 Parameter values for uncertain car plant study 48
2.2 MPC controller parameters 52
2.3 Common simulation parameters 54
2.4 Variable simulation parameters 55

3.1 Parameter values for MPC behaviour analysis 78

4.1 Main PC parameters 97
4.2 Results of the PC consistency test (100 runs) 98

5.1 Monte-Carlo simulations parameters 113
5.2 Main PC parameters 113
5.3 Continuous traffic simulation parameters 127

6.1 Main parameters for UP-MVC proof of concept 144
6.2 UP-MVC parameters for continuous traffic simulation with human drivers 150

Chapter 1

General Introduction and Thesis Overview

1.1 Context of the thesis

The PhD thesis presented in this manuscript is a joint effort by Cranfield University (UK) and the Université Clermont Auvergne (UCA, France). Each university has participated for half of the funding of the thesis. This collaboration stems from Pr. Antonios Tsourdos from Cranfield University and Mr. Lounis Adouane from the UCA.

Cranfield University is a UK based university founded in 1946. It originally served to teach test pilots, and later diversified in Aeronautics related engineering. It has become a strong actor in Unmanned Aerial Vehicles (UAV) -and Unmanned vehicles in general- in the last two decades. It has now extended its interests to ground vehicles for passenger transportation. The work at Cranfield University takes place in the *School of Aerospace, Transport and Manufacturing* under the supervision of Pr. Antonios Tsourdos.

The UCA is a generalist French university based in the city of Clermont-Ferrand in the Auvergne-Rhône-Alpes region. The UCA hosts a number of laboratories ranging from geology to particle physics. This thesis has been done as part of the *Institut Pascal* (IP) laboratory `http://www.institutpascal.uca.fr/index.php/fr/`. It is a multi-disciplinary laboratory focused on engineering sciences. The IP's mission is to help develop the technologies for next generation transports, hospitals and factories. The *Institut Pascal* has several team clusters. The author has worked under the *Images, Systèmes de Perception, Robotique* (Visual Sensing, Perception Systems, Robotics) cluster.

Cranfield University and the UCA entered a collaboration for a joint project called *Integrated Risk Management Architecture for Autonomous Vehicle Navigation* (IRMA-AVN) project, led by Pr. Lounis Adouane in the context of Laboratory of Excellence IMobS3 (Innovative Mobility: Smart and Sustainable Solutions). The aim of this project is to improve the flexibility and safety of autonomous vehicles performing complex navigation tasks in urban environments. Three PhD theses have been funded within this project. Two of them are related to sensing and collaborative sensing. This thesis is the third one and is focused on the planning and control of the vehicles.

The Institut Pascal has a strong background in robotics applied to ground vehicles and passenger transportation vehicles [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]. These works span robotics for agricultural applications, Lyapunov based navigation of autonomous vehicles, platooning, and bayesian reasoning for decision making.

The Institut Pascal has a test track for autonomous vehicles called the *Plateforme d'Auvergne pour les Véhicules Intelligents* (PAVIN, Auvergne Platform for Intelligent Vehicles). It is

an approximately 1600sqm zone contaning roads with all the hallmarks of urban environments: a roundabout, traffic lights, narrow roads, blind corners and pedestrian crossings. The track is narrower than a regular road, and the vehicles are scaled accordingly for testing. The PAVIN is shown in Figure 1.1.

Figure 1.1: PAVIN test track in Clermont-Ferrand (France)

The test vehicles used for this thesis are the IPCars from the Institut Pascal. The IPCars are electric test vehicles with 4 seats, with approximately the size of a golf cart. They are equipped for autonomous navigation with steering actuators and a range of sensors: LIDAR, GPS, Real-Time Kinematics GPS (RTK-GPS), cameras and an Inertial Measurement Unit (IMU). Their maximal speed is $3m/s$, which mimicks regular speeds and reaction times in a real urban environment due to the dimensions of the test track. IPCar test vehicles are shown on Figure 1.2.

Figure 1.2: IPCar test vehicles

1.2 Research gap

The aim of this thesis is to address several topics spanning the whole architecture of an autonomous vehicle related to decision and action. The following sections replace the topic in its context and the state of the art.

1.2.1 Autonomous vehicles background

Autonomous Vehicles (AV) and Connected Autonomous Vehicles (CAVs) are seen as a great tool to improve safety on the roads and reduce congestion in and out of urban areas [12]. CAVs in particular would be able to mitigate traffic jams due to speed oscillations on highways.

There have been several big challenges in the last decade in order to push the developments and adoption of autonomous vehicles:

- The *Defense Advanced Research Projects Agency* (DARPA) Grand Challenge (2005): [13, 14, 15]
- The DARPA Urban Challenge (2007): [16, 17, 18, 19, 20, 21]
- The Grand Cooperative Driving Challenge (GCDC) [22, 23, 24]

Nowadays, developments in autonomous vehicles are not only focused on the vehicles themselves but also their integration in urban traffic infrastructure and their influence on traffic, whether at the scale of an single intersection or a city.

Private actors like Tesla, Google and historic car manufacturers such as Renault, BMW and Volvo have also been getting involved in autonomous vehicles research, highlighting the interest for the field. However, the adoption of autonomous vehicles is still slow [25]. There are still technical challenges, legal challenges and also public acceptance challenges [26].

The work in this thesis aims to propose methods that would accelerate the adoption of autonomous vehicles on the short and medium term. Those methods include risk management technique that would help with the legal and public acceptance challenges, which are often foreshadowed by the purely technical challenges. Of course, the algorithms developed must at least attain, if not surpass, the performance of current algorithms for autonomous vehicles. The following sections highlight the state of the art for autonomous navigation and multi-vehicle coordinations. They also point to where contributions can be

made with respect to risk evaluation and management of the algorithms implemented in autonomous vehicles. Some researchers have started going in this direction, with a focus on minimizing risk during navigation but for single vehile navigation only.

1.2.2 Navigation of autonomous vehicles

In the field of ground vehicles, the term "navigation" entails all actions related to reaching a goal (defined by a configuration of the vehicle in the environment). The distinction between low level control and trajectory tracking is usually not made, as those functions are most often fulfilled by a single functional block.

For example, such integrated approaches are presented in [15, 27]. Such integrated approaches usually resort to nonlinear control techniques or optimal techniques such as Model Predictive Control.

The work presented in [28] fits in this category but adds flexibility to the algorithm: The goal of the algorithm is defined as a target which is either static or mobile. This kind of flexibility is targeted in this thesis. However, these approaches usually do not implement safety metrics/constraints or robustness because they already have to fulfill a lot of tasks.

Some other works cover the two tasks but with separate controllers for each one, which has the advantage to separate the objectives for each task and give an adapted answer. These approaches usually use two controllers in a cascade architecture. Such works for ground vehicles include [29] and [30]. Linear control is often used for the inner control loop. A similar cascade architecture for aerospace applications is presented in [31]. It is a common approach in Unmanned Aerial Vehicles (UAV) design.

There is a desire in this thesis to implement safety guarantees and safety constraints. In the literature, this naturally leads to the field of robust control. Robust control is a general term defining control techniques that will be largely unaffected by perturbations from the environment or in the behaviour of the vehicle. For example, a vehicle steering controller could be desired to be robust to slipping, or gusts of wind (for trucks mostly). Robust controllers offer a form of safety at the expense of performance: robust controllers are more conservative to prevent oscillations and instabilities of the system.

Several types of robustness can be distinguished:

- Robustness to external perturbations like wind, slippery surface (perturbation rejec-

tion).

- Robustness to model uncertainty. This is useful when the exact model describing the behaviour of a system cannot be known.
- Robustness to an actuator malfunction (special kind of perturbation rejection). This characteristic is often sought after in aircraft control, where an engine could stop and destabilize the aircraft without proper design.
- *High-level robustness*. This form of robustness is about having an external algorithm that oversees the controller's performance and takes appropriate action if the performance is not within acceptable bounds. This form of robustness is not directly linked to the control aspect and is quite recent in the literature. A possible example would be limiting the speed of a vehicle to ensure the performance of a lateral tracking controller (assuming the lateral control is more precise at low speeds).

A corpus of works have been dealing with robust controller synthesis on linear models. This is the now mainstream robust control theory. Usually a H_∞ controller is designed to control the uncertain model [32, 33], H_∞ being a form of optimally designed controller that can take robustness criteria as a design input. Very rarely, some robustness proofs have been achieved with nonlinear (Lyapunov) based control [34].

When the controller depends on a measurable parameter, Linear Parameter Varying (LPV) models can be used. In this case the varying parameter is also used as a parameter for the controller, for example with gain-scheduling techniques [35, 36, 37]. An example is provided in [38] with a gain-scheduled robust controller for the steerability and lateral stability of a vehicle designed on an LPV model.

Besides robustness considerations, optimality of the control actions is often a big consideration. Optimal control describes a set of techniques where the control signal is computed as the result of an optimization procedure. Model Predictive Control (MPC) is the most popular of these techniques in the recent literature. It consists of formulating an optimization problem to find the best control sequence of the vehicle within a fixed time horizon in the future. It uses an internal model to compute what is best with respect to the objective. MPC can have different forms, such as nonlinear [39] and linear [40, 41, 42]. The linear form of MPC is a bit more constrained in its formulation and relies on more assumptions, but it does provide the convergence proof and low computational cost that the nonlinear version fails to deliver. Most MPC implementations for trajectory tracking are done this way [40, 41, 42]. Other refinements of MPC exist such as robust MPC [43],

constrained MPC [44] or linear MPC applied on time-varying (LTV) models [45].

This thesis aims to steer away from integrated approaches in order to spread the design effort to more functional blocks. Robust control is a popular technique but has been seldom used for the control of ground vehicles: so far the preference in the literature has been for integrated and nonlinear methods. In addition to that, this thesis aims to bring new safety metrics and risk monitoring to the field of urban vehicles. As the goal is urban navigation with multi-vehicle maneuvers, a degree of flexibility should be kept in the algorithm such as in [6]. The work presented here thus aims to fill the gap between non-flexible algorithms with supervision mechanisms [46], robust control techniques [33], and flexible navigation techniques for urban use [6].

1.2.3 Multi-Vehicle navigation

The aim is to bridge the gap between single autonomous vehicle applications and multi-robot algorithms applied to small robots (cf. Figure 1.3). In general, multi-robot coordination fails to propose systematic risk management or safety proofs, which is why few real-world applications have been seen to this day. The existing projects implementing multi-vehicle coordination focus on:

- Infrastructure-heavy centralized solutions, assuming all vehicles share the same standards and all vehicles on the road are autonomous.
- Simple interactions between vehicles, such as a mutual exclusion from an intersection or navigation in a fixed formation.

Previous works at the Institut Pascal have implemented and demonstrated more complex interactions [6, 11] in recent years. This thesis follows this research direction and aims to build upon these works by implementing risk monitoring and management techniques for multi-vehicle maneuvers.

Among the possible use-cases target for multi-vehicle interaction, one could mention:

- Merging (insertion on a highway)
- Overtaking
- Intersection crossing
- Platooning

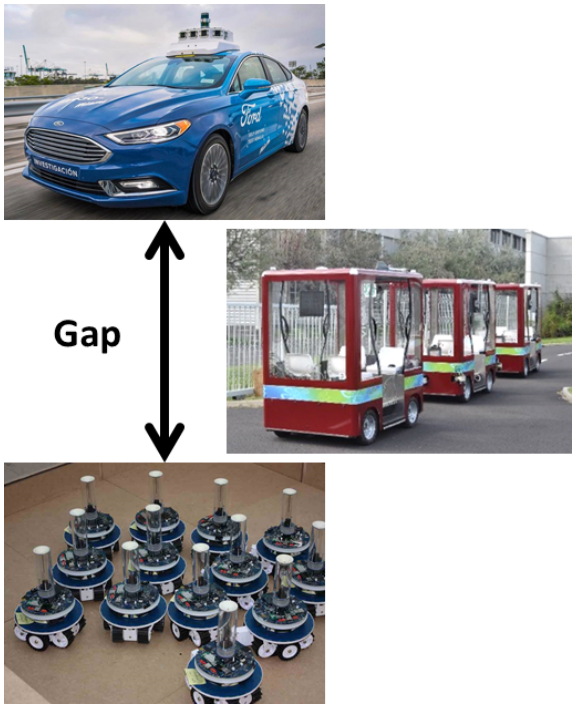

Figure 1.3: Schematic illustration of the targeted research gap

Platooning maneuvers include maneuvers realized by a group of connected vehicles that travel on the same road. Usually, the goal is to maintain a desired formation shape. This can be for example about maintaining an equal longitudinal distance between several vehicles in the same lane [47, 48, 49]. Platooning is thought to bring benefits in terms of traffic throughput even though it is a very local action [50]. Platooning has been extensively studied at the Institut Pascal [1, 6]. It is also used as an additional tool to improve the throughput at intersections [51, 52]. The main risk associated with platooning is to enure that any oscillations between the relative positions of the vehicles are dampened. This is the concept of *string stability* [53].

Merging is a maneuver that could be considered specific because of the constraints: the maneuver has to succeed. The objectives of a merging maneuver are to reduce travel time, to prevent oscillations that cause traffic jams, and ensure safe completion of the maneuver. A study of the impact of Connected and Automated Vehicles at merging roadways has been proposed in [54]. Several dedicated optimal methods have been proposed in the literature [55, 56, 57, 58].

Overtaking maneuvers have also been a focus of the community recently, in order to increase the capabilities of current *Adaptive Cruise Control* systems beyond following a vehicle in a single lane [59, 10, 11, 60]. Some works have started to propose probabilistic

frameworks for overtaking maneuvers [61, 11].

Intersection management is an extension of platooning on vehicles that do not travel on the same road/in the same direction. The aim is usually to maximize the throughput of the intersection under some constraints of connectivity and safety. The methods can be similar to the two previous topics. Due to the number of vehicles involved, centralized approaches are sometimes preferred.

Intersection management is an important goal for many research groups as well as city planners. Intelligent traffic lights are currently being rolled out [62] in an attempt to curb congestion with adaptive traffic lights cycles. Connected Autonomous vehicles (CAV) are seen as a valuable tool to further improve the traffic flow in congested areas through efficient intersection crossing. A study of the possibilities of reducing the overall energy expenditure at signalized intersections is presented in [63]. Even relatively simple techniques such as platooning could double throughput at intersections [50]. The main objective seeked is usually a maximization of throughput/minimization of waiting time. These objectives are similar, even though throughput maximization more refers to a wider scale. Waiting time minimization takes more the point of view of a passenger at a single intersection. From the point of view of a city planner, pollution minimization would also be an important objective, hence some works on minimization of energy expenditure at intersections [64]. As most of the algorithms rely on optimization techniques, the optimization criterion could easily be focused on one or the other. It means that these methods are not mutually exclusive for the most part. Reviews of planning at intersections are proposed in [65] and [66].

A first main group of techniques can be characterized by their hypothesis that 100% of vehicles on the road are autonomous and connected. The majority of the research done on intersection management nowadays works under this hypothesis. It could become true over the long term, but only after a transition phase where autonomous vehicles and human-driven vehicles would coexist on the road. This hypothesis also implies that there will be universal standards of communication and procedures between autonomous vehicles (*V2V*, Vehicle-to-Vehicle communication) and/or with a centralized infrastructure (*V2I*, Vehicle-to-Infrastructure communication).

Among this first group of techniques, most of them directly assign a trajectory to vehicles [67, 68, 60, 69, 70, 71, 72]. Vehicles are often controlled individually and the intersection is assumed to be non-signalized. In addition to coordination mechanisms, platoon formation is sometimes used to alleviate the computing costs while having little

impact on the overall performance [73, 52] (platoon formation can also be used on its own with regular intersections. This will be detailed later). In particular, [64] proposes an energy minimization mechanism. These algorithms usually consider that the intersection is unsignalized, and thus no rules are imposed on the vehicles actions. This is the very property that excludes the sharing of the road with human drivers.

Some others work under a more restrictive hypothesis of mutual exclusion from a shared zone [74, 75, 76, 77, 78]. These techniques lead to less optimal solutions because of this strict mutual exclusion condition. However it means that human drivers could in theory use an intersection managed by such an algorithm as well.

The method in [79] proposes an interesting variant of the mutual exclusion class of algorithms by defining several small exclusion zones where the respective paths of vehicles cross each other (instead of the whole intersection). The solution is formulated as an ensemble of permutations which is the order of the vehicles at each conflicting point.

Still within this first category, some methods leave the door open for sharing the road with humans. In this class of algorithms, the intersection is signalized so that human drivers know what behaviour to follow. For example, [80], [81], [82] and [83] propose to control both the trajectory of autonomous vehicles and the traffic lights patterns. This is a way of making a signalized intersection more flexible for autonomous vehicles while still being compatible with human drivers. The main drawback of these systems are that they restrict the degree of freedom available to autonomous vehicles: they have to conform to what the traffic light says. However, this is a logical consequence of leaving the door open to accommodating human drivers. A similar method that only acts on the speed of vehicles is presented in [84], using Cooperative Adapative Cruise Control (CACC) with a heuristic optimization algorithm to reduce delays at an intersection. CACC is a driver assistance feature that could be also found on human driven vehicles (or semi-autonomous). To be collaborative, such vehicles would still need to be connected.

The second group of techniques includes those which explicitly consider a mixed traffic in their design. To the knowledge of the author, all these methods rely on traffic light management in addition of assigning trajectories to the autonomous vehicles. The method proposed in [85] explicitly considers three types of vehicles: conventional, connected non-automated, and automated. This application relies on connected vehicles to gather information on the incoming flow rates at an intersection and optimize the traffic lights pattern accordingly. The traffic light based method proposed in [86] fixes the dependency to connected vehicles by using external sensors. It has been successfully tested with

various market penetration rates of autonomous vehicles. A similar testing protocol has been used in [66] with human driven vehicles in the simulations. It has been shown that even a small proportion of Autonomous Vehicles can improve the smoothness of the traffic flow and increase throughput of the intersection.

Besides the traffic light management, some approaches focus on enabling platoons of autonomous vehicles at normal intersections to improve throughput [87, 88]. Arguably, these methods are not really intersection management methods as they make do with existing intersection mechanisms.

There is currently a gap in the literature for intersection management methods that:

- Explicitly consider human driven vehicles.
- Work on existing infrastructure (i.e., without using intelligent traffic lights).
- Are able to provide risk metrics and risk management techniques.

Regarding the last point, probabilistic techniques that allow to manage the level of risk have so far been reserved for single vehicle applications [61, 89, 90].

1.3 Objectives

An outline of the algorithmic architecture developed during this thesis is shown in Figure 1.4. The lower part of this architecture is typical for ground vehicles and autonomous vehicles navigation in general. The *Risk Management* block has been added because it is one of the contributions of the thesis.

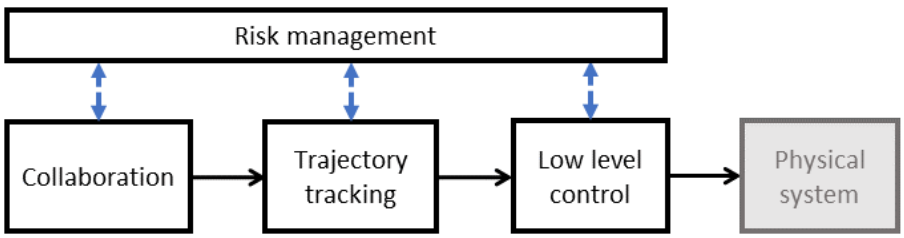

Figure 1.4: Different functional blocks addressed in the thesis

The general aims of the thesis can be categorized either by theme or by the functional block that is concerned.

The aims categorized by topics are:

- Using robust methods across the architecture
- Implementing systematic risk monitoring metrics
- Implementing risk management/mitigation techniques using the monitoring metrics
- Using optimal algorithms when possible

These objectives will be applied across the whole algorithmic architecture of autonomous vehicles. More precisely, they will be applied to:

- The low-level control block
- The Trajectory tracking block
- The Collaborative planning block
- The Risk monitoring block created for the needs of this thesis.

Overall, the expected result of this thesis is a coherent architecture that provides the functions necessary for the navigation of autonomous vehicles in urban environments. The vehicles should be able to cooperate together as a result of the developed algorithms. The risk management techniques should be as independant as possible from the specific implementation of each function, such that they are generalizable to a whole range of specific navigation techniques across different works.

The following sections will detail those aims following the functional block classification. The objectives concerning the risk monitoring function will be addressed in the sections for the other blocks for clarity.

1.3.1 Control and trajectory tracking

There are several targeted aims for the control and trajectory tracking architecture:

- It should provide flexibility with regards to the tasks performed: trajectory tracking, target following, navigation in platoon
- It should be explicitly comfortable and safe, in a way that is easy to validate
- It should provide robustness to perturbations and model uncertainties: these characteristics are often absent from the works in passenger vehicle navigation.
- It should provide risk monitoring metrics and the ability to perform risk management/mitigation.

These characteristics have been seen in different works but not at the same time. Approaches that are flexible are usually performance based and do not provide risk metrics. Likewise, Robust approaches often fail to provide flexibility.

The proposed combination of characteristics is seen as satisfying the constraints for higher adoption of autonomous vehicles technology for urban transportation.

A risk monitoring function should be clearly identified and produce meaningful metrics understandable by humans. In fact, the risk monitoring and mitigation shall be dissociated as much as possible from the actual implementation of the algorithms. This is to be able to provide systematic risk management methods that are transposable to different algorithmic architectures.

The resulting algorithms should provide all three characteristics at the same time. Success will also be evaluated in light of the possibility of implementing the algorithms and control techniques in real-life on the test vehicles. In fact, an objective of the thesis is to test the control algorithms on the IPCar vehicles under the ROS framework. As this framework is becoming more standard in the autonomous vehicles industry, this would be another big argument towards the applicability of the work developed in this thesis to real life situations.

1.3.2 Multi-Vehicle Coordination (MVC)

The development of Multi-Vehicle Coordination (MVC) techniques in this thesis has been focused on intersection management. There is a research gap in the literature for dealing with short and medium term scenarios (mixed traffic on the road), and even more so without using dedicated and expensive infrastructure. Furthermore, even fewer methods propose systematic risk management as is targeted in this thesis.

Other multi-vehicle maneuvers can be seen as particular cases of intersection management. An intersection is in essence the least structured space the vehicles have to face. There are vehicles coming from every direction, multiple destination options and in general no clear structure to the maneuver that should be achieved. By contrast, the vehicles all go in the same direction for merging, platooning and overtaking. The number of potential conflicts and the differential speeds between vehicles are much smaller in these cases.

The goal is thus to develop a general algorithm for multi-vehicles collaboration that will

be tested on intersection management in the most difficult conditions. These difficult conditions will entail high densities of vehicles, several intersection layouts and the inclusion of human drivers with unknown intentions. It is expected that an algorithm that can cope with intersection management under those conditions can also cope with merging and overtaking.

A typical intersection use-case to challenge the design is presented in Figure 1.5. Its main features are the presence of a human driver with unknown intentions and the potential use of platooning as an additional tool to reduce complexity of the maneuver. This type of scenario could happen in the short to medium term because of the presence of humans on the road. Some algorithms work under the longer term hypothesis of 100% of CAV on the road. It is an explicit aim of this thesis to focus on shorter term scenarios by keeping perturbations in the form of human drivers on the road.

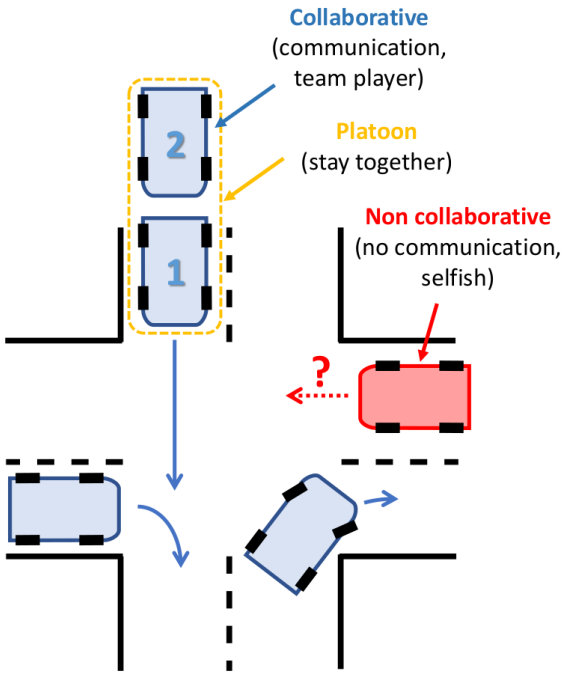

Figure 1.5: Example of a targeted use-case for the coordination algorithm

Success will be evaluated in light of the actual capabilities of the algorithm: optimality, scalability, real-life applicability and risk management abilities. Scalability of the algorithm means that it could work with arbitrarily complex intersection layouts and high numbers of vehicles without explosion of the computational cost. Risk management abilities are linked to the fact that the algorithm can output relevant risk metrics and has a risk-adverse behaviour.

Real-life applicability in the case of this work is composed of several factors:

- Ease of implementation: the algorithm is easily transferrable to real vehicles and does not need dedicated and expensive infrastructure.
- Low computational cost: The algorithm should not need supercomputers to work at a satisfactory speed.
- Ease of deployment: an algorithm is easy to deploy if it does not rely on overly strict software and data standards. It should also be easily understandable to engineers and developers that are not the designers of the algorithm.

1.3.3 Path to achieve the aims

The author has approached the problems in a bottom-up order. The architectural considerations have been tackled first (in Chapter 2) because they condition the rest of the work. Then the low level control has been tackled, followed by the trajectory tracking and the multi-vehicle coordination algorithm. The risk management techniques have been developed along with the three functional blocks mentioned before. Usually, a given risk monitoring function is closely related to one of the blocks.

In the scope of this thesis, three validation techniques are available and have all been used:

- Matlab simulations
- *4D-Virtualiz* (4DV) simulation software
- Implementation on test vehicles

The 4D-Virtualiz software `http://www.4d-virtualiz.com/en/` is an advanced simulation environment for the test and validation of robotic systems. Figure 1.6 shows the environment in 4DV. The PAVIN and IPCars are simulated in the software as well as all the vehicle's sensors. It allows for a very realistic simulation from where it is easy to transfer code to real vehicles. The test vehicles are the IPCars described in Section 1.1.

The validation workflow intended for the control and trajectory tracking contributions is presented in Figure 1.7. All algorithms are prototyped and evaluated on Matlab first. A C++ implementation for the *Robot Operating System* (ROS) framework is then carried away. ROS is a popular middleware to implement software on robots. It provides a programmation environment to the user in which the sensors and actuators are easily

Figure 1.6: Illustration of the 4DV simulator environment

accessible (`https://www.ros.org/`). The ROS code can then be tested in the 4DV environment or on the real vehicles. Due to the high level of simulation accuracy provided by 4DV, the transfer from 4DV to the real vehicles is very easy and can be done back and forth.

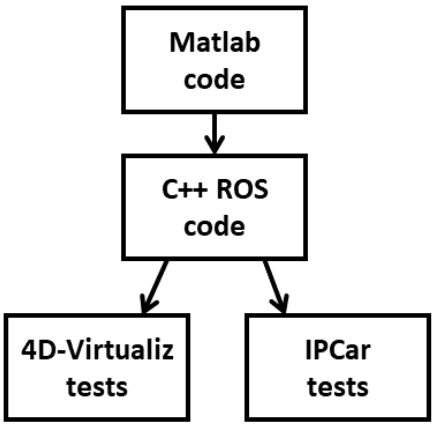

Figure 1.7: Validation workflow for control and trajectory tracking contributions

The intended workflow is slightly different for developing the Multi-Vehicle Coordination (MVC) algorithm. It is depicted in Figure 1.8. As most MVC techniques work on simpler hypotheses than is intended here (100% of CAV and no explicit risk management), the developments will start from there and try to build on the complexity of the scenarios that the algorithm can tackle. If mature enough, it will then be transferred to C++ code in order to be tested in simulation or on real vehicles.

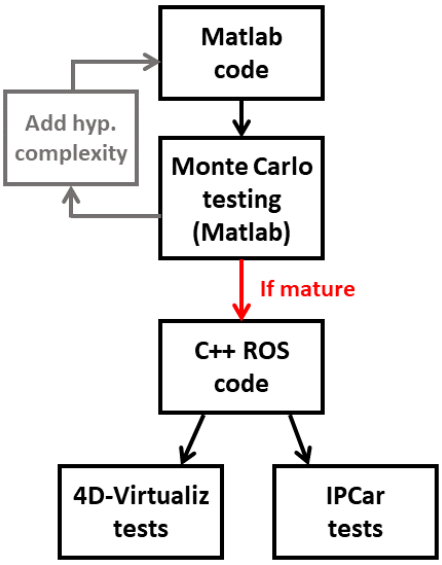

Figure 1.8: Validation workflow for Multi-Vehicle Coordination (MVC) algorithm development

1.4 Manuscript organization

The remainder of the manuscript is divided as follows:

- Chapter 2 investigates the necessity to split the navigation function between the *Low level control* block and the *Trajectory tracking* block. It also proposes a robust controller for the *Low level control*.
- Chapter 3 focuses on the *Trajectory tracking* block and its relation to the *Risk management* supervision block. As the trajectory tracking function is freed from low level considerations thanks to the split between low level control and trajectory tracking, a risk-based approach can be implemented without asking too much of this block. A risk management technique is also demonstrated to successfully enforce tracking performance.
- Chapter 4 focuses solely on the *Collaboration* block. It presents a decentralized multi-vehicle collaboration technique that aims to be compatible with a risk management module. This is achieved by developing an algorithm that works on probabilistic assumptions and can output risk-related metrics.
- Chapter 5 extends the algorithm proposed in Chapter 4 in order to make it useable in real-life situations. It is related mostly to the *Collaboration* block.
- Finally, Chapter 6 presents a further development of the collaboration technique

proposed, called the *Unified Probabilistic Multi-Vehicle Coordination* algorithm (UP-MVC). This algorithm is a polyvalent multi-vehicle planner able to cope with human drivers on the road. Due to its probabilistic nature, it naturally takes risk adverse decisions and outputs relevant, easy to understand risk metrics. Thus the developments in this chapter are related to the *Collaboration* block and the *Risk management block*.

- This thesis manuscript ends with general conclusions of the proposed contribution and several prospects for this PhD.

1.5 List of published and submitted works

The first three articles have been published in international conferences. They correspond respectively to the content in Chapter 2, 3 and 4:

- Philippe C., Adouane L., Tsourdos A., Shin H.-S., Thuilot B., "Risk and comfort management for multi-vehicle navigation using a flexible and robust cascade control architecture". In: *European Conference on Mobile Robots* (6-8 Sept. 2017, Paris). [91]

- Philippe C., Adouane L., Tsourdos A., Shin H.-S., Thuilot B., "Safe and Online MPC for Managing Safety and Comfort of Autonomous Vehicles in Urban Environment". In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC* (Nov. 4-7 2018, Maui, Hawaii), pp. 300-306. [92]

- Philippe C., Adouane L., Tsourdos A., Shin H.-S., Thuilot B., "Probability Collectives Algorithm applied to Decentralized Intersection Coordination for Connected Autonomous Vehicles". In *Intelligent Vehicles Symposium, IV* (9-12 June 2019, Paris), pp. 1928-1934. [93]

The following paper has been submitted to the journal *Transaction on Intelligent Vehicles*. It is currently pending for review. It corresponds to the content given in Chapter 5:

- Philippe C., Adouane L., Tsourdos A., Shin H.-S., Thuilot B., "Robust and Decentralized Traffic Flow Management of Autonomous Vehicles at Intersections based on Probability Collectives Algorithm". Submitted to *Transactions on Intelligent Vehicles, IV*.

Finally, the content of Chapter 6 has been submitted in a shortened form to the *International Conference on Robotics and Automation* (ICRA '20'):

- Philippe C., Adouane L., Tsourdos A., Shin H.-S., Thuilot B., "Unified Probabilistic Multi-Vehicle Coordination (UP-MVC) Algorithm: A Decentralized Optimal Intersection Crossing Algorithm Compatible with Mixed-Traffic Scenarios". Submitted to *International Conference on Robotics and Automation, ICRA*.

References

[1] José Miguel Vilca, Marco H. Terra, and Valdir Grassi. "Formation Control with Leadership Alternation for Obstacle Avoidance". In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 1090–1095.

[2] José Miguel Vilca, Lounis Adouane, and Youcef Mezouar. "An Overall Control Strategy Based on Target Reaching for the Navigation of an Urban Electric Vehicle". In: *International Conference on Intelligent Robots and Systems (IROS)* (2013), pp. 728–734.

[3] Ahmed Benzerrouk, Lounis Adouane, and Philippe Martinet. "Obstacle avoidance controller generating attainable set-points for the navigation of multi-robot system". In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2013, pp. 487–492.

[4] Ahmed Benzerrouk, Lounis Adouane, and Philippe Martinet. "Stable navigation in formation for a multi-robot system based on a constrained virtual structure". In: *Robotics and Autonomous Systems* 62.12 (2014), pp. 1806–1815.

[5] José Miguel Vilca. "Safe and Flexible Hybrid Control Architecture for the Navigation in Formation of a Group of Vehicles". PhD thesis. Université Blaise Pascal, 2015.

[6] Jose M.V. Vilca. "Safe and Flexible Hybrid Control Architecture for the Navigation in Formation of a Group of Vehicles". In: (2015).

[7] José Vilca, Lounis Adouane, and Youcef Mezouar. "Adaptive leader-follower formation in cluttered environment using dynamic target reconfiguration". In: *Distributed Autonomous Robotic Systems*. Springer, 2016, pp. 237–254.

[8] Pierre Avanzini. "Modélisation et Commande d'un Convoi de Véhicules Urbains par Vision". In: (2010).

[9] Audrey Guillet. “Commande locale décentralisée de robots mobiles en formation en milieu naturel”. In: (2015).

[10] Dimia Iberraken, Lounis Adouane, and Dieumet Denis. “Safe autonomous overtaking maneuver based on inter-vehicular distance prediction and multi-level bayesian decision-making”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 3259–3265.

[11] Dimia Iberraken, Lounis Adouane, and Dieumet Denis. “Multi-level bayesian decision-making for safe and flexible autonomous navigation in highway environment”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 3984–3990.

[12] Daniel J. Fagnant and Kara Kockelman. “Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations”. In: *Transportation Research Part A: Policy and Practice* 77 (2015), pp. 167–181.

[13] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerk, Eric Jensen, Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian, and Pamela Mahoney. “Stanley: The robot that won the DARPA Grand Challenge”. In: *Journal of Field Robotics* 23.9 (2006), pp. 661–692.

[14] Deborah Braid, Alberto Broggi, and Gary Schmiedel. “The TerraMax autonomous vehicle: Field Reports”. In: *J. Robot. Syst.* 23.9 (2006), pp. 693–708.

[15] Gabriel M. Hoffmann, Claire J. Tomlin, Michael Montemerlo, and Sebastian Thrun. “Autonomous Automobile Trajectory Tracking for Off-Road Driving : Controller Design , Experimental Validation and Racing”. In: *Proceedings of the American Control Conference* (2007), pp. 2296–2301.

[16] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, M N Clark, John Dolan, Dave Duggins, Tugrul Galatali, and Chris Geyer. “Autonomous driving in urban environments: Boss and the urban challenge”. In: *Journal of Field Robotics* 25.8 (2008), pp. 425–466.

[17] Jörn Marten Wille and Thomas Form. “Low level control in a modular system architecture for realizing precise driving maneuvers of the autonomous vehicle Caroline”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. IEEE, Oct. 2008, pp. 705–710.

[18] John Leonard, Jonathan How, Seth Teller, Mitch Berger, Stefan Campbell, Gaston Fiore, Luke Fletcher, Emilio Frazzoli, Albert Huang, and Sertac Karaman. “A perception-driven autonomous urban vehicle”. In: *Journal of Robotic Systems* 25.10 (Oct. 2008), pp. 727–774.

[19] Michael Montemerlo, Jan Becker, Suhrid Bhat, Hendrik Dahlkamp, Dmitri Dolgov, Scott Ettinger, Dirk Haehnel, Tim Hilden, Gabe Hoffmann, Burkhard Huhnke, Doug Johnston, Stefan Klumpp, Dirk Langer, Anthony Levandowski, Jesse Levinson, Julien Marcil, David Orenstein, Johannes Paefgen, Isaac Penny, Anna Petrovskaya, Mike Pflueger, Ganymed Stanek, David Stavens, Antone Vogt, and Sebastian Thrun. “Junior: The stanford entry in the urban challenge”. In: *Springer Tracts in Advanced Robotics*. Vol. 56. 2009, pp. 91–123.

[20] Marten Wille, Falko Saust, and Markus Maurer. “Stadtpilot : Driving Autonomously on Braunschweig ’ s Inner Ring Road”. In: *2010 IEEE Intelligent Vehicles Symposium* (June 2010), pp. 506–511.

[21] Jörn Marten Wille, Falko Saust, and Markus Maurer. “Comprehensive treated sections in a trajectory planner for realizing autonomous driving in Braunschweig’s urban traffic”. In: *13th International IEEE Conference on Intelligent Transportation Systems*. IEEE. 2010, pp. 647–652.

[22] Roozbeh Kianfar, Bruno Augusto, Alireza Ebadighajari, Usman Hakeem, Josef Nilsson, Ali Raza, Reza S Tabar, Naga VishnuKanth Irukulapati, Cristofer Englund, Paolo Falcone, et al. “Design and experimental validation of a cooperative driving system in the grand cooperative driving challenge”. In: *IEEE transactions on intelligent transportation systems* 13.3 (2012), pp. 994–1007.

[23] Andreas Geiger, Philip Lenz, and Raquel Urtasun. “Are we ready for autonomous driving? the kitti vision benchmark suite”. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2012, pp. 3354–3361.

[24] Cristofer Englund, Lei Chen, Jeroen Ploeg, Elham Semsar-Kazerooni, Alexey Voronov, Hoai Hoang Bengtsson, and Jonas Didoff. “The grand cooperative driving challenge 2016: boosting the introduction of cooperative automated vehicles”. In: *IEEE Wireless Communications* 23.4 (2016), pp. 146–152.

[25] Prateek Bansal and Kara M Kockelman. “Forecasting Americans’ long-term adoption of connected and autonomous vehicle technologies”. In: *Transportation Research Part A: Policy and Practice* 95 (2017), pp. 49–63.

[26] Scott Le Vine, Alireza Zolfaghari, and John Polak. “Autonomous cars: The tension between occupant experience and intersection capacity”. In: *Transportation Research Part C: Emerging Technologies* 52 (Mar. 2015), pp. 1–14.

[27] Pedro Aguiar, Andrea Alessandretti, and Colin N Jones. “Trajectory-tracking and path-following controllers for constrained underactuated vehicles using Model Predictive Control”. In: *European Control Conference* (2013), pp. 1371–1376.

[28] José Miguel Vilca, Lounis Adouane, and Youcef Mezouar. “A novel safe and flexible control strategy based on target reaching for the navigation of urban vehicles”. In: *Robotics and Autonomous Systems* 70 (Aug. 2015), pp. 215–226.

[29] Joshué Pérez, Vicente Milanés, and Enrique Onieva. “Cascade architecture for lateral control in autonomous vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 12.1 (Mar. 2011), pp. 73–82.

[30] Alexey S. Matveev, Hamid Teimoori, and Andrey V. Savkin. “A method for guidance and control of an autonomous vehicle in problems of border patrolling and obstacle avoidance”. In: *Automatica* 47.3 (2011), pp. 515–524.

[31] Dongkyoung Chwa and Jin Young Choi. “Adaptive Nonlinear Guidance Law Considering Control Loop Dynamics”. In: *IEEE Transactions on Aerospace and Electronic Systems* 39.4 (2003), pp. 1134–1143.

[32] Ji-Chang Lo and Min-Long Lin. “Robust H∞ nonlinear modeling and control via uncertain fuzzy systems”. In: *Fuzzy Sets and Systems* 143.2 (Apr. 2004), pp. 189–209.

[33] Simon Hecker. “Robust Hinf-based vehicle steering control design”. In: *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE*. IEEE, Oct. 2006, pp. 1294–1299.

[34] A. Pedro Aguiar. “Trajectory-Tracking and Path-Following of Underactuated Autonomous Vehicles With Parametric Modeling Uncertainty”. In: *IEEE Transactions on Automatic Control* 52.8 (Aug. 2007), pp. 1362–1379.

[35] Pierre Apkarian, Pascal Gahinet, and Greg Becker. “Self-scheduled H-inf control of linear parameter-varying systems: a design example”. In: *Automatica* 31.9 (Sept. 1995), pp. 1251–1261.

[36] Franco Blanchini, Daniele Casagrande, Stefano Miani, and Umberto Viaro. “Parametric gain-scheduling control via LPV-stable realization”. In: *Control of Linear Parameter Varying Systems with Applications*. Boston, MA: Springer US, 2014, pp. 61–89.

[37] Masih Hanifzadegan and Ryozo Nagamune. “Switching gain-scheduled control design for flexible ball-screw drives”. In: *Journal of Dynamic Systems, Measurement, and Control* 136.1 (2014).

[38] Moustapha Doumiati, Sebastian Erhart, J Martinez, Olivier Sename, and Luc Dugard. “Adaptive control scheme for road profile estimation: application to vehicle dynamics”. In: *IFAC Proceedings Volumes* 47.3 (2014), pp. 8445–8450.

[39] Paolo Falcone, H. Eric Tseng, Francesco Borrelli, Jahan Asgari, and Davor Hrovat. “MPC-based yaw and lateral stabilisation via active front steering and braking”. In: *Vehicle System Dynamics* 46.March 2015 (Sept. 2008), pp. 611–628.

[40] F Künhe, J Gomes, and W Fetter. “Mobile Robot Trajectory Tracking Using Model Predictive Control”. In: *II IEEE latin-american robotics* (2005), pp. 1–7.

[41] Guilherme V. Raffo, Guilherme K. Gomes, Julio E. Normey-Rico, Christian R. Kelber, and Leandro B. Becker. “A predictive controller for autonomous vehicle path tracking”. In: *IEEE Transactions on Intelligent Transportation Systems* 10.1 (Mar. 2009), pp. 92–102.

[42] Paolo Falcone and Francesco Borrelli. “Low complexity MPC schemes for integrated vehicle dynamics control problems”. In: *International Symposium on Advanced Vehicle Control* 1 (2008), pp. 875–880.

[43] Pedro Rodríguez and Didier Dumur. “Generalized predictive control robustification under frequency and time-domain constraints”. In: *IEEE Transactions on Control Systems Technology* 13.4 (July 2005), pp. 577–587.

[44] Benjamin Gutjahr, Lutz Groll, and Moritz Werling. “Lateral Vehicle Trajectory Optimization Using Constrained Linear Time-Varying MPC”. In: *IEEE Transactions on Intelligent Transportation Systems* (2016), pp. 1–10.

[45] Alexander Katriniok and Dirk Abel. “LTV-MPC approach for lateral vehicle guidance by front steering at the limits of vehicle dynamics”. In: *Proceedings of the IEEE Conference on Decision and Control*. IEEE, Dec. 2011, pp. 6828–6833.

[46] Jean Baptiste Braconnier, Roland Lenain, and Benoit Thuilot. “Ensuring path tracking stability of mobile robots in harsh conditions: An adaptive and predictive velocity control”. In: *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, May 2014, pp. 5268–5273.

[47] Assad Al Alam, Ather Gattami, Karl H. Johansson, and Claire J. Tomlin. “Establishing safety for heavy duty vehicle platooning: A game theoretical approach”. In: *IFAC Proceedings Volumes (IFAC-PapersOnline)*. Vol. 18. PART 1. Elsevier, 2011, pp. 3818–3823.

[48] Maytheewat Aramrattana, Tony Larsson, Jonas Jansson, and Cristofer Englund. “Dimensions of cooperative driving, its and automation”. In: *2015 IEEE intelligent vehicles symposium (IV)*. IEEE. 2015, pp. 144–149.

[49] Peiyao Shen, Hengfei Zou, Xuebo Zhang, Yongfu Li, and Yongchun Fang. “Platoon of Autonomous Vehicles with Rear-End Collision Avoidance Through Time-Optimal Path-Constrained Trajectory Planning”. In: (2017), pp. 232–237.

[50] Jennie Lioris, Ramtin Pedarsani, Fatma Yildiz Tascikaraoglu, and Pravin Varaiya. “Platoons of connected vehicles can double throughput in urban roads”. In: *Transportation Research Part C: Emerging Technologies* 77 (2017), pp. 292–305. arXiv: 1511.00775.

[51] S. Ilgin Guler, Monica Menendez, and Linus Meier. “Using connected vehicle technology to improve the efficiency of intersections”. In: *Transportation Research Part C: Emerging Technologies* 46 (2014), pp. 121–131.

[52] Masoud Bashiri, Hassan Jafarzadeh, and Cody H Fleming. “Paim: Platoon-based autonomous intersection management”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 374–380.

[53] Di Feng, Lars Rosenbaum, Claudius Glaeser, Fabian Timm, and Klaus Dietmayer. “Can we trust you? on calibration of a probabilistic object detector for autonomous driving”. In: *arXiv preprint arXiv:1909.12358* (2019).

[54] Jackeline Rios-Torres and Andreas A Malikopoulos. “Impact of partial penetrations of connected and automated vehicles on fuel consumption and traffic flow”. In: *IEEE Transactions on Intelligent Vehicles* 3.4 (2018), pp. 453–462.

[55] Jackeline Rios-Torres, Andreas Malikopoulos, and Pierluigi Pisu. “Online Optimal Control of Connected Vehicles for Efficient Traffic Flow at Merging Roads”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2015-Octob. IEEE, Sept. 2015, pp. 2432–2437.

[56] Wenjing Cao, Masakazu Mukai, Taketoshi Kawabe, Hikaru Nishira, and Noriaki Fujiki. “Cooperative vehicle path generation during merging using model predictive control with real-time optimization”. In: *Control Engineering Practice* 34 (2015), pp. 98–105.

[57] Ioannis A. Ntousakis, Ioannis K. Nikolos, and Markos Papageorgiou. “Optimal vehicle trajectory planning in the context of cooperative merging on highways”. In: *Transportation Research Part C: Emerging Technologies* 71 (2016), pp. 464–488.

[58] Jackeline Rios-Torres and Andreas A. Malikopoulos. “Automated and Cooperative Vehicle Merging at Highway On-Ramps”. In: *IEEE Transactions on Intelligent Transportation Systems* 18.4 (2017), pp. 780–789.

[59] Simon Ulbrich and Markus Maurer. “Situation Assessment in Tactical Lane Change Behavior Planning for Automated Vehicles”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings*. Vol. 2015-Octob. IEEE, Sept. 2015, pp. 975–981.

[60] Christoph Burger and Martin Lauer. “Cooperative multiple vehicle trajectory planning using miqp”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 602–607.

[61] Michael Ardelt, Constantin Coester, and Nico Kaempchen. “Highly automated driving on freeways in real traffic using a probabilistic framework”. In: *IEEE Transactions on Intelligent Transportation Systems* 13.4 (Dec. 2012), pp. 1576–1585.

[62] Kyle Baker, Matthew Cooper, Peter Heidlauf, and Timothy Sands. “Autonomous trajectory generation for deterministic artificial intelligence”. In: *Electrical and Electronic Engineering* 8.3 (2018), pp. 59–68.

[63] Kwangseok Oh, Sungyoul Park, Jongmin Lee, and Kyongsu Yi. “Functional perspective-based probabilistic fault detection and diagnostic algorithm for autonomous vehicle using longitudinal kinematic model”. In: *Microsystem Technologies* 24.11 (2018), pp. 4527–4537.

[64] Andreas A. Malikopoulos, Christos G. Cassandras, and Yue J. Zhang. “A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections”. In: *Automatica* 93 (2018), pp. 244–256. arXiv: 1602.03786.

[65] Shravan Krishnan, Rahul Ramakrishnan, Vijay Arvindh, et al. “A look at motion planning for autonomous vehicles at an intersection”. In: *arXiv preprint arXiv:1806.07834* (2018).

[66] Changxi Ma, Wei Hao, Aobo Wang, and Hongxing Zhao. “Developing a coordinated signal control system for urban ring road under the vehicle-infrastructure connected environment”. In: *Ieee Access* 6 (2018), pp. 52471–52478.

[67] Seyed Alireza Fayazi and Ardalan Vahidi. “Mixed-Integer Linear Programming for Optimal Scheduling of Autonomous Vehicle Intersection Crossing”. In: *IEEE Transactions on Intelligent Vehicles* 3.3 (2018), pp. 287–299.

[68] Stefanie Manzinger and Matthias Althoff. “Tactical Decision Making for Cooperative Vehicles Using Reachable Sets”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. IEEE, Nov. 2018, pp. 444–451.

[69] Liuhui Zhao, Andreas Malikopoulos, and Jackeline Rios-Torres. “Optimal Control of Connected and Automated Vehicles at Roundabouts: An Investigation in a Mixed-Traffic Environment”. In: *IFAC-PapersOnLine* 51.9 (2018), pp. 73–78. arXiv: 1710.11295.

[70] Eduardo Rauh Müller, Rodrigo Castelan Carlson, and Werner Kraus Junior. “Intersection control for automated vehicles with MILP”. In: *IFAC-PapersOnLine* 49.3 (2016), pp. 37–42.

[71] Mohamed Tlig, Olivier Buffet, and Olivier Simonin. “Decentralized traffic management: A synchronization-based intersection control”. In: *2014 International Conference on Advanced Logistics and Transport, ICALT 2014*. 2014, pp. 109–114.

[72] Mohamed Tlig, Olivier Buffet, and Olivier Simonin. “Stop-free strategies for traffic networks: Decentralized on-line optimization”. In: *Frontiers in Artificial Intelligence and Applications*. Vol. 263. 2014, pp. 1191–1196.

[73] Kun Wei and Bingyin Ren. “A method on dynamic path planning for robotic manipulator autonomous obstacle avoidance based on an improved RRT algorithm”. In: *Sensors* 18.2 (2018), p. 571.

[74] Robert Hult, Gabriel R. Campos, Paolo Falcone, and Henk Wymeersch. “An approximate solution to the optimal coordination problem for autonomous vehicles at intersections”. In: *Proceedings of the American Control Conference*. Vol. 2015-July. IEEE, July 2015, pp. 763–768.

[75] Robert Hult, Mario Zanon, Sebastien Gros, and Paolo Falcone. “Primal decomposition of the optimal coordination of vehicles at traffic intersections”. In: *2016 IEEE 55th Conference on Decision and Control, CDC 2016*. 2016, pp. 2567–2573.

[76] Robert Hult, Mario Zanon, Sébastien Gros, and Paolo Falcone. “Energy-optimal coordination of autonomous vehicles at intersections”. In: *2018 European Control Conference (ECC)*. IEEE. 2018, pp. 602–607.

[77] Mario Zanon, Sébastien Gros, Henk Wymeersch, and Paolo Falcone. “An Asynchronous Algorithm for Optimal Vehicle Coordination at Traffic Intersections”. In: *IFAC-PapersOnLine* 50.1 (2017), pp. 12008–12014. arXiv: 1705.01197.

[78] Changliu Liu, Chung-Wei Lin, Shinichi Shiraishi, and Masayoshi Tomizuka. “Distributed Conflict Resolution for Connected Autonomous Vehicles”. In: *IEEE Transactions on Intelligent Vehicles* 3.1 (2018), pp. 18–29.

[79] Erik Karlsson and Nasser Mohammadiha. “A Statistical GPS Error Model for Autonomous Driving”. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2018, pp. 754–759.

[80] Di Feng, Lars Rosenbaum, and Klaus Dietmayer. “Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 3266–3273.

[81] Hironori Suzuki and Yoshitaka Marumo. “A New Approach to Green Light Optimal Speed Advisory (GLOSA) Systems for High-Density Traffic Flowe”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. 2018, pp. 362–367.

[82] Biao Xu, Shengbo Eben Li, Yougang Bian, Shen Li, Xuegang Jeff Ban, Jianqiang Wang, and Keqiang Li. “Distributed conflict-free cooperation for multiple connected vehicles at unsignalized intersections”. In: *Transportation Research Part C: Emerging Technologies* 93 (2018), pp. 322–334.

[83] Xian Xu and Chiang-Ku Fan. “Autonomous vehicles, risk perceptions and insurance demand: An individual survey in China”. In: *Transportation research part A: policy and practice* 124 (2019), pp. 549–556.

[84] Ismail H. Zohdy and Hesham Rakha. “Game theory algorithm for intersection-based cooperative adaptive cruise control (CACC) systems”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. IEEE, Sept. 2012, pp. 1097–1102.

[85] Kaidi Yang, S. Ilgin Guler, and Monica Menendez. “Isolated intersection control for various levels of vehicle technology: Conventional, connected, and automated vehicles”. In: *Transportation Research Part C: Emerging Technologies* 72 (2016), pp. 109–129. arXiv: arXiv:1011.1669v3.

[86] Lu Feng, Clemens Wiltsche, Laura Humphrey, and Ufuk Topcu. “Synthesis of human-in-the-loop control protocols for autonomous systems”. In: *IEEE Transactions on Automation Science and Engineering* 13.2 (2016), pp. 450–462.

[87] Falco Creemers, Alejandro Ivan Morales Medina, Erjen Lefeber, and Nathan van de Wouw. “Design of a supervisory controller for cooperative intersection control using model predictive control”. In: *IFAC-PapersOnLine* 51.33 (2018), pp. 74–79.

[88] Weiming Zhao, Dong Ngoduy, Simon Shepherd, Ronghui Liu, and Markos Papageorgiou. “A platoon based cooperative eco-driving model for mixed automated and human-driven vehicles at a signalised intersection”. In: *Transportation Research Part C: Emerging Technologies* 95 (2018), pp. 802–821.

[89] Yoriyoshi Hashimoto, Yanlei Gu, Li Ta Hsu, Miho Iryo-Asano, and Shunsuke Kamijo. “A probabilistic model of pedestrian crossing behavior at signalized intersections for connected vehicles”. In: *Transportation Research Part C: Emerging Technologies* 71 (2016), pp. 164–181.

[90] Ashwin Carvalho, Stéphanie Lefévre, Georg Schildbach, Jason Kong, and Francesco Borrelli. “Automated driving: The role of forecasts and uncertainty—A control perspective”. In: *European Journal of Control* 24 (2015), pp. 14–32.

[91] Charles Philippe, Lounis Adouane, Benoit Thuilot, Antonios Tsourdos, and Hyo Sang Shin. “Risk and comfort management for multi-vehicle navigation using a flexible and robust cascade control architecture”. In: *2017 European Conference on Mobile Robots, ECMR 2017*. IEEE, Sept. 2017, pp. 1–7.

[92] Charles Philippe, Lounis Adouane, Benoît Thuilot, Antonios Tsourdos, and Hyo Sang Shin. “Safe and Online MPC for Managing Safety and Comfort of Autonomous Vehicles in Urban Environment”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. 2018, pp. 300–306.

[93] Charles Philippe, Lounis Adouane, Antonios Tsourdos, Hyo-Sang Shin, and Benoît Thuilot. “Probability Collectives Algorithm applied to Decentralized Intersection Coordination for Connected Autonomous Vehicles”. In: *Intelligent Vehicles Symposium* (2019).

Chapter 2

Risk and Comfort Management for Multi-Vehicle Navigation using a Flexible and Robust Cascade Control Architecture

Abstract

This chapter presents a new cascade control architecture formulation for addressing the problem of autonomous vehicle trajectory tracking under risk and comfort constraints. The integration of these constraints has been split between an inner and an outer loop. The former is made of a robust controller dedicated to stabilizing the car dynamics while the latter uses a nonlinear Model Predictive Control (MPC) scheme to control the car trajectory. The proposed structure aims to take into account several important aspects, such as robustness considerations and disturbance rejection (inner loop) as well as control signal and state constraints, tracking error monitoring and tracking error prediction (outer loop). The proposed design has been validated in simulation while comparing mainly with common kinematic trajectory controllers.

2.1 Introduction

Autonomous vehicles are getting more and more important in the academic field as well as in the industry. Constructors such as BMW, Volvo, Tesla and other companies such as Uber are trying to reach the next step of autonomy for consumer cars. Recent incidents or accidents [1] have highlighted the need for safe and robust architectures.

The objectives of this chapter are to describe the development of a control architecture for autonomous vehicle under the constraints specific to urban passenger transportation. The two major categories of constraints are the safety and the comfort of the passengers. Indicators of tracking performance and health monitoring will be developed to allow for the future development of a supervision layer in the architecture.

In the end, the proposed architecture aims to be a generic and easily transposable solution for single and multi-vehicle navigation. These aims will be reached with a combination of an MPC controller for the tracking and a robust H_∞ controller for yaw stabilization.

The numerical applications are done for the IPCar vehicles, which are autonomous electric vehicles for urban transportation (cf. Fig. 2.1).

Figure 2.1: IPCar autonomous transport vehicles in a coordinated maneuver

The remainder of this chapter will be organized as follows. Section 2.2 will give an overview on the works on autonomous vehicle control and will describe the pertinence and novelty of the proposed architecture. Section 2.3 will explain the design of the two blocks of the cascade control architecture. Section 2.4 will show comparative simulations in a range situations for different controllers and architectures.

2.2 Towards a Flexible And Robust Control Architecture

2.2.1 Related works

The existing works on autonomous vehicle lateral control can be separated in two main categories. There are trajectory/target tracking algorithms on one side and yaw stabilization algorithms on the other side. Some approaches will be described as *integrated* and aim to fulfil those two tasks at the same time. For the tracking task, kinematic controllers are used in [2] and [3]. Depending on the implementation they can cover a range of speeds and situations but the lack of consideration of dynamic effects can be problematic for highly dynamic situations. Yaw stabilization (or active steering) schemes include [4] and [5] which use respectively the linear robust control framework and the MPC framework. Both approaches show very good performances, at the expense of a robustness proof in [5] with the MPC design. As for integrated approaches, a first example based on a nonlinear kinematic controller is presented in [6]. Empirical terms have been added to a trajectory controller for dynamic effects compensation. In [7] is presented another integrated approach based on linear adaptive control. In [8] is presented an approach based on MPC. The choice of the technique mainly depends on the design objectives. Some other works cover the two tasks but with separate controllers for each one, which has the advantage to separate the objectives for each task and give an adapted answer. These approaches usually use two controllers in a cascade architecture. Such works for ground vehicles include [9] and [10]. Linear control is often used for the inner control loop. A similar cascade architecture for aerospace applications is presented in [11]. It is a common approach in Unmanned Aerial Vehicles (UAV) design. Usually, integrated architectures fail to take into account the comfort, safety and implementability at the same time since they are more performance based. This is what the proposed design aims to do.

2.2.2 Architecture design

Compared to integrated design, the advantage of cascade architectures is the flexibility for multi-vehicle navigation and the natural separation between kinematic and dynamic phenomenons. Moreover, the trajectory tracking error dynamics are not on the same time scale than the car yaw dynamics. This separation is analogous to the Guidance/Control framework [12] even though it is more a kinematic/dynamic separation.

The MPC has been used with great success to address trajectory tracking and yaw stabilization as an integrated approach. However it seems more relevant in this application to separate the yaw stabilization in a cascade architecture.

The inner loop for yaw stabilization has to deal with the uncertainty on the vehicle physical parameters. For instance the weight and inertia of the vehicle will undergo wide variations because of the variable passenger repartition. The friction coefficient μ will also be unknown and unmeasurable.

For these reasons the linear robust control framework has been chosen to design the inner loop controller. It enables the robustness assessment of the controller under the model uncertainty, which is a big asset for ensuring safety.

The designed architecture is shown in Fig. 2.2. The MPC controller generates a desired yaw rate r_d and a desired speed v_d to track the reference trajectory X_{ref}. The inner controller tracks the signal (r_d, v_d) by generating a desired steering angle δ_d and acceleration a_d to the vehicle. The state X_{dyn} (resp. X_{kin}) is the dynamic (resp. kinematic) state of the vehicle. It will be defined in section 2.3.1 (resp. 2.3.2).

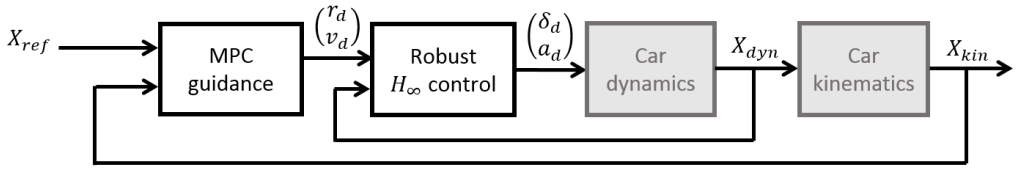

Figure 2.2: Designed cascade architecture

2.3 Proposed Cascade Control Architecture

This architecture will have to fulfil the tasks of trajectory following as well as target tracking in order to be fit for multi-vehicle navigation. Trajectory following is the action of following a predefined line (defined in (x, y) coordinates) with a reference speed v_{ref} at each point. For target following the same information is available but only at the current instant. More states can be known about the target, whether by sensing or communication. As mentioned in section 2.2, the proposed architecture will have to handle the two tasks while ensuring passenger comfort and safety. The inner loop design will be described first and then the outer loop design, since it depends on the former.

2.3.1 Inner loop design

A mixed sensitivity H_∞ controller has been chosen because it is an optimal design technique, and it has explicit disturbance rejection and frequency domain performance specifications. The disturbance rejection characteristics are interesting because it will dictate the vehicle's behaviour under wind gusts. Too strong of a reaction could be seen as dangerous and uncomfortable.

For this application the model in the state space form is shown in Equation (2.1). It is based on the kinematic bicycle representation. It consists of a 2 degrees of freedom (DoF) model for the sideslip β and yaw rate r dynamics and a first order model of time constant τ_{act} for the evolution of δ, the steering angle. The overall state is $X_{dyn} = (\beta, r, \delta)^T$ (resp. sideslip, yaw rate and steering angle defined in Fig. 2.3). The input is the desired steering angle δ_d. This model has been presented in depth in [13]. It is suitable for low speed situations such as urban traffic.

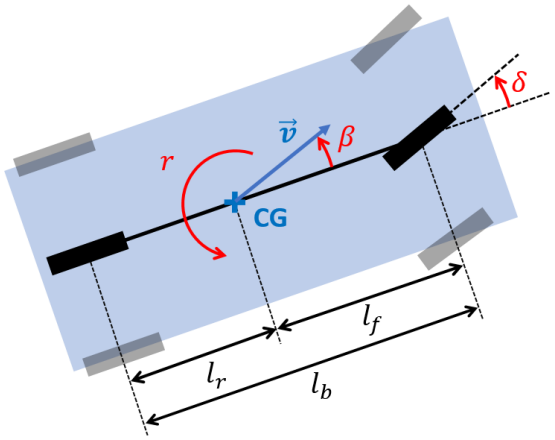

Figure 2.3: Dynamic bicycle model conventions

$$\dot{X}_{dyn} = \begin{pmatrix} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \\ 0 & 0 & -\tau_{act}^{-1} \end{pmatrix} X_{dyn} + \begin{pmatrix} 0 \\ 0 \\ \tau_{act}^{-1} \end{pmatrix} \delta_d \tag{2.1}$$

The coefficients a_{ij} and b_i depend on the constants $c_f, c_r, m, v, J, l_b, l_f$ defined in Table 2.1.

The following range of configurations has been considered:

- from zero passenger to four passengers of 100kg each
- speeds from 1.5m/s to 4.5m/s

Table 2.1: Parameter values for uncertain car plant study

parameter	**notation**	**value**
wheelbase	l_b	3m
inertial radius	i_r	1.5m
cornering stiffnesses	c_f, c_r	700N/deg
actuator time constant	τ_{act}	0.6s
mass	m	600kg ± 30%
CG position to front axle	l_f	1.4m ± 20%
speed	v	3m/s ± 50%
friction coefficient	μ	0.65 ± 50%

- friction coeff in [0.3, 1] (from a slippery wet road to a dry road)
- Centre of Gravity (CoG) from 1.1m to 1.6m to front axle (because of the passenger repartition)

The corresponding uncertain variables have been summarized in Table 2.1. As a way to reduce the number of uncertain variables, J has been considered proportional to m with the intermediate of the inertial radius i_r (c.f. [13]).

The result of the H_∞ design is shown in (2.2) with K_{CF} being the feedforward filter and K_{DR} being the feedback filter. The filters have been approximated by second order transfer functions to make the implementation faster.

$$\begin{cases} K_{CF}(s) &= \dfrac{s^2 + 21.2s + 158.2}{s^2 + 20s + 156.3} \\ \\ K_{DR}(s) &= \dfrac{250(s + 124)(s + 1.67)}{s(s + 17.5)} \end{cases} \tag{2.2}$$

The robustness analysis shows that the design is robust to the modelled uncertainty. The performance analysis shows that the rise time for the yaw rate is always between $0.3s$ and $0.8s$ with a nominal value at $0.5s$ and the system is always well damped (as seen in Fig. 2.4).

Since the system is always well damped (there is no overshoot), it will be approximated by a first order system in the following developments, making for a simpler model for the MPC.

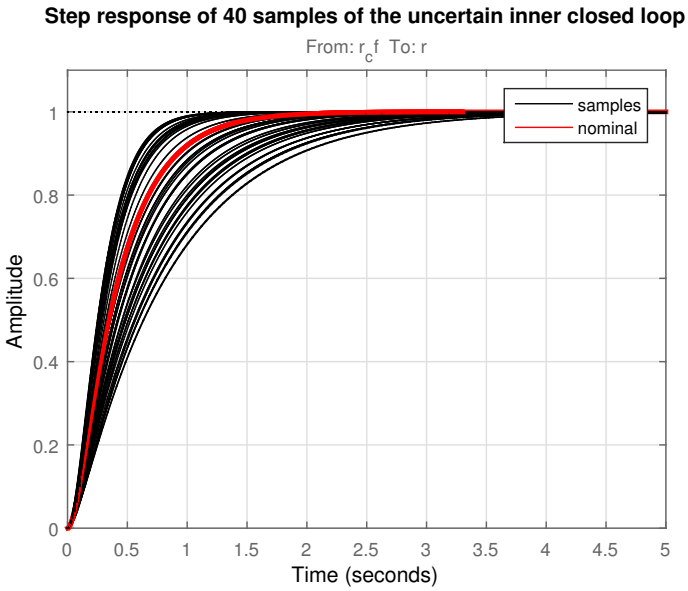

Figure 2.4: Closed-loop step response envelope

2.3.2 Outer loop design

The outer loop controller (cf. Fig 2.2) needs to address the problem of stabilizing the vehicle around a moving target or a reference trajectory.

Even though it is a classical approach, MPC has good qualities to address our problem: it finds an optimal solution, can inherently deal with non linear processes such as car steering and can include constraints on the states and control signal (linked to comfort and safety). The prediction is also valuable to deal with slow dynamics and to anticipate future situations.

The main problem of MPC is that it always needs a reference trajectory over a finite horizon. In multi-vehicle navigation it is not always available and thus needs to be predicted to enable the use of MPC. Fig. 2.5 shows a situation where a reference trajectory for the virtual target T that V_F (the *follower*) has to follow is only partially known over the MPC horizon. In this situation the aim is to keep constant offsets Δy_{ref} and Δs_{ref} with the trajectory of V_L (the *leader*). As the prediction horizon of the MPC depends on the model dynamics, it is not correlated to how far the target is to the leading vehicle. Thus a big MPC horizon and a small leader/target distance can lead to such a situation.
The reference trajectory prediction is based on the hypotheses that the yaw rate r and speed v of the leader are constant. The states for the prediction are the same as the MPC presented later in (2.3). It generates a reference trajectory $\hat{\mathbf{X}}_{\text{ref}}(k+N|k)$ over the MPC prediction horizon N. The MPC scheme can then be used seamlessly whether there is a

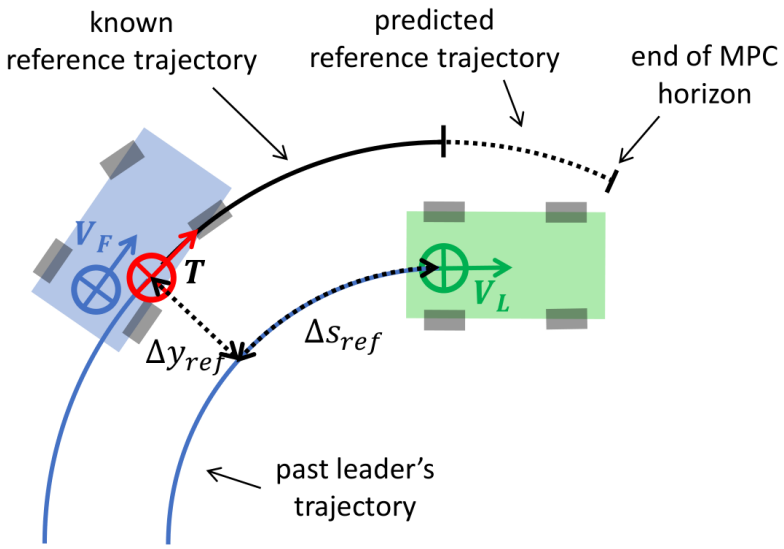

Figure 2.5: Partial availability of a reference trajectory in leader/follower navigation

reference trajectory over the whole prediction horizon or not and makes the architecture more flexible.

The chosen model for the MPC is shown in (2.3). At a timestep k, the state is $X_{kin}(k) = (x_k, y_k, \psi_k, r_k, v_k)^T$ where x_k, y_k and ψ_k are defined in Fig. 2.6, r_k is the yaw rate of the car and v_k its speed. The input is $U = (r_d, v_d)^T$ (the desired yaw rate and speed). The only parameters that describe the vehicle's dynamics are τ_r and τ_v, the time constants for the yaw rate and the speed responses which have been identified on the closed inner loop (cf. Table 2.2). The last parameter is the sampling time T_s of the loop. These responses are assumed to be described by first order models. It is a realistic hypothesis as long as the real responses are well damped. This is the case here as seen in Fig. 2.4 thanks to the inner loop designed in section 2.3.1.

$$\begin{cases} x_{k+1} &= x_k + T_s v_k \cos\psi_k \\ y_{k+1} &= y_k + T_s v_k \sin\psi_k \\ \psi_{k+1} &= \psi_k + T_s r_k \\ r_{k+1} &= r_k + \tau_r^{-1}(r_d - r_k) \\ v_{k+1} &= v_k + \tau_v^{-1}(v_d - v_k) \end{cases} \tag{2.3}$$

The MPC scheme finds a control input $\mathbf{U}_{opt}$ that minimizes a cost function $J(\mathbf{U})$, where $\mathbf{U} = (U_k, ..., U_{k+N})$ is series of control signals to test over the prediction horizon N. This function is usually composed of a term that penalizes the errors to the reference trajectory and other terms to smooth the control signal.

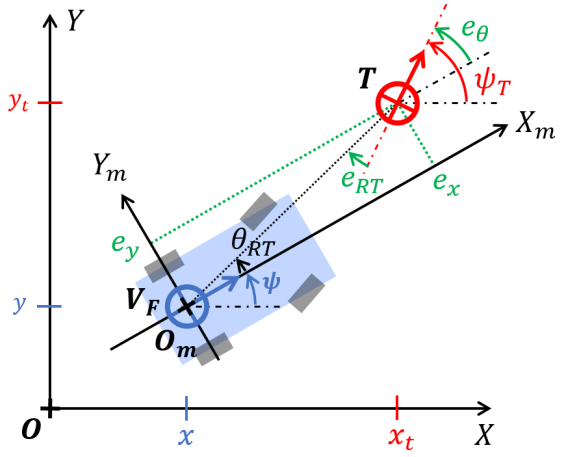

Figure 2.6: Navigation errors definition

The penalized terms are:

- $\tilde{\mathbf{X}}$, the matrix of differences w.r.t. the reference trajectory over the prediction horizon N
- $\hat{\mathbf{E}}(k+N|k)$ the navigation errors over the prediction horizon N. These errors are defined in Fig. 2.6 and further detailed in [2]. They are a convenient way to work in a local frame that is in the vehicle's orientation.
- $\tilde{\mathbf{U}}$ the difference between the test input signal and the previous optimal input signal.
- $\mathbf{U}_\Delta$ the differences between two successive input values in the tested input signal (a.k.a. control effort).

The chosen cost function is defined in (2.4). It is a weighted sum of the penalty terms:

$$\begin{aligned} J(\mathbf{U}) = \tilde{\mathbf{X}}^T Q \tilde{\mathbf{X}} + \hat{\mathbf{E}}(k+N|k)^T S\, \hat{\mathbf{E}}(k+N|k) \\ + \tilde{\mathbf{U}}^T R \tilde{\mathbf{U}} + \mathbf{U}_\Delta^T R_\Delta \mathbf{U}_\Delta \end{aligned} \tag{2.4}$$

The penalties on $\tilde{\mathbf{U}}$ and $\mathbf{U}_\Delta$ tend to smooth the tracking and are often encountered in non-linear MPC schemes. The penalty on the navigation errors allows to separately penalize the lateral and longitudinal errors (e_x and e_y) and have been preferred to the penalty on the state difference $\tilde{\mathbf{X}}$. The values of the weight matrices Q, S, R and R_Δ are compiled in Table 2.2. The raw state difference $\tilde{\mathbf{X}}$ has not been penalized except for the speed difference, which was found to smooth the longitudinal tracking.

For comfort and safety, the following constraints have been introduced:

- $|r_d| \leq r_{max}$

Table 2.2: MPC controller parameters

N	14
τ_r	0.5s
τ_v	1.4s
Q	diag(0, 0, 0, 0, 0.1)
S	diag(1, 2, 0, 0)
R	diag(0, 0)
R_Δ	diag(15, 15)
r_{max}	30 deg/s
dr_{max}	50 deg/s^2
v_{max}	4.5 m/s
$a_{y,max}$	5 m/s^2
$a_{x,max}$	3 m/s^2
$e_{x,max}$	0.5 m
$e_{y,max}$	0.2 m

- $|\dot{r}_d| \leq dr_{max}$
- $v_{min} \leq v_d \leq v_{max}$
- $|a_{y,d}| = |v_d r_d| \leq a_{y,max}$
- $|a_{x,d}| = \Delta v_d / T_s \leq a_{x,max}$
- $|\kappa_d| = |r_d / v_d| \leq \kappa_{max}$

The constraints on the yaw rate r_d, the yaw rate derivative $\dot{r}_d$ and the lateral/longitudinal accelerations $a_{y,d}$ and $a_{x,d}$ are here for comfort. The constraints on the speed v_d and the curvature κ_d are physical limitations of the vehicle. Two additional constraints on the tracking errors have been added: $|e_x| \leq e_{x,max}$ and $|e_y| \leq e_{y,max}$. These constraints are used to ensure the vehicle will stay within given bounds around the target. For example, the constraint on e_y will depend on the road width. The numerical values used for the constraints are compiled in Table 2.2. The optimization function used for the simulations is the *fmincon* Matlab optimization function under constraints. The input constraints will always be respected (it restricts the search space) and the function will try to find a solution that respects the additional constraints on the tracking errors, unless impossible. In the latter case, the optimization function's output will mention that the constraints were not respected. This information can be used to prevent the violation of constraints before it actually happens. Finally, the MPC scheme runs approximately in 20 seconds for a 10 seconds Simulink simulation with a modern computer (a 2013 Intel i5 equipped laptop) and a loop rate of 10Hz.

2.3.3 Conclusion

The proposed architecture is comprised of two loops in cascade that aim to solve the trajectory tracking/target following problems under comfort and safety constraints. The task separation allows to split the constraints and to focus on different aspects of the problem at each stage as well as allowing a simpler design overall.

2.4 Simulations and Results

2.4.1 Model Used

The model used for the simulations consists of a 3DoF bicycle chassis model with a linear tyre model (for both the longitudinal and lateral forces). The simulations have been performed in Simulink.

The model for the simulations is based on the one presented in section 2.3.1. The rotational wheel dynamics have been added in order to have a realistic longitudinal behaviour. Thus the state vector is defined as $X = (\beta, r, v, \omega_f, \omega_r)^T$ with the three first states already defined in Equation (2.1) and ω_f (resp. ω_r) is the front (resp. rear) wheel rotational speed. The input vector is $U = (\delta_d, a_d)^T$, the desired steering angle and longitudinal acceleration. These inputs are transformed into actual wheel angle and acceleration by the {actuator + controller} systems modelled by first order transfer functions of time constant 0.6s (resp 1s) and damping 0.8. The front and rear axles inertias are $J_f = J_r = 0.45\text{kg.m}^{-2}$ and the longitudinal tyre stiffnesses are $C_{l,f} = C_{l,r} = 10^4\text{N/deg}$. The other model parameters will be within the range of values used for controller design (cf. Table 2.1). Unless specified otherwise, the nominal values have been taken.

2.4.2 Simulation scenarios

Comparisons will be made with two kinematic controllers. The first controller is the one presented in [2] and [14]. For simplicity it will be referenced as the “Vilca” controller. It is a nonlinear control law designed for both dynamic target following and waypoint navigation. It is based on a Lyapunov function design and is a recent example of a flexible kinematic controller. The other one is the “Pure Pursuit” controller [15], a widely used kinematic controller for trajectory tracking because of its simplicity and efficiency. It is

Table 2.3: Common simulation parameters

parameter	**value**
initial vehicle position	$(1, 1)$ (m)
initial vehicle heading	$30°$
target path curvature variation rate	0.1 Hz
max target path curvature	$1/15\text{m}^{-1}$
Vilca's law coefficients K_V	$(1, 2.2, 8, 0.1, 0.01, 0.6)$
Pursuit law look-ahead coeff. k_{PP}	0.5
Pursuit law look-ahead dist. bounds	$[1, 5]$ (m)
outer loop sampling time $T_{s,g}$	$1/10$s
inner loop sampling time $T_{s,c}$	$1/50$s

a non linear controller that computes a curvature to reach a point on the trajectory at a look-ahead distance. This distance is usually proportional to the vehicle speed with a coefficient k_{PP} within a lower and upper bound (cf. Table 2.3). Both algorithms compute the steering angle corresponding to the desired curvature under kinematic hypotheses, thus not taking into account actuator delays and slip.

For the simulations, the virtual target follows a sinusoidal path at a constant speed (cf. Fig 2.7). For the MPC controller it is assumed that no information of the target's future path is available to put it in difficult conditions. As a consequence, the prediction module described in section 2.3.2 will be used. It is on the other hand assumed that the trajectory is entirely available when using the pursuit controller, thus giving it more favorable conditions.

Two test cases are presented:

- *Behaviour comparison*: A test at low speed to compare the behaviour of the three controllers. For this test the nominal values for the car model will mostly be used.
- *Safety and comfort assessment*: A test at a higher speed and non-nominal car model values to check the robustness of the approaches. This test will also serve to check if the comfort constraints with our approach are respected in more aggressive maneuvers.

The common (resp. variable) parameters for the simulation scenarios are compiled in Table 2.3 (resp. Table 2.4).

Table 2.4: Variable simulation parameters

parameter	**value**	
	First simulation	*Second simulation*
initial target position	(1.5, 1.5) (m)	(2, 2) (m)
initial target heading	30°	40°
Target speed	2m/s	4m/s
vehicle mass	600kg	750kg
friction coefficient μ	0.65	0.4

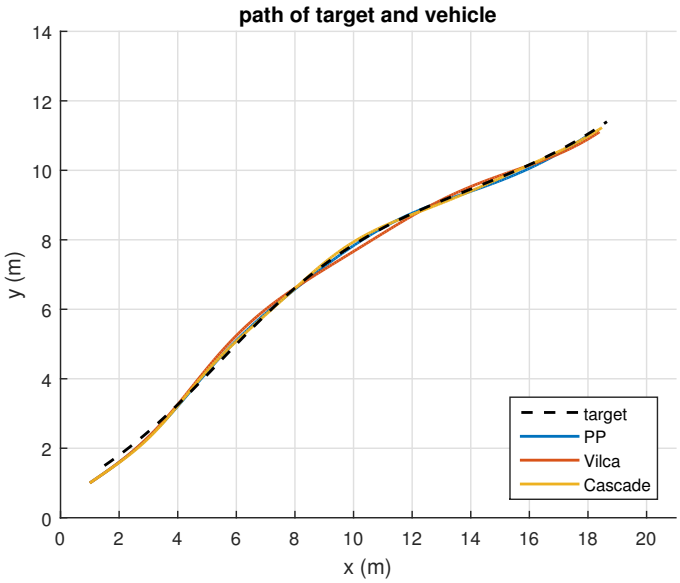

Figure 2.7: Example of path for the simulations (target and follower)

2.4.3 Behaviour comparison

All the nominal parameters for the model have been taken except the speed which is $2m/s$. The MPC shows a very good tracking compared to the two other methods (cf Fig. 2.8). However the control signals (Fig. 2.9) and the comfort indicators (Fig. 2.10) are approximately of the same magnitude for the three approaches. In easy maneuvers like this one the proposed architecture behaves well but has an edge on neither the pursuit controller or on Vilca's controller in terms of comfort.

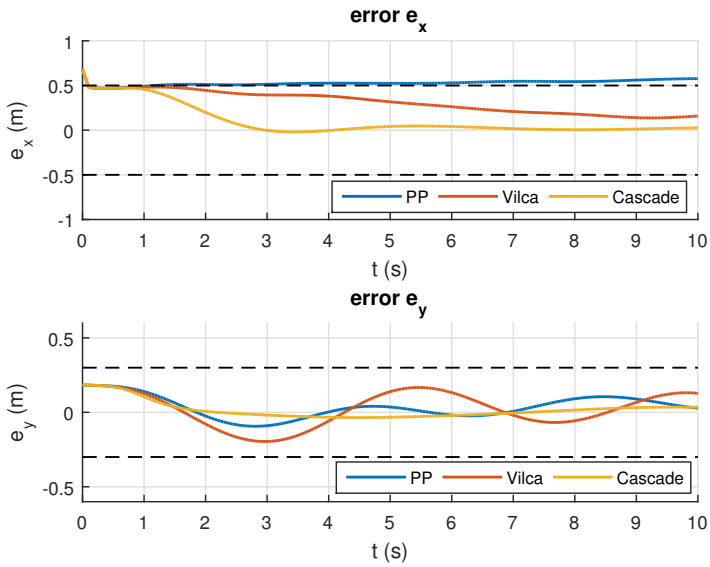

Figure 2.8: Tracking performance (1^{st} series)

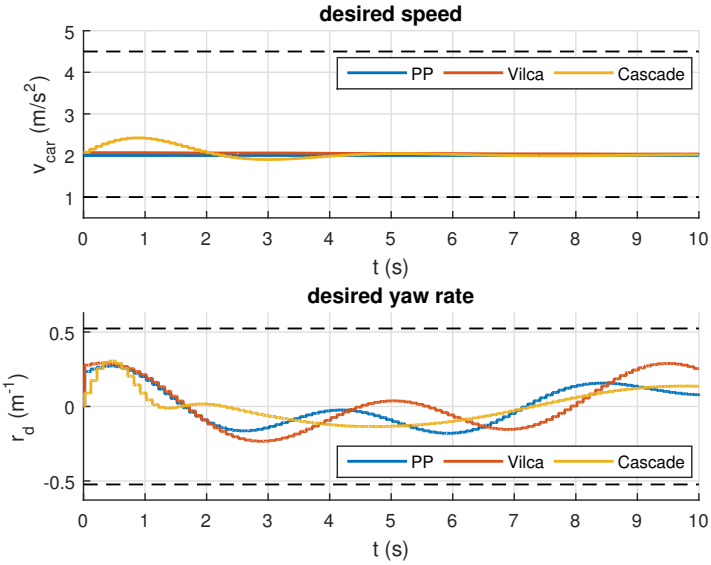

Figure 2.9: Control signals (1^{st} series)

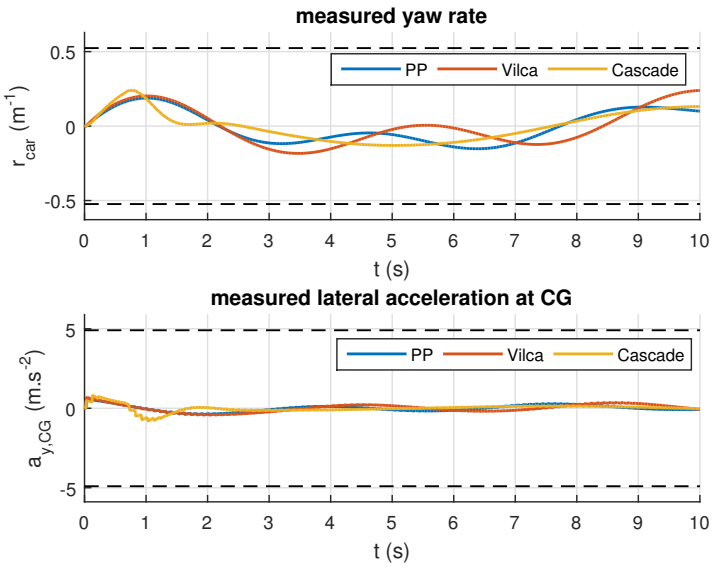

Figure 2.10: Comfort indicators (1^{st} series)

2.4.4 Safety and comfort assessment

In this simulation, the safety has been assessed by checking the tracking performance under a change of mass, friction coefficient and speed. The respect of the comfort constraints with the proposed cascade architecture has been tested and compared with the behaviour of the other controllers. In this series, the target initial position was further from the vehicle initial position (cf. Table 2.4) to study the behaviour of the controllers when a more aggressive maneuver is required to follow the target.

The tracking performance of the designed cascade architecture is now far better than both the Pursuit controller and Vilca's controller (cf. Fig. 2.11). The latter one shows an unstable oscillatory behaviour at these higher speeds because it neither anticipates the trajectory nor the actuator delay. The performance of the cascade architecture also shows the effectiveness of the inner controller to stabilize the yaw dynamics with a non nominal model. The control signals (Fig. 2.12) show an effective capping of both the desired speed and yaw rate with the cascade architecture, leading to a faster tracking errors convergence while respecting the introduced constraints. The comfort indicators (cf. Fig. 2.13) show undamped oscillations for both the Pursuit and Vilca's controller, and a slight overshoot for the cascade design. It could be removed with finer tuning of the MPC controller or the use of the real target's trajectory information.

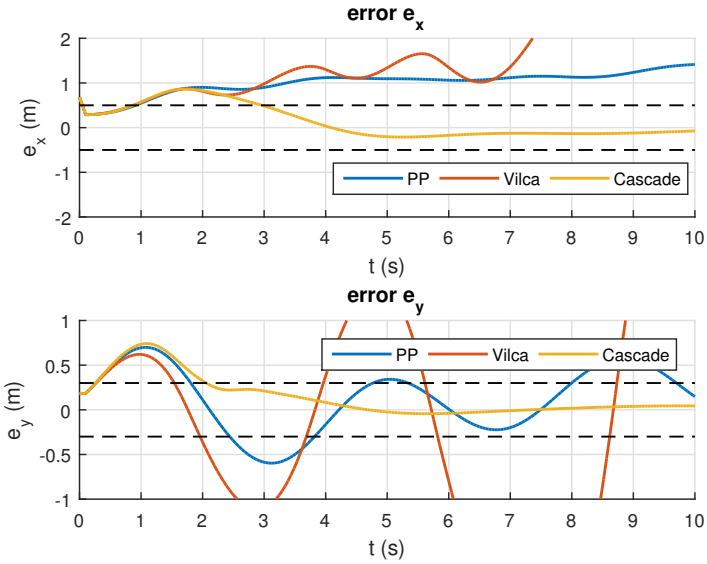

Figure 2.11: Tracking performance (2^{nd} series)

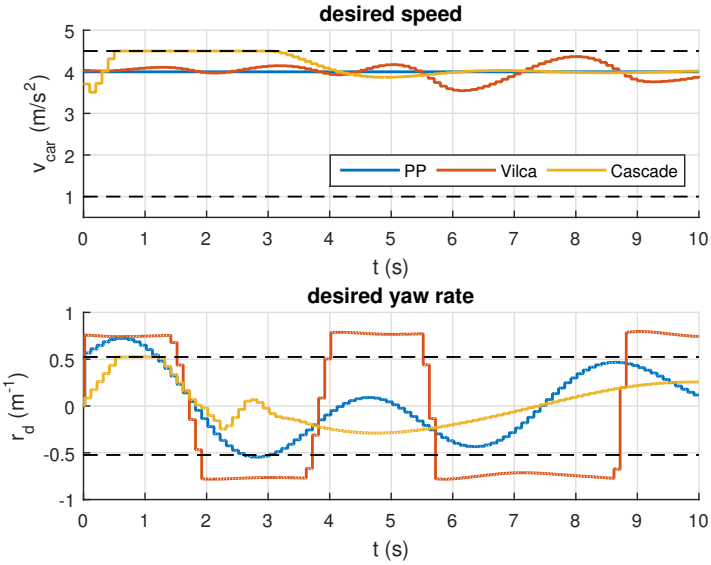

Figure 2.12: Control signals (1^{st} series)

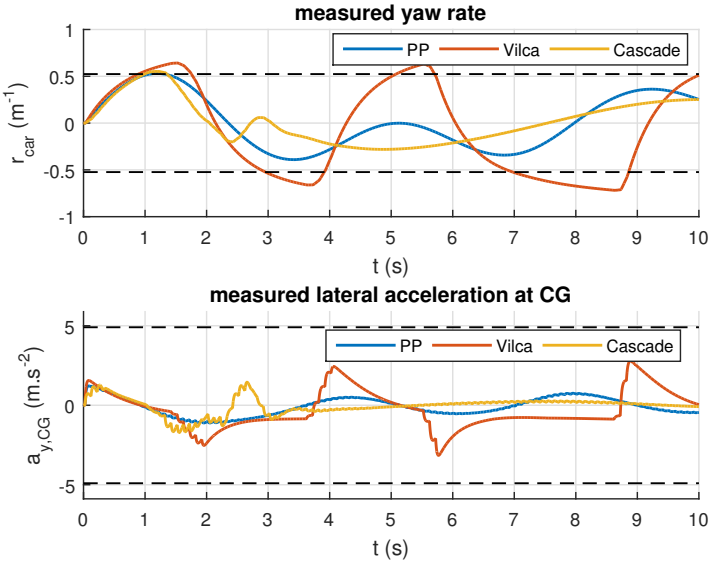

Figure 2.13: Comfort indicators (2^{nd} series)

2.5 Conclusion

In this chapter has been proposed a cascade architecture for autonomous vehicle navigation. This architecture seamlessly fills the tasks of trajectory/path tracking as well as dynamic target following and thus can cope with multi-vehicle scenarios. The architecture is divided into a robust low-level yaw stabilization controller that focuses on the vehicle's dynamics and a high-level tracking MPC controller that focuses mainly on the kinematics. This architecture shows an improvement in tracking performances, safety and flexibility compared to usual kinematic controllers for trajectory tracking. It is not intended to have an edge on performances compared to integrated approaches for trajectory tracking but to improve robustness and implementability. Further work will be carried out to check the performances of a linear MPC controller as well as a robustification of the MPC scheme. Real time implementability will be evaluated and Linear Parameter Varying (LPV) techniques for the low-level controller (parametrized by speed) will be investigated to improve its performance and operational envelope.

References

[1] Danny Yadron and Dan Tynan. "Tesla driver dies in first fatal crash while using autopilot mode". In: *The Guardian* (2016).

[2] José Miguel Vilca, Lounis Adouane, and Youcef Mezouar. “An Overall Control Strategy Based on Target Reaching for the Navigation of an Urban Electric Vehicle”. In: *International Conference on Intelligent Robots and Systems (IROS)* (2013), pp. 728–734.

[3] A. Broggi, P. Medici, P. Zani, A. Coati, and M. Panciroli. *Autonomous vehicles control in the VisLab Intercontinental Autonomous Challenge*. 2012.

[4] Simon Hecker. “Robust Hinf-based vehicle steering control design”. In: *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE*. IEEE, Oct. 2006, pp. 1294–1299.

[5] Paolo Falcone and Francesco Borrelli. “Low complexity MPC schemes for integrated vehicle dynamics control problems”. In: *International Symposium on Advanced Vehicle Control* 1 (2008), pp. 875–880.

[6] Gabriel M. Hoffmann, Claire J. Tomlin, Michael Montemerlo, and Sebastian Thrun. “Autonomous Automobile Trajectory Tracking for Off-Road Driving : Controller Design , Experimental Validation and Racing”. In: *Proceedings of the American Control Conference* (2007), pp. 2296–2301.

[7] Pushkar Hingwe, Han Shue Tan, Andrew K. Packard, and Masayoshi Tomizuka. “Linear parameter varying controller for automated lane guidance: Experimental study on tractor-trailers”. In: *IEEE Transactions on Control Systems Technology* 10.6 (Nov. 2002), pp. 793–806.

[8] Pedro Aguiar, Andrea Alessandretti, and Colin N Jones. “Trajectory-tracking and path-following controllers for constrained underactuated vehicles using Model Predictive Control”. In: *European Control Conference* (2013), pp. 1371–1376.

[9] Joshué Pérez, Vicente Milanés, and Enrique Onieva. “Cascade architecture for lateral control in autonomous vehicles”. In: *IEEE Transactions on Intelligent Transportation Systems* 12.1 (Mar. 2011), pp. 73–82.

[10] Alexey S. Matveev, Hamid Teimoori, and Andrey V. Savkin. “A method for guidance and control of an autonomous vehicle in problems of border patrolling and obstacle avoidance”. In: *Automatica* 47.3 (2011), pp. 515–524.

[11] Dongkyoung Chwa and Jin Young Choi. “Adaptive Nonlinear Guidance Law Considering Control Loop Dynamics”. In: *IEEE Transactions on Aerospace and Electronic Systems* 39.4 (2003), pp. 1134–1143.

[12] Farid Kendoul. “Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems”. In: *Journal of Field Robotics* 29.2 (Mar. 2012), pp. 315–378. arXiv: 10.1.1.91.5767.

[13] Jürgen Ackermann. *Robust Control for Car Steering*. Tech. rep. German Aerospace Center, Institute of Robotics and System Dynamics, 1999.

[14] José Miguel Vilca, Lounis Adouane, and Youcef Mezouar. “A novel safe and flexible control strategy based on target reaching for the navigation of urban vehicles”. In: *Robotics and Autonomous Systems* 70 (Aug. 2015), pp. 215–226.

[15] Jesús Morales, Jorge L. Martínez, María A Martínez, and Anthony Mandow. “Pure-pursuit reactive path tracking for nonholonomic mobile robots with a 2D laser scanner”. In: *Eurasip Journal on Advances in Signal Processing* 2009.1 (2009), p. 935237.

Chapter 3

Safe and Online MPC for Managing Safety and Comfort of Autonomous Vehicles in Urban Environment

Abstract

In this chapter is presented a linear MPC controller design for autonomous cars navigation. It combines both the lateral and longitudinal control. This controller is designed to manage comfort and safety as well as being compliant with specific platooning tasks (such as target following, ACC and collaborative obstacle avoidance) and the switches that can occur between those tasks. The MPC cost function has been designed to account for human driving behaviours, i.e., it smoothes out coarse reference trajectories. Furthermore, a safety monitoring module has been implemented. It computes an estimated time before reaching an unacceptable situation (w.r.t. comfort constraints and tracking performance) under the current tracking conditions. The overall benefit of this controller is to guarantee trajectory smoothness while outputting information on its performance. In terms, this information can be used to re-plan safe trajectories in dynamic environments. The proposed linear MPC controller has been tested in a typical urban scenario based on a realistic simulator.

3.1 Introduction

The field of mobile robotics for passenger transportation has been very active in the recent years. Many advances have been done and highlighted by challenges such as the *DARPA Grand Challenge* and more recently by the *Grand Cooperative Challenge*, as well as advances from companies such as Volvo, Uber, Google or Tesla. Recent events have highlighted the necessity to ensure very high levels of reliability in the algorithmic developments to deal with uncertainties in the environment as well as foreseeing and preventing problems that could arise from sensors and actuators. This chapter brings a contribution to solving these problems for trajectory tracking.

Model Predictive Control (MPC) is now a widely used and performant technique for optimal trajectory tracking [1], [2] and/or optimal trajectory generation [3]. It allows to cater for future events and the predictive nature often helps generating smoother control signals when the MPC is used as a controller.

For all the above mentioned applications, a linear MPC formulation has to be used, mainly for computational complexity reasons. This formulation introduces limitations in the way the problem can be formulated, since it ultimately has to come down to a Quadratic Programming (QP) solving. As a consequence, the cost functions used in the nonlinear formulations cannot always be transformed into a corresponding linear formulation for real time implementation.

Other interesting works have moved towards human-like behaviour, such as in [4] for a plannning algorithm, showing good results. Such an approach potentially allows to limit the difference between humans and machines in terms of perceived behaviour and could ultimately allow the passengers to feel less uncomfortable. The approach shown in this chapter is to include elements of "smoothness" in the trajectory tracking algorithm (mimicking human behaviour) while ensuring safety and precision. Guaranteeing smoothness of the car trajectory at the controller level additionally means it is not needed to tackle this problem when designing the trajectory planner. Thus, some complexity is removed from the planner and other elements can be taken into account, such as risk minimization.

Some recent planning algorithms have introduced such functions to assess the performance of the subsequent trajectory controller in order to compute a maximal safe speed (c.f. [5] for an offroad application). It has the benefits of ensuring a given tracking performance. The main risks of failure to guarantee tracking performances are actuator saturation and tracking precision (due mainly to disturbances and model inacurracies).

This chapter follows up a first contribution for an architecture design including guidance and control presented in [6]. It goes further by linearizing the MPC algorithm, proposing a safety monitoring module and testing it in a realistic simulation environment using the ROS middleware.

The approach proposed in this chapter focuses on improving the attainable tradeoff between smoothness of the control signal and tracking performance. This objective has been reached using a linear MPC formulation that gives a smoother control signal than traditional linear MPC controllers, and a risk monitoring module.

The remainder of this chapter is organized as follows. Section 3.2 explains the reasons behind the architecture and controller choice, and describes the design of the controller. The characteristics and performance of the proposed controller are assessed in section 3.3 through experiments on a realistic simulation software. A conclusion and prospects are given in section 3.4.

3.2 Proposed Architecture

3.2.1 Architecture design

The proposed MPC algorithm is intended to be part of a common Guidance, Navigation and Control (GNC) architecture. The navigation layer is not detailed here since it is not the focus of the presented works. The aim is to propose a versatile architecture for navigation of urban vehicles. This architecture should ease the support of convoy navigation and ensure safety. The proposed MPC algorithm is designed as a trajectory controller for both the lateral and longitudinal axis. The lateral control signal generated by the MPC feeds a low-level yaw-rate controller that controls the car steering. Such a low level controller has been shown in [6]. The MPC controller is coupled with a safety monitoring algorithm that computes a risk associated to the current tracking situation: reference trajectory, desired speed and predicted performance of the MPC controller. This risk is computed based on the following criterions:

- degree of MPC model accuracy
- predicted lateral error

These two criteria give an indication of the tracking performance that can be expected. they can also be used to obtain an estimation of the likelihood of failing to track accurately

the reference trajectory. The general architecture is shown in Fig. 3.1.

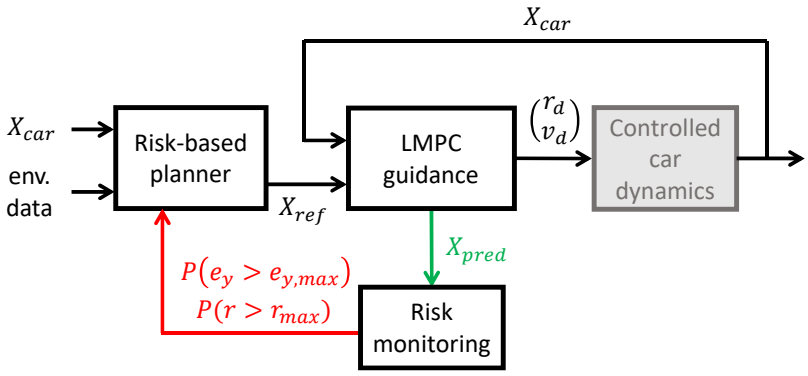

Figure 3.1: Planning and risk monitoring architecture

3.2.2 MPC controller

A linear MPC controller has been designed. In an earlier publication [6], a nonlinear MPC controller has been presented and tested in simulations against a widely used trajectory controller [7] and a versatile Lyapunov function based nonlinear control law [8] that supports trajectory tracking, mobile target tracking and waypoint-based navigation. This controller showed a significant increase in terms of tracking performance while keeping respective qualities of the two other control laws. However, its computational complexity is too much for real-time applications and the non-convex optimization due to the non linearity leads to local minima whose qualities are difficult to evaluate.

In this chapter, the MPC controller has been linearized around a reference trajectory [1]. This reference trajectory is time-variable and is the output of the local planning algorithm. Such a linearization allows to overcome the two common drawbacks of nonlinear MPC schemes (computational burden and non-convexity of the problem). The linear formulation presented here can run in real-time on commercial grade computers. To keep the same behaviour as the nonlinear MPC algorithm given in [6], a formulation has been derived to introduce a penalty on the control signal derivative.

The continuous model $\dot{X} = f(X)$ for the MPC algorithm is shown in equation (3.1). A second order model has been chosen for the yaw rate. This model represents the inner loop as designed in [6]. A first order model has been kept for the longitudinal control as it gives satisfactory accuracy in our application. The state is defined as $X = (x, y, \psi, r, v, \dot{r})$ and the input $U = (r_d, v_d)$. In this chapter, r denotes a yaw rate, ψ a heading, v a linear

speed.

$$f(X) = \begin{pmatrix} v\cos\psi \\ v\sin\psi \\ r \\ \dot{r} \\ \tau_v^{-1}(v_d - v) \\ -2\zeta_r\omega_r\dot{r} - \omega_r^2(r - G_r r_d) \end{pmatrix} \tag{3.1}$$

Where τ_v is the first order time constant of the longitudinal actuator model, ζ_r is the damping of the 2^{nd} order lateral actuator model, ω_r the pulsation and G_r the gain. The number of states is $n = 6$ and the number of inputs is $p = 2$. The model conventions are shown in Fig. 3.2.

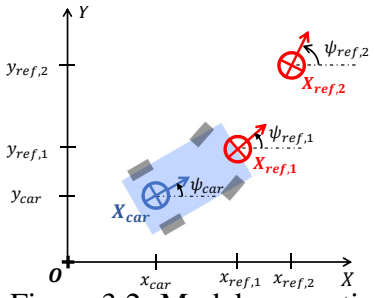

Figure 3.2: Model conventions

The series of reference points $X_{ref,i}$ in Fig. 3.2 define a reference trajectory to track and is generated by a trajectory planner. Each $X_{ref,i}$ has the same dimension than the model state X.

A discrete model is derived from the continuous one with a first order Taylor series around the linearization point (X_{ref}, U_{ref}) with sampling time T_s:

$$\tilde{X}(k+1) = A_c(k, T_s)\tilde{X}(k) + B_c(k, T_s)\tilde{U}(k) \tag{3.2}$$

Where U_{ref} is a reference input that allows to follow perfectly the reference states $X_{ref,i}$. The difference vectors are defined as $\tilde{X} = X - X_{ref}$ and $\tilde{U} = U - U_{ref}$. The reference input U_{ref} is computed through an "inversion" of the model between successive reference states. For example, between two reference states $X_{ref,0}$ and $X_{ref,1}$, the reference input would be ideally defined by the perfect following equality:

$$X_{ref,1} = A_c(0, T_s)X_{ref,0} - B_c(0, T_s) * U_{ref} \tag{3.3}$$

As the input matrix $B_c(0, T_s)$ is not always invertible, the reference input has been com-

puted by means of a least square approximation.

The equation (3.2) can be vectorized over the MPC prediction horizon of size N_{MPC}:

$$\tilde{\mathbf{X}}(k+1) = \mathbf{A_c}(k)\tilde{X}(k|k) + \mathbf{B}_c(k)\tilde{\mathbf{U}}(k) \tag{3.4}$$

Where $\tilde{\mathbf{X}}(k+1) = (\tilde{X}(k+1),\ \ldots\ , \tilde{X}(k+N_{MPC}))^T$ and $\tilde{\mathbf{U}}(k) = (\tilde{U}(k),\ \ldots\ , \tilde{U}(k+N_{MPC}-1))^T$

The computation of the model matrices $\mathbf{A_c}(k)$ and $B_c(k)$ is explained in [1]. They depend on the non vectorized model matrices $A_c(k+i, T_s)$ and $B_c(k+i, T_s)$ at the different timesteps i in the MPC horizon. To mitigate the discretization errors without increasing the control sampling rate, the model matrices have been computed at an upsampled rate T_{up}. The MPC horizon size is denoted N_{MPC}. The model sampling time is T_{up}, and the upsampling factor $K_{up} = T_s/T_{up}$. The upsampled model matrices are then defined as:

$$A_{c,up}(k) = A_c(k, T_{up})^{K_{up}}$$

$$B_{c,up}(k) = \left(\sum_{i=0}^{K_{up}-1} A_c(k, T_{up})^i \right) B_c(k, T_{up})$$

These matrices correspond to the propagation K_{up} times of a model sampled at T_{up} around the linearization point k.

The optimization problem can be expressed as the usual quadratic optimization problem with the cost function $J(k)$ defined as:

$$J(k) = \frac{1}{2}\tilde{\mathbf{U}}(k)^T\mathbf{H}(k)\tilde{\mathbf{U}}(k) + \mathbf{f}^T\tilde{\mathbf{U}}(k) \tag{3.5}$$

With:

$$\begin{aligned} \mathbf{H}(k) &= 2\left(\mathbf{B_c}(k)^T\mathbf{Q}\mathbf{B_c}(k) + \mathbf{R}\right) \\ \mathbf{f}(k) &= 2\mathbf{B_c}(k)^T\mathbf{Q}\mathbf{A_c}(k)\tilde{X}(k) \end{aligned} \tag{3.6}$$

In this formulation, the current difference between the state and the first reference $\tilde{X}(k)$ needs to be fully measurable to propagate the model.

The matrix $\mathbf{Q} = \text{diag}(Q_1, ..., Q_{N_{MPC}})$ is used for weighting the penalty of the state error $\tilde{X}$ in the cost function. Each Q_i is a n-by-n square matrix containing the weights for each state. In this application, these weights have been kept constant over the MPC horizon.

Thus, each Q_i is defined as:

$$Q_i = (Q_{11}, Q_{22}, \ \dots \ , Q_{nn}) \tag{3.7}$$

The matrix $\mathbf{R}$ weights the error to the reference input series $\mathbf{U}_{ref}$. $\mathbf{R}$ is built the same way as the matrix $\mathbf{Q}$: it is a diagonal matrix of N_{MPC} square blocks $R_i = \text{diag}(R_{11}, R_{22})$.

An augmentation of the cost function is proposed in this chapter to include terms on the control signal derivatives (which could easily be translated into cost on the states derivatives). This term is often used in nonlinear MPC algorithms to smooth the control signal. In this application, it is also used as a way to make the MPC algorithm less sensitive to noise in the reference trajectory. Such a noisy reference can happen because of noisy localization data or when following a vehicle.

In this formulation, the derivatives are computed by means of finite differences. Let $\mathbf{U}$ be the input series to impose the weight on, $\mathbf{U}_{ref}$ the reference input and $\tilde{\mathbf{U}} = \mathbf{U} - \mathbf{U}_{ref}$.

The subscript Δ denotes backwards differences operation. Hence $\mathbf{U}_\Delta$ is the vector of backwards differences of the terms in $\mathbf{U}$.

For vectorized notations, the subscript 0 denotes the terms for the first $N_{MPC} - 1$ timesteps in an input vector. The subscript 1 denotes the terms for the last $N_{MPC} - 1$ timesteps. Hence, $\mathbf{U}_\Delta = \mathbf{U}_1 - \mathbf{U}_0$.

The aim is to find two matrices $\mathbf{H}_d$ and $\mathbf{f}_d$ so that:

$$\|\mathbf{U}_\Delta\|^2 = \tilde{\mathbf{U}}^T \mathbf{H}_d \tilde{\mathbf{U}} + \mathbf{f}_d^T \tilde{\mathbf{U}} \tag{3.8}$$

As a consequence, the matrices $\mathbf{H}_d$ and $\mathbf{f}_d$ (after weighting) can be added to $\mathbf{H}(k)$ and $\mathbf{f}(k)$ in the quadratic problem.

Developing the expression of $\|\mathbf{U}_\Delta\|^2$ yields:

$$\begin{aligned} \|\mathbf{U}_\Delta\|^2 \ = \ & \left\|\tilde{\mathbf{U}}_1 - \tilde{\mathbf{U}}_0\right\|^2 \\ & + 2\left(\mathbf{U}_{ref,1} - \mathbf{U}_{ref,0}\right).\tilde{\mathbf{U}}_1 \\ & - 2\left(\mathbf{U}_{ref,1} - \mathbf{U}_{ref,0}\right).\tilde{\mathbf{U}}_0 \\ & + \left\|\mathbf{U}_{ref,1} - \mathbf{U}_{ref,0}\right\|^2 \end{aligned} \tag{3.9}$$

The quadratic and linear terms in $\tilde{\mathbf{U}}$ are clearly separated. Hence, the following identifi-

cation (because of the unicity of a polynomial coordinates):

$$\tilde{\mathbf{U}}^T \mathbf{H}_d \tilde{\mathbf{U}} = \left\| \tilde{\mathbf{U}}_1 - \tilde{\mathbf{U}}_0 \right\|^2 \tag{3.10}$$

and:

$$\mathbf{f}_d^T \tilde{\mathbf{U}} = 2\left(\mathbf{U}_{ref,1} - \mathbf{U}_{ref,0}\right).\tilde{\mathbf{U}}_1 - 2\left(\mathbf{U}_{ref,1} - \mathbf{U}_{ref,0}\right).\tilde{\mathbf{U}}_0 \tag{3.11}$$

Developing $\|\mathbf{U}_\Delta\|^2$ in equation (3.8) and identifying the quadratic and linear terms yields the following definition for $\mathbf{H}_d$:

$$\mathbf{H}_d = \begin{pmatrix} \mathbf{I_p} & -\mathbf{I_p} & & & \mathbf{0} \\ -\mathbf{I_p} & 2\mathbf{I_p} & -\mathbf{I_p} & & \\ & \ddots & \ddots & \ddots & \\ & & -\mathbf{I_p} & 2\mathbf{I_p} & -\mathbf{I_p} \\ \mathbf{0} & & & -\mathbf{I_p} & \mathbf{I_p} \end{pmatrix} \tag{3.12}$$

Where $\mathbf{H}_d$ is a square matrix of size N_{MPC}-by-p (N_{MPC} is the size of the MPC horizon and p denotes the number of model inputs). The matrix $\mathbf{I_p}$ is the identity matrix of size p. The vector $\mathbf{f}_d$ is a column vector of dimension N_{MPC} defined as:

$$\mathbf{f}_d = 2\begin{pmatrix} \mathbf{U}_{ref,0} - \mathbf{U}_{ref,1} \\ 0 \end{pmatrix} + 2\begin{pmatrix} 0 \\ \mathbf{U}_{ref,1} - \mathbf{U}_{ref,0} \end{pmatrix} \tag{3.13}$$

With each block in the vector having p rows.

The new matrices for the optimization are finally obtained as follows:

$$\begin{aligned} \mathbf{H}(k) &= 2\left(\mathbf{B_c}(k)^T \mathbf{Q} \mathbf{B_c}(k) + \mathbf{R} + \mathbf{R}_d \mathbf{H}_d\right) \\ \mathbf{f}(k) &= 2\left(\mathbf{B_c}(k)^T \mathbf{Q} \mathbf{A_c}(k) \tilde{X}(k) + \mathbf{R}_d \mathbf{f}_d\right) \end{aligned}$$

Where $\mathbf{R}_d$ is used to weight the penalty on the control signal derivative. It is built in the same way as $\mathbf{R}$. The scalar weight for the lateral input derivative is denoted $R_{d,11}$ and for the longitudinal input $R_{d,22}$.

This formulation introduces a trade-off between the tracking performance and the passenger comfort in the usual MPC formulation (in favor of the smoothness of the control signal). Usually, the weight $\mathbf{R}$ is used to tune the convergence rate of MPC controllers. The main drawback is that it penalizes the difference between U and U_{ref}: this difference can be noisy due to the computation of U_{ref}. It arises because the model inversion that

is used to get U_{ref} from X_{ref} tends to amplify noise on U_{ref}. If the aim is to be able to give coarse reference paths to the MPC, the convergence should not be set by increasing $\mathbf{R}$.

Because of this new weight $\mathbf{R}_d$ on the control signal derivative in the penalty function, no hard constraints needs to be introduced for comfort features. Instead, comfort requirements will be dealt with the trajectory planner, in order to keep even aggressive emergency trajectories controllable by the MPC algorithm.

However, a constraint on the maximal required curvature has to be implemented to consider the vehicle's geometry and mechanical constraints. With the linear MPC formulation, it has to be implemented through an approximation. From the variables in $U = (r_d, v_d)^T$, the desired curvature is $c = r_d/v_d$ which is not a linear relation with respect to U. As a consequence, the curvature is expressed approximately with $\bar{c} = r_d/v_{ref}$ around the linearization trajectory. This approximation holds perfectly as long as the vehicle's speed is close to the reference speed.

The constraint is then expressed as:

$$\begin{cases} \bar{c} = r/v_{ref} & \leq & c_{max} \\ \bar{c} = r/v_{ref} & \geq & -c_{max} \end{cases} \tag{3.14}$$

Subtracting $c_{ref} = r_{ref}/v_{ref}$ on each side and rearranging gives:

$$\begin{cases} \dfrac{1}{v_{ref}}\tilde{r} & \leq & c_{max} - c_{ref} \\ -\dfrac{1}{v_{ref}}\tilde{r} & \leq & c_{max} + c_{ref} \end{cases} \tag{3.15}$$

With $\tilde{r} = r - r_{ref}$.

To express these two constraints in the form $D\tilde{U} \leq d$, the following 2-dimensional matrices are used:

$$\begin{aligned} D &= \begin{pmatrix} v_{ref}^{-1} & 0 \\ -v_{ref}^{-1} & 0 \end{pmatrix}^t \\ d &= \begin{pmatrix} c_{max} - c_{ref} \\ c_{max} + c_{ref} \end{pmatrix} \end{aligned} \tag{3.16}$$

They can be generalized for the input $\tilde{\mathbf{U}}$ over the MPC horizon.

Additionally, it has been chosen not to implement tracking constraints on the vehicle's

state as they can destabilize the optimization if being impossible to fulfill. In that case, a linear quadratic solver may prefer to violate constraints on the input (maximum speed or yaw rate) and give completely unacceptable solutions. A monitoring solution has been preferred, in order to introduce a form of feedback on the tracking performance. This is detailed in the next section.

3.2.3 MPC behaviour monitoring

As tracking performance should never be compromised for safe trajectory tracking, a behaviour monitoring module for the MPC algorithm has been developed. This module uses the knowledge of the MPC controller performance to generate a risk estimation associated to what is currently asked to the MPC controller. It takes advantage of the predictive nature of the MPC. This information is intended to be used in a trajectory planner in order to find acceptable trajectories risk-wise.

The safety monitoring module has two subfunctions in order to assess:

- The risk of violating the comfort constraints
- The risk of lateral tracking constraint violation

In this application, comfort constraints are defined as state constraints on the yaw rate and yaw rate derivative. The risk of violating the comfort constraints is evaluated as a critical time $t_{c,comfort}$ at which those constraints would be violated by tracking the current trajectory. The optimal input found $\mathbf{U_{opt}}$ is used to propagate the states of the vehicle over the MPC horizon. The predicted states are then checked for violation yaw rate constraints, and of derivative constraints. If such a point is found at the critical index $i_{c,comfort}$, then:

$$t_{c,comfort} = T_s i_{critical} \tag{3.17}$$

For tracking accuracy monitoring, an estimation of the uncertainty of the MPC predicted lateral error $\hat{e}_y$ is first carried out. At each time step k, the N_{MPC} past inputs are applied from the vehicle's position N_{MPC} time steps ago. The lateral error predictions $\left[\hat{e}_y(k-N_{MPC}), ..., \hat{e}_y(k)\right]$ obtained are compared with the known lateral errors up to the present time: $\left[e_y(k-N_{MPC}), ..., e_y(k)\right]$. At the time k, it gives the errors associated to the prediction of e_y:

$$\tilde{e}_y(i,k) = \hat{e}_y(k-i) - e_y(k-i)\ (\forall i \in [1, N_{MPC}]) \tag{3.18}$$

These errors are due to modelling inaccuracies and measurement errors. They are compiled in a matrix $\mathbf{E}_{y,model}$ of size (N_{MPC}, N_{mem}) over a finite number of timesteps N_{mem}. A column j of $\mathbf{E}_{y,model}$ contains the model errors for each step in the MPC horizon computed j timesteps ago.

The uncertainty associated to $\hat{e}_y$ is computed from the data in $\mathbf{E}_{y,model}$. Assuming an unbiased normal distribution for each $\hat{e}_y(i)$, its uncertainty is computed as the standard deviation (denoted $\sigma_{e_y}(i)$) of the prediction errors for each step i in the MPC horizon:

$$\sigma_{e_y}(i) = \sqrt{\frac{1}{N_{MPC}-1}\sum_{j=1}^{N_{MPC}} \mathbf{E}_{y,model}(i,j)^2} \tag{3.19}$$

The bigger the memory time N_{mem}, the finer the estimation of the standard deviation, but the more lag it has. This can be problematic if the model error is dependant on external parameters such as the curvature of the road. The main risk is to underestimate the standard deviation of $\hat{e}_y$, and as a consequence to trust the model too much. A short horizon of $N_{mem} = 5$ has been chosen.

Thanks to the information on e_y uncertainty, a probabilistic risk prediction for the tracking error violation can be carried out. The result of this prediction will be a critical time $t_{c,\hat{e}_y}$ within the MPC horizon at which the probability of overshooting the lateral tracking constraint is above a given threshold probability P_c (typically 5%):

$$\begin{cases} i_{c,\hat{e}_y} = \min\limits_{i\in[1,N_{MPC}]} \left(i, \text{ so that } P\left(\hat{e}_y(i) \geq e_{y,max}\right) \geq P_c\right) \\ t_{c,\hat{e}_y} = T_s i_{c,\hat{e}_y} \end{cases} \tag{3.20}$$

It is equivalent as looking for the first i for which the required confidence interval of $\hat{e}_y$ is not included in $[-e_{y,max}, e_{y,max}]$.
For instance, the 95% confidence interval of $\hat{e}_y(i)$ ($P_c = 5\%$) is defined as:

$$I_{c,95\%} = [\hat{e}_y(i) - 2\sigma_{e_y}(i), \hat{e}_y(i) + 2\sigma_{e_y}(i)]$$

Thus, the index looked for is the first i so that:

$$|\hat{e}_y(i)| + 2\sigma_{e_y}(i) \geq e_{y,max} \tag{3.21}$$

This relation can be generalized for any threshold probability P_c, knowing the confidence intervals of the normal distribution of standard deviation $\sigma_{e_y}(i)$. This way of computing

a critical time is more conservative than using $\hat{e}_y$ without considering its uncertainty. Its aim is to allow earlier notice of risky situations and avoid false negatives.

The two indexes developed in this section provide in-depth and clearly understandable information about different kind of risks linked to the trajectory tracking performance. Their additional computational cost is very low because all the information used comes from the MPC algorithm, which makes them interesting as a systematic probabilistic risk-monitoring approach for MPC applications. The risk evaluation defined in this section will be assessed in section 3.3.2.

3.3 Experiments

The simulations shown in this section have been performed with the ROS framework and the 4D-Virtualiz simulation engine[1]. A screenshot of the environment is shown in Fig. 3.3. This simulation environment provides a realistic physical model for the vehicle and actuators as well as integration within the ROS framework. The vehicle used in simulation is an *IPCar*, an electric urban vehicle of maximal speed $3m/s$ and minimum turning radius $R_{min} \approx 3m$. The roads used in simulation are an exact copy of the *Plateforme d'Auvergne pour les Véhicules Intelligents*[2] (PAVIN). The PAVIN is half scale test track representing a neighbourhood, with red lights and a roundabout. Thus, realistic urban driving situations can be mimicked at lower speeds with narrower roads and tighter turns. All the vehicle's sensors have been modelled as on the real ones: Real-Time Kinematics GPS (RTK-GPS), LIDAR, odometry sensors on the back wheels, angle sensors on the front wheels. The yaw acceleration is not measured and thus is simply computed by means of finite differences.

For the experiments presented here, the GPS data is perfect and has been sampled at $F_s = 10Hz$. The odometry sensors on the back wheels give a noisy estimation of the yaw rate and linear speed.

3.3.1 MPC behaviour analysis

The behaviour of the MPC controller has been assessed against its main tuning coefficients for lateral dynamics R_{11} and the term introduced in this chapter $R_{d,11}$. The other

[1] http://www.4d-virtualiz.com/. A demonstration video has been joined to the chapter submission.
[2] IPDS, "http://ipds.univ-bpclermont.fr," Dec. 2017, The Institut Pascal Data Sets.

Figure 3.3: 4D-Virtualiz simulation environment

tuning term for lateral dynamics Q_{11} has been kept constant, since multiplying all the coefficients by a scalar ultimately does not change the "shape" of the cost function. The aim of this section is to show that the adaptation of the weight on the control signal derivative to the linear case makes sense and allows to tune the MPC controller so that it gives a smooth trajectory even in the presence of noisy perception/reference trajectories. The path followed is a 90^o left hand turn of minimum radius of curvature $R_c \approx 4m$. Its characteristics are shown in shown in Fig. 3.4.

In this section, the weights for the state error on the x-axis and the y-axis have been kept constant at $Q_{11} = Q_{22} = 1$. The reference trajectory is followed at a constant speed of $v_{ref} = 2$m/s. Thus, only the lateral control signal has been plot. For the plots, the measured yaw rate data has been smoothed *a posteriori* with a moving average of 5 samples.

The first experiment shows the effect of the introduced term $R_{d,11}$ when following a smooth reference trajectory. The second experiment shows the effect of $R_{d,11}$ when following a noisy reference trajectory. The default MPC controller parameters have been summarized in table 3.1. This table also contains actuator parameters, which have been identified with a least-squares method with data from the simulator.

In Fig. 3.5, the curve for $R_{d,11} = 0$ corresponds to a classic linear MPC formulation. The increase coefficient $R_{d,11}$ shows very little improvement when following the smooth reference path. Even though the control signal r_d is smoother, the only noisy event at $t = 4s$ present in r_d in the classic case has no influence on the measured yaw rate and does not introduce any oscillations.

However, the introduced weight on the control signal derivative shows a beneficial effect under a noisy reference trajectory in Fig. 3.6. it shows an improvement in the control signal smoothness and in the measured yaw rate, while not degrading the tracking performance. It also helps preventing the noise on the control signal r_d that is picked by the

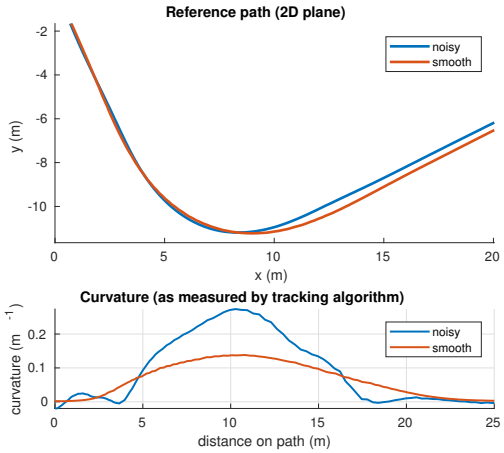

Figure 3.4: Reference trajectory for MPC coefficients assessment

Table 3.1: Parameter values for MPC behaviour analysis

parameter	**notation**	**value**
reference speed	v_{ref}	2 m/s
loop rate	F_s	10 Hz
MPC horizon	N	15
weight on longitudinal error	Q_{22}	1
default weight on lateral ref. input	R_{11}	3
weight on longitudinal ref input	R_{22}	5
lateral actuator gain	G_r	0.92
lateral actuator pulsation	ω_r	6.9 rad/s
lateral actuator damping	ζ_r	0.7
longitudinal actuator time constant	τ_v	0.5

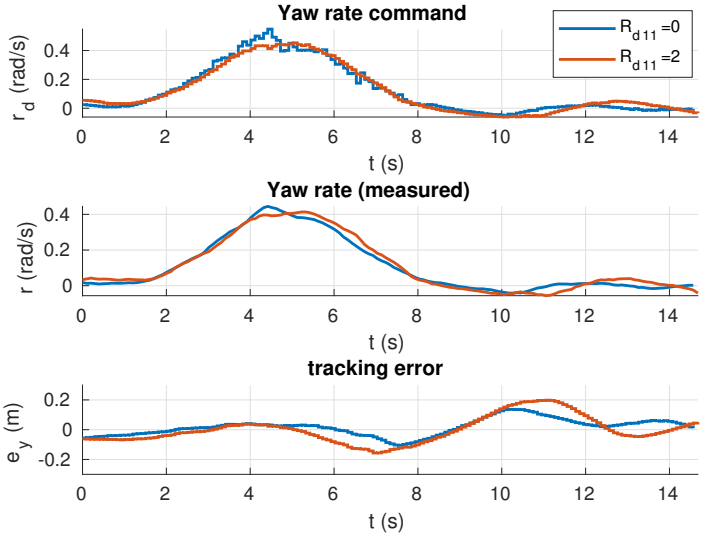

Figure 3.5: Effect of $R_{d,11}$ coefficient, smooth reference trajectory

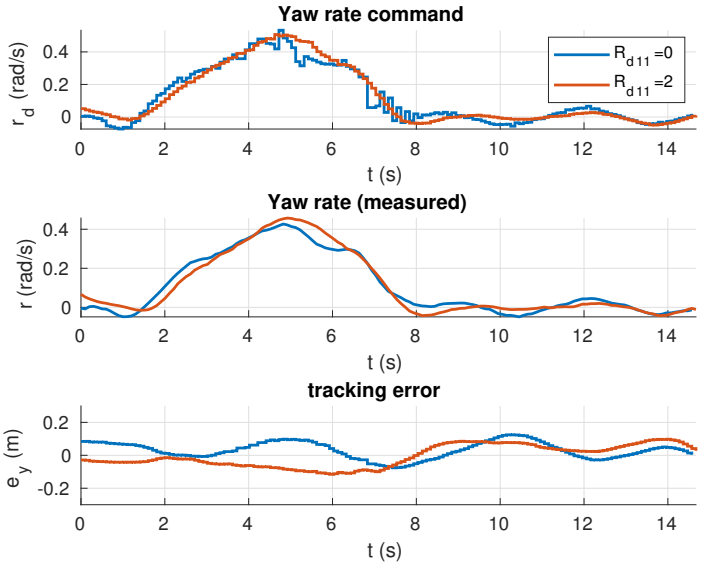

Figure 3.6: Effect of $R_{d,11}$ coefficient, noisy reference trajectory

classic formulation, and successfully smoothes the turn exit. Furthermore, it prevents a constant oscillatory behaviour of the classic controller which is not smoothed out by the car dynamics (as seen on the two upper plots of Fig. 3.6).

In conclusion, the weight on the input derivative allows to find a better tradeoff between tracking accuracy and control signal smoothness by acting as a filter for noisy reference signals and inaccuracies in the model. Ultimately, it makes the MPC controller more confortable and robust. It also has the added benefit of not needing the higher order derivatives to be in the model. Those higher order state derivatives are often noisy and difficult to estimate.

3.3.2 Risk monitoring

For the experiments with the risk monitoring module, the noisy reference trajectory has been used.

A limit value of $e_{y,max}$ has been set for the lateral tracking error. A higher speed of $2.5m/s$ has also been used, which is close to the maximum speed of the *IPCar*.

Two simulations have been run, to show the interest of assessing the risk as presented in the article. In the first one, the reference speed of $2.5m/s$ is carried out through the whole maneuver, independently of the risk prediction. In the second one, the reference speed has been set to go down to $2m/s$ as soon as $t_{c,\hat{e}_y} < 0.5s$. It is thus a very simple form of

speed replanning depending on the risk estimation. Results are shown in Fig. 3.7.

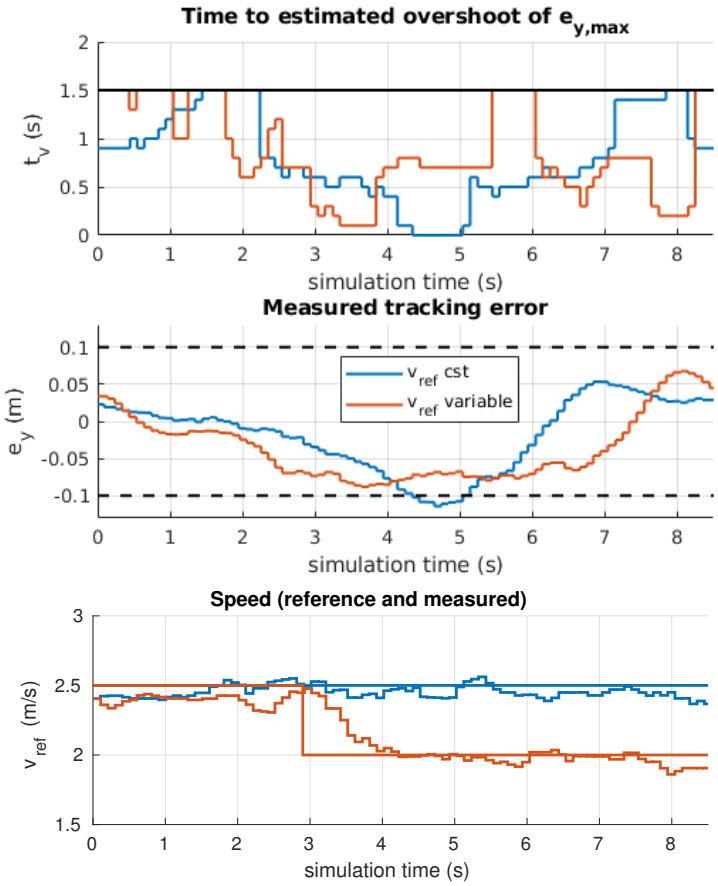

Figure 3.7: Lateral tracking risk prediction and speed replanning

Fig. 3.7 shows the results for the lateral tracking risk estimator. The blue line corresponds to the simulation where no speed replanning is applied, and the orange one to the simulation where the speed is reduced if $t_{c,\hat{e}_y} < 0.5s$. The top plot shows the critical time $t_{c,\hat{e}_y}$ based on a 5% risk of violating the lateral tracking constraint. At $t = 2.9s$, the replanning happens for the second simulation, as seen in the bottom plot. As a consequence, it effectively manages to prevent the overshooting of the lateral error constraint (dotted line on the middle plot). When no speed replanning happens, the tracking constraint ends up being violated. A more elaborate path planner could re-plan the speed in a smoother way, instead of the step that has been imposed on the reference speed in the second simulation.

3.4 Conclusion

In this chapter, a control architecture for risk and comfort management for an urban vehicle has been proposed. This architecture solves the problem of trajectory tracking for ground vehicles while being robust to noise on the reference trajectory. A safety monitoring module has been added in order to have a probabilistic way of assessing the risk linked to the trajectory tracking (defined as the violation of constraints). This safety management has been shown to be able to foresee future dangerous situations and prevent them.

Thanks to these developments, further work on planning and collaborative intersection management will be made easier. With the proposed MPC controller, such algorithms would not have to deal with smoothness of the trajectory and thus can have a lower complexity. The freed computing power can then be used for probabilistic risk-based approaches that are the continuity of the approach hereby developed. For instance, risk estimation could be used in platooning: when following a leader, the following vehicles could notify the leader it makes them take too high risks to keep the formation.

References

[1] F Künhe, J Gomes, and W Fetter. "Mobile Robot Trajectory Tracking Using Model Predictive Control". In: *II IEEE latin-american robotics* (2005), pp. 1–7.

[2] Guilherme V. Raffo, Guilherme K. Gomes, Julio E. Normey-Rico, Christian R. Kelber, and Leandro B. Becker. "A predictive controller for autonomous vehicle path tracking". In: *IEEE Transactions on Intelligent Transportation Systems* 10.1 (Mar. 2009), pp. 92–102.

[3] Benjamin Gutjahr, Lutz Groll, and Moritz Werling. "Lateral Vehicle Trajectory Optimization Using Constrained Linear Time-Varying MPC". In: *IEEE Transactions on Intelligent Transportation Systems* (2016), pp. 1–10.

[4] Tianyu Gu and John M. Dolan. "Toward human-like motion planning in urban environments". In: *2014 IEEE Intelligent Vehicles Symposium Proceedings*. IEEE, June 2014, pp. 350–355.

[5] Jean Baptiste Braconnier, Roland Lenain, and Benoit Thuilot. “Ensuring path tracking stability of mobile robots in harsh conditions: An adaptive and predictive velocity control”. In: *Proceedings - IEEE International Conference on Robotics and Automation*. IEEE, May 2014, pp. 5268–5273.

[6] Charles Philippe, Lounis Adouane, Benoit Thuilot, Antonios Tsourdos, and Hyo Sang Shin. “Risk and comfort management for multi-vehicle navigation using a flexible and robust cascade control architecture”. In: *2017 European Conference on Mobile Robots, ECMR 2017*. IEEE, Sept. 2017, pp. 1–7.

[7] Jesús Morales, Jorge L. Martínez, María A Martínez, and Anthony Mandow. “Pure-pursuit reactive path tracking for nonholonomic mobile robots with a 2D laser scanner”. In: *Eurasip Journal on Advances in Signal Processing* 2009.1 (2009), p. 935237.

[8] José Miguel Vilca, Lounis Adouane, and Youcef Mezouar. “A novel safe and flexible control strategy based on target reaching for the navigation of urban vehicles”. In: *Robotics and Autonomous Systems* 70 (Aug. 2015), pp. 215–226.

Chapter 4

Probability Collectives Algorithm applied to Decentralized Intersection Coordination for Connected Autonomous Vehicles

Abstract

In this chapter, a multi-agent probabilistic optimization algorithm is applied to the problem of multi-vehicle coordination. The algorithm is known as "Probability Collectives" (PC) and has roots in Game Theory and Optimization theory. It is traditionally used for finding optimal solutions of NP-hard problems such as the travelling salesman problem. On the other end, the proposed PC formulation presented in this chapter focuses on a minimal complexity implementation for solving the coordination problem in a time of the order of magnitude of $0.1s$. Besides time constraints, the emphasis in the design is put on ensuring that the algorithm always comes up with a feasible solution. Simulations show that both objectives are reached while having a decentralized algorithm, and flexible with respect to the type of situations it can deal with. Additional benefits of the PC algorithm include robustness to agent failure and the possibility to accommodate non-collaborative vehicles (market penetration of autonomous vehicles $< 100\%$).

4.1 Introduction

In the last decade, autonomous vehicles have emerged as having a great potential to reduce congestion in cities and reduce casualties on the road [1]. The transition phase from only human-driven vehicles on the roads to only autonomous vehicles will be the most difficult to cope with. Some studies even underline the challenges linked to estimating the public acceptance and what passengers would be likely to accept from autonomous vehicles [2].

In the field of intersection coordination, several approaches coexist depending on the hypotheses considered by the authors. For instance, coordination can be achieved by changing the traffic lights pattern [3], by assigning slots to vehicles [4] or by direct vehicle control [5]. While some of those approaches work under the hypothesis of 100% of connected autonomous vehicles on the road [4, 6], the others focus on shorter-term hypotheses in which some vehicles present on the road would be neither connected nor autonomous [3]. In this case when not all vehicles are autonomous it is more difficult to find and enforce truly optimal solutions regarding the specific problem objectives (fuel consumption, time for crossing the intersection, ...). It is of interest to notice that even "simply" using platooning can double intersection throughput [7]. Thus, it seems that big gains are reachable despite not having a truly optimal coordination.

Some coordination techniques rely on mutual exclusion from a shared zone [4] (the centre of the intersection for example). More generally, the main difficulty in intersection coordination is to avoid conflicting motions which can lead to collisions. Mutual exclusion techniques are one way of achieving that which is compatible with human driving since it is what traffic lights achieve. The work presented in this chapter does not work under this hypothesis, while keeping the door open for humans to share the road. This is achieved through the use of the Probability Collectives (PC) algorithm. While the demonstrations in this chapter only include autonomous vehicles, the PC algorithm has been proven to be robust to *agent failure* [8]. Its probabilistic nature also allows the insertion of probabilistic hypotheses about the behaviour of human driven vehicles without changing the way it works.

The proposed approach thus fills a gap between human-compatible intersection coordination (based on traffic light management and/or mutual exclusion of vehicles from a shared space) and optimal coordination techniques based on 100% of connected autonomous vehicles on the road. Its decentralized nature allows it to be used even without dedicated

infrastructure. Other significant benefits of the algorithm are: the robustness to agent failure [8], compatibility with mixed-traffic scenarios (human drivers on the road) and its risk-averse behaviour based on its probabilistic characteristics.

In this chapter the demonstrations focus on showing the useability of the proposed algorithm for road scenarios since it has never been applied this way. The examples shown are for intersection crossing but the algorithm can be used for any kind of coordination, including platooning or highway insertion. The trade-off between performance and execution time will be investigated to evaluate its potential to run in real time. The overall aim is to allow the algorithm to run a full optimization in around $0.2s$ (cf. Section 4.3.2). This would allow reducing the distance at which the vehicles have to start synchronizing, and maybe running the optimization several times whenever a new event occurs.

The proposed method uses the Probability Collectives (PC) algorithm. It originates from the field of optimization and Game-Theory [8]. It is a decentralized agent-based algorithm based on probabilistic hypotheses and has been shown to be able to accommodate agent failure (if one of the agents is non-collaborative).

The remainder of this chapter is organized as follows:

- Section 4.2 presents the state of the art on the Probability Collectives algorithm and the proposed developments to apply it to intersection coordination.
- Section 4.3 presents some experiments to show the capabilities of the proposed algorithm. The first experiment is a proof of concept with detail on the optimization variables. The second experiment is a repetition of an optimization on a fixed initial situation. The aim is to determine how variable the result is from the PC algorithm (since it is based on a Monte-Carlo sampling, it has an element of randomness). The third experiment serves to study the effect of the search space sampling.

4.2 Proposed intersection coordination algorithm

4.2.1 Probability Collectives algorithm

The Probability Collectives (PC) algorithm is extensively presented in [9, 10, 8]. It is a multiagent optimization method in which different agents are playing a game iteratively. For each agent, the game consists of finding its own action (among a set of possible

actions) that maximizes the overall expected utility. In mathematical uses of the PC algorithm, agents are variables of a problem and "actions" are possible values of the variables. In the proposed approach and in [11], agents are the actual vehicles trying to solve a conflicting situation and the actions are possible trajectories. The performance of the PC algorithm has been shown to be superior to classic Genetic Algorithms (GA) [12]. It can also accommodate agent failure and non-collaborative agents [10].

The PC algorithm has the following main characteristics:

- It is **probabilistic**. Each agent computes the expected utility for each of its possible actions. To do so it gets (or estimates) the probability of the actions of the other agents.
- It does not directly output a specific action, but rather a **probability distribution** $\mathbf{q}^i(\mathbf{X}^i)$ for its set $\mathbf{X^i} = (X_1^i, ..., X_N^i)$ of possible actions (for the vehicle i). An action X_k^i more likely to be the best choice will have a high probability number.
- It is **collaborative**: the probability distribution is communicated to other agents (cf. first bullet point). It has been shown that the algorithm properties and agents behaviour (rational players) make the algorithm converge to a Nash equilibrium, which is at least a local minimum of the utility function [11].
- The probabilistic aspect is coupled with **Simulated Annealing** (SA) to allow good exploration properties when starting up the algorithm and exploitation of promising solutions at the end. To achieve that, SA techniques use a "temperature" T to define an "entropy" that starts high and is close to zero at the end of the optimization.

Most of the applications of the PC algorithm are for NP-hard problems like the salesman problem, the circle packing problem or others. These applications consist of one-off offline optimizations and the goal is to find a high-quality solution with a high computational time (several minutes). To the knowledge of the authors, the only application of the PC algorithm close to the topic of this chapter is presented in [11] for airplane conflict resolution. The PC algorithm is used as either a fully decentralized or semi-centralized intensive optimization algorithm. The implementation in [11] relies on a high volume of communications and a relatively complex problem formulation. It has been favourably compared to the *Iterative Peer-to-Peer Collision Avoidance* (IPPCA) algorithm which is a benchmark in the domain of airplane collision avoidance [11].

A coarse outline of the PC algorithm is presented below in Algorithm 1 from the point of view of one of the agents. For a more detailed description, the reader is invited to read

[10], [11] or any reference mentioned above on the PC algorithm.

Algorithm 1 Basic outline of the PC optimization

Data: Own set of possible actions $\mathbf{X^i}$
Result: Probabilities vector $\mathbf{q}^i(\mathbf{X}^i)$
Initialize $\mathbf{q}^i(\mathbf{X}^i)$ to a uniform distribution
Initialize the SA temperature T
while no convergence of $\mathbf{q}^i(\mathbf{X}^i)$ **do**
  for vehicles $j \neq i$ **do**
    if j is collaborative **then**
      Get set $\mathbf{X^j}$ from communication
      Get $\mathbf{q}^j(\mathbf{X}^j)$ from communication
    else
      Estimate $\mathbf{X^j}$
      Estimate $\mathbf{q}^j(\mathbf{X}^j)$
    end if
  end for
  for each X_k^i in $\mathbf{X^i}$ **do**
    for $j \neq i$ **do**
      Randomly sample some $\mathbf{X^j}$ based on the probabilities $\mathbf{q}^j(\mathbf{X}^j)$
    end for
    Compute expected utility $E(X_k^i)$ of action X_k^i
    (based on the sampled strategies for other vehicles)
    Store $E(X_k^i)$ in a vector $\mathbf{E}(X^i)$
  end for
  Find $\mathbf{q}^i(\mathbf{X}^i)$ minimizing $f(\mathbf{E}(X^i), T)$ (f is described in Eq. (4.2))
  Update T
end while
Apply action $X_{opt}^i = argmax(\mathbf{q}^i(\mathbf{X}^i))$

Usually the algorithm starts in a synchronized manner for all agents but there is no obligation to do so. In this chapter the simulations have been run with a synchronized start of the algorithm on all vehicles. The effects of inserting a new vehicle in the optimization while it is running will be explored later.

4.2.2 Application to Intersection Coordination

The proposed formulation of the PC algorithm introduces several novelties. All the design choices have been oriented towards a low complexity and fast optimization. The problem is also very different from airplane collision avoidance because of the highly constrained environment, its dynamic nature and the low time available to solve the coordination problem. The proposed PC was also made specifically for dealing with any type

of collaborative manoeuvre (intersection crossing, highway insertion, platooning). Thus the proposed algorithm and formulation should not be seen exclusively as an intersection coordination algorithm but more of a polyvalent collaborative planner.

Different aspects of the proposed PC formulation for intersection crossing will be detailed. The main contributions are the formulation of the search space, the specific 2-step optimization to break down complexity, the form of the utility function and the specific PC parameters to reach an interesting trade-off in terms of solution quality and execution time.

4.2.2.1 Problem formulation

In this chapter, the path of the vehicles is considered fixed through the intersection. Thus the only degree of freedom is their speed. The problem geometry is thus entirely defined by the set of all the 2D paths of the vehicles on the road. This set of 2D paths will depend on the topology of the intersection. However, the only information needed by the algorithm is the shape of the paths. As a consequence, the algorithm can cope with any type of situation as long as some paths are defined. An example of such a situation is shown in Fig. 4.1 for a "cross" intersection and a roundabout. The vehicles are at their initial position and have to follow the dotted paths to their intersection exit at a suitable speed.

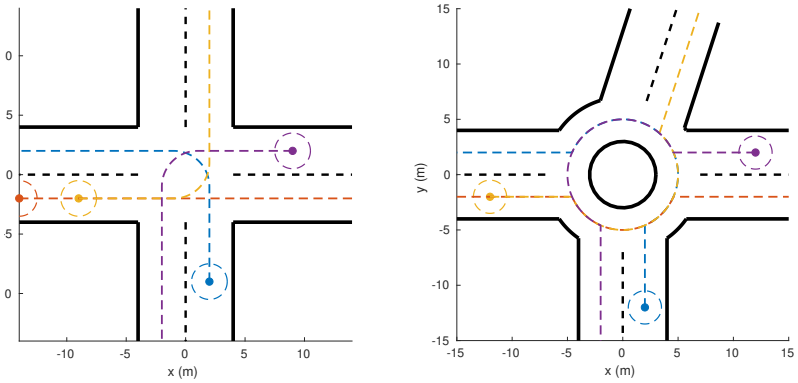

Figure 4.1: Example of intersection configuration and paths of the vehicles

In the proposed PC formulation, an element of the search space is a preset speed profile over the whole intersection. Thus, the size of the search space does not depend on the length of the conflict zone but on the number of sampled speed profiles considered. Changing this number is an easy way to modify the complexity of the optimization. The consequences of that will be discussed later.

An illustration of the possible actions (speed profiles) of a vehicle at each step of the optimization is shown in Fig. 4.2. On the upper plot, there are 10 available options to the vehicle. These options are all composed of an acceleration/deceleration of constant magnitude to a fixed speed. In this formulation the speed profiles could be summed up as a couple (a_{max}, v_{end}) (acceleration, end speed). However, it shall be noted that this set could contain randomly shaped speed profiles and is not limited to such simple shapes. Any speed profile of the form $v = f(t)$ (where f is any continuous function that respects the vehicle's dynamics) could be considered. On the bottom plot, the available options are 10 speed profiles that all start following the profile chosen at phase 1 and have reaccelerations to a nominal speed at different times. The available options could have any shape that the designer sees fit.

For the intersection application, the optimization is done in two steps (Fig. 4.2). First of all the agents are looking for a speed profile with a fixed end speed ("Phase 1" subplot) that allows them to avoid any collision. At this step, the algorithm is guaranteed to find a feasible solution if it is started when the vehicles are far enough upstream of the intersection. This way, the vehicles have the option to come to a complete stop (or an arbitrary low speed, here $0.1m/s$) before the shared zone. However their aim is to find a better speed profile that allows them to clear the intersection without coming to a stop and without colliding with any other vehicle. The second step is a reacceleration to a speed that allows the vehicles to clear the intersection as fast as possible while maintaining the collision-free characteristic of the solution.

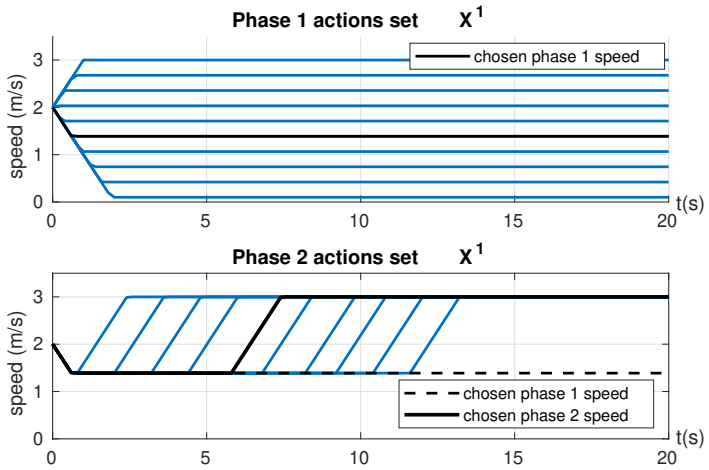

Figure 4.2: Search space representation with two-step optimization

To ensure a sampling of good quality during the first step, the final speeds have to span:

- A wide range, that needs to include $0m/s$ (a complete stop, to ensure the availability of a safe solution) and v_{max} (the maximal speed in the intersection for fast clearing)
- Closely "enough" spaced speeds so that the vehicle can "squeeze" in slots between other vehicles.

If the speed profiles sampling is too coarse, some solutions could be unattainable. For example, a car willing to insert itself on a lane between two other cars could be unable to find a suitable speed profile if its options are too coarse. One action will be too fast and the next slightly slower variation will be too slow. Thus, it will lead to a suboptimal solution through suboptimal use of space.

4.2.2.2 Objective Function

In this chapter, the proposed local utility to minimize is as follows for an agent i:

$$\begin{aligned} J(X) =& W_{sep} \sum_{i_v \neq i} \sum_{k=1}^{k_{max}} \frac{1}{d_k(i_v, i_{self})^2} \\ &+ W_{speed}(v_{max} - v_{avg})^2 \\ &+ W_{control} \sum_{k=1}^{k_{max}} \left| v_{i_{self}}(k) - v_{i_{self}}(0) \right| \end{aligned} \tag{4.1}$$

Where $d_k(i_v, i_{self})$ is the distance between the ego vehicle and the vehicle i_v at time step k. The sum over k is for all the time steps. The first term penalizes low separation distances between the vehicles. The second term penalizes slow average speeds through the intersection, and thus favours a fast crossing. The last term penalizes the control effort, that is the deviation from the initial speed of the vehicle when it approaches the intersection.

Constraints are imposed on the separation distance. For simplicity the vehicles are represented as discs and thus the separation constraint is a distance between the vehicles centres.

As usual in the PC framework, the vehicle i will not directly optimize this function. Instead, it will find a probability distribution $\mathbf{q}^i(\mathbf{X}^i) = (q^i(X_1^i), ..., q^i(X_N^i))$ for its set of N actions $\mathbf{X}^i = (X_1^i, ..., X_N^i)$ such that:

$$\mathbf{q}^i(\mathbf{X}^i) = argmin\left(\sum_{k=1}^{N} q^i(X_k^i) E(J(X_k^i)) - TS(\mathbf{q}^i(\mathbf{X}^i)) \right) \tag{4.2}$$

Where $E(J(X_k^i))$ is the expectancy of the utility function J for the strategy X_k^i of the vehicle i. It is computed by randomly sampling the other vehicles' strategies and computing the obtained utility J. The obtained utility is averaged over several samplings of the other vehicles' strategies. This random sampling is done according to the latest probability distribution of the actions of the other vehicles', as they communicated it to vehicle i. This distribution is denoted $\mathbf{q}^j(\mathbf{X}^j)$ (for $j \neq i$). The parameter T is specific for the Simulated Annealing (SA) and is called the temperature. The function S is an entropy defined as:

$$S(\mathbf{q}^i(\mathbf{X}^i)) = -\sum_{k=1}^{N} q^i(X_k^i) \ln(q^i(X_k^i)) \tag{4.3}$$

At the beginning of the PC optimization, the parameter $T \in \mathbb{R}$ is big, which weighs the entropy term more. If T is infinite, the optimal $\mathbf{q}^i(\mathbf{X}^i)$ is uniform and the $q^i(X_k^i)$ are all equal to $1/N$. This favors the exploration of all possible solutions. At the end, T is brought close to zero to weight the expectancy term more.

As the speed profiles span the whole duration of the intersection crossing, the PC optimization can be run just once. This comes from the fact that the shared road space has finite dimensions and well-defined bounds. One run of the PC algorithm corresponds to several iterations in which the probabilities of each action are exchanged at each iteration between the vehicles as they update their probability distribution.

It shall be noted that the probability distributions $\mathbf{q}^j(\mathbf{X}^j)$ of other vehicles' actions may not be available. For instance, some vehicles may be human-driven. In that case, an estimation of the probability distribution should be carried out based on sensory information: use of blinkers, speed of entry...

4.2.3 Comparison with existing work using PC

Several key differences exist between our approach and the approach for airplane collision avoidance presented in [11]. In this airplane collision avoidance approach, the search space is a succession of heading changes over a rolling horizon. The headings are determined one by one at regular time intervals. In our approach, an element of the search space is a whole speed profile over a fixed time horizon, greatly limiting the complexity of the search space. In [11] the environment is not dense (airspace) so a solution can always be found. The minimal separation distance is implemented as a soft constraint. The emphasis is not on finding a feasible solution (easy to do in this case) but on finding a really optimal solution. In our approach the space is very constrained: there are a lot

of vehicles in a small area. The emphasis is on finding a feasible collision in a very short time that corresponds to the time constants found in ground traffic.

In [11], objective functions are shared and agreed upon so they are *homogeneous* through the range of all agents. It is interesting to note that the optimization runs at regular interval when the horizon has moved. Thus, the future collisions (or separation distance violations) are less penalized than the imminent collisions in the cost function. Because of the heavily constrained space for traffic management, the horizon of the optimization has been chosen to span the whole intersection. This is possible for ground traffic because the boundaries of the shared zone are usually clearly identified.

Finally, a typical optimization in [11] will see 800MB of data exchanged between airplanes in the fully decentralized version. The time spent for optimization is up to 120s for a round of PC optimization with 25 aircraft. There is a linear complexity dependence for the decentralized PC implementation under some realistic hypotheses (broadcasting messages about partial costs and predictions). The semi-centralized PC operation takes longer, at 610s for the same conflict. The fully decentralized approach is thus better at scaling. In the proposed PC formulation and implementation, the data exchange is intended to be minimal and a solving time of $0.2s$ is targeted ($0.8s$ have been achieved so far for 4 vehicles on non-optimized Matlab code). The typical data exchange would be in the range of several MB (detailed in section 4.3.1).

4.3 Experiments

This section presents experiments done with a first implementation in Matlab. Simulations show a cross intersection scenario, but the work is generalizable to any type of collaborative maneuver.

The algorithm has been implemented in its fully decentralized version for better scaling [11].

4.3.1 Proof of concept

This section details a simulation done on a given initial situation with 4 vehicles at a cross intersection. Table 4.1 shows the main parameters chosen for the PC algorithm. The sampling time is used to represent the speed profiles and to integrate the positions

of the vehicles over the time horizon. This information is used to compute the cost J. In this simulation, the cost function contains only the terms on the average crossing time (altruistic objective) and on the separation distance. If a solution violates the separation constraint (represented by the dotted circles in Fig. 4.3), an additional cost J_{cons} is added for each constraint violated. The algorithm stops if the global solution does not change for N_{stop} iterations. The weight on the control effort has been left at zero for the moment to focus on the "altruistic" terms of the utility function. The $M1$ column is the regular mode of the PC algorithm for fast optimization, and the $M2$ mode has been used as a longer optimization to find an optimal solution. The annealing schedule is much slower and starts at a higher temperature to allow the algorithm to explore more possibilities. The number of possible actions is also higher. This mode will be used in section 4.3.2.

A snapshot of the results of the optimization is shown in Fig. 4.3. The vehicles are represented as circles so far for the sake of simplicity. The solution seems intuitive, because it allows the two vehicles with the highest entry speed (yellow and orange) to carry their speed through the intersection. The purple (resp. blue) vehicle lowers its speed just enough to yield to the yellow (resp. orange) vehicle. Thus during this fully decentralized optimization, the algorithm has arrived at a relevant solution without any constraint violation.

At the beginning of the simulation, each vehicle broadcasts its set of strategies to other vehicles. Each strategy is a *float* vector of size 200 ($40s$ horizon and $0.2s$ sampling time) and the set has 10 strategies. That is a total of 2000 floats exchanged per vehicle, or 8kB (kilobytes). This is done again at the beginning of the phase 2 (reacceleration, cf. subsection 4.2.2.1). Then for each iteration the vehicle broadcasts its updated probability vector $\mathbf{q}^i(\mathbf{X}^i)$ of 10 floats. The optimization in the $M1$ mode runs in around 20 iterations so it will be a total broadcasted per vehicle of 200 floats (0.8kB). For 4 vehicles, the data volume broadcasted will be around 16.8kB per vehicle and 67.2kB in total. If the objective of doing the optimization in $0.2s$ is respected, it means the required network thoughput should be in the order of magnitude of 0.4MB/s. Of course this does not consider the overhead due to the communication protocols, and considers that messages can be broadcasted. Even with a pessimistic hypothesis of a requirement ten times superior, it would stay well within what is physically possible (4MB/s required).

Table 4.1: Main PC parameters

Parameter	**Notation**	**Value**	
		M1	*M2*
Number of strategies	N_s	10	20
Sampling time	T_s	$0.2s$	
Weight on control effort	$W_{control}$	0	
Weight on separation dist.	W_{sep}	1	
Weight on avg. crossing time	W_{avg}	10	
Penalty for constraint violation	J_{cons}	10^5	
Samples to get expected utility	$N_{samples}$	10	20
Stopping criteria	N_{stop}	4	10
SA start temperature	T_{init}	1	10
SA end temperature	T_{end}	0	0
SA temperature step	T_{step}	0.2	0.66

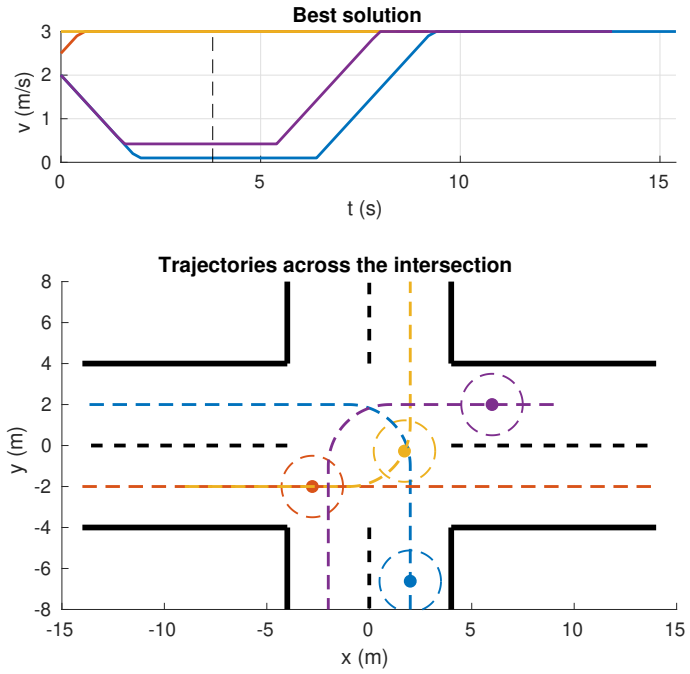

Figure 4.3: Snapshot of the application of the best solution when $t = 4s$. The purple vehicle is entering from the right lane and will exit at the bottom. It slowed down just enough to yield for the yellow vehicle (yellow vehicle came from the right, exits at the top). The red vehicle (coming from the left, exits to the right) accelerated up to its maximal allowed speed so that the blue one has to wait the minimal amount of time before entering the intersection (blue enters from the bottom). Full video available at `https://youtu.be/XTgY-4RUFz0`

Table 4.2: Results of the PC consistency test (100 runs)

Parameter	**Notation**	**Distribution**
$M1$ mode		
Average crossing time	$t_{avg,M1}$	$8.4s \pm 0.7s$
Max crossing time	$t_{max,M1}$	$12.5s \pm 1.9s$
Execution time	$t_{exec,M1}$	$3.3s \pm 0.4s$
$M2$ mode		
Average crossing time	$t_{avg,M2}$	$7.8s \pm 0.2s$
Max crossing time	$t_{max,M2}$	$10.6s \pm 0.2s$
Execution time	$t_{exec,M2}$	$14.6s \pm 1.0s$

4.3.2 Analysis of consistency on fixed initial situation

For this series of simulations, the initial situation has been fixed to be the same as in section 4.3.1. The algorithm has been run several times to check consistency. For reference, the performance metric distribution has been compared with the performance found with a run of the algorithm in the $M2$ mode (slower and more "optimal" solution). The results are shown in Table 4.2. It shall be noted that the algorithm provided solutions with no constraint violation in 100% of the cases.

The distribution is shown in Fig. 4.4. The distribution in the histograms seems discrete for the average time and max time because of the discrete nature of the search space. The algorithm used in *Mode 1* found most of the time a single solution yielding a crossing time of $8.5s$, and sometimes solutions yielding either $t_{avg} = 7.5s$ or $t_{avg} = 9.5s$.

When running the algorithm in the more precise mode $M2$, the crossing times for the vehicles are slightly improved in terms of average value and highly improved in terms of standard deviation. This comes however with a 4.4 times higher computation time. The fast mode (and low complexity) of the PC optimization thus seems to give a good balance between the quality of the solution and the execution time. The overall consistency is quite good and even the suboptimality is not critical. The main interest of the M2 mode is the increased consistency. In both cases, the best solution the algorithm can output (at $7.2s$ of average crossing time) has only been found a handful of times.

It shall be noted that computation time is given for a Matlab simulation where the code for all the vehicles is not parallelized. The execution time *per vehicle* would be 4 times less for these conditions with the same code. It means that would be around $0.8s$ in this case. It is expected that the code could run much faster when implemented in C++. With the current figure or a vehicle going at $14m/s$ ($50km/h$), it means that the synchronization

should start around $11m$ before the point where any braking should start. For reference, an emergency braking to a full stop at this speed is around $14m$ on modern cars and dry roads. Thus, the distance needed for synchronization is of an acceptable magnitude for the considered application. The authors expect to bring the optimization time down to $0.2s$ ($3m$ travelled at $50km/h$).

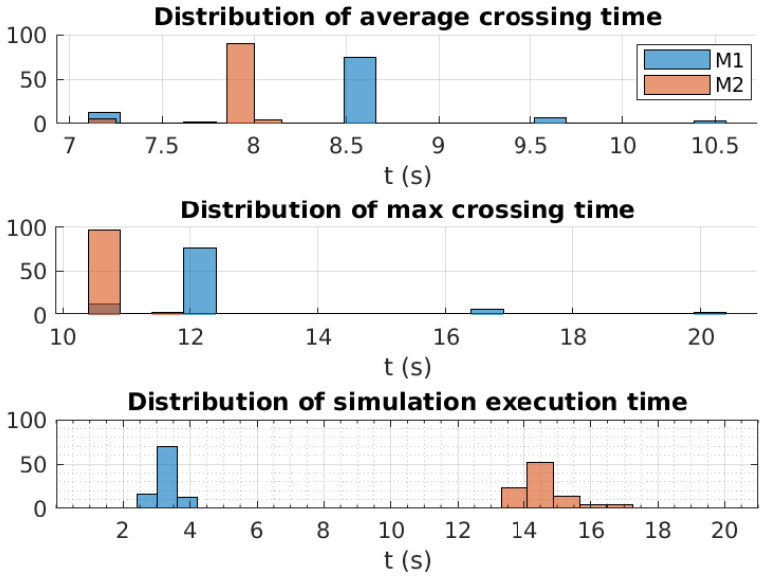

Figure 4.4: Results of 100 runs of the PC optimization for the same initial conditions. Mode 1 in blue and Mode 2 in orange.

4.3.3 Influence of search space sampling

In this experiment the effect of changing the number of available actions is tested. The *available actions* refer here to several speed profiles that the vehicle can choose from to cross the intersection. Instinctively, if a vehicle has fewer actions to chose from, it should show a decrease in performance. The simulations are done on the same settings as before on a cross intersection and with 4 vehicles. Optimization parameters are the same as shown in Table 4.1 (Mode 1). The number of available actions is changed between 4 and 14.

The results are shown in Fig. 4.5. The average performance does not really decrease (average crossing time, max crossing time) but more and more outliers appear on the middle plot (distribution of the crossing time of the last vehicle). This means that the algorithm struggles to guarantee consistent performance. The outliers most likely correspond to situations where an additional intermediate action should have been undertaken to squeeze between two (or several) vehicles.

However, the execution time plot shows a significant decrease in execution time. For a search space of size $N_{strats} = 4$, the average execution time is $1.7s$. This corresponds to $0.4s$ per vehicle, and $5.6m$ travelled at $14m/s$ ($50km/h$). With this setting, the algorithm starts to show real time capabilities (considering the code is not yet optimized) but fails to propose refined enough solutions in some cases. Ideally, a very low number of available options could be kept if a post-processing of the speed profile is applied: it is easy to detect if a vehicle waits more than is necessary because of not enough available options. It would keep the execution time low but would not solve the situations where a vehicle failed to find a solution to "squeeze" between other vehicles.

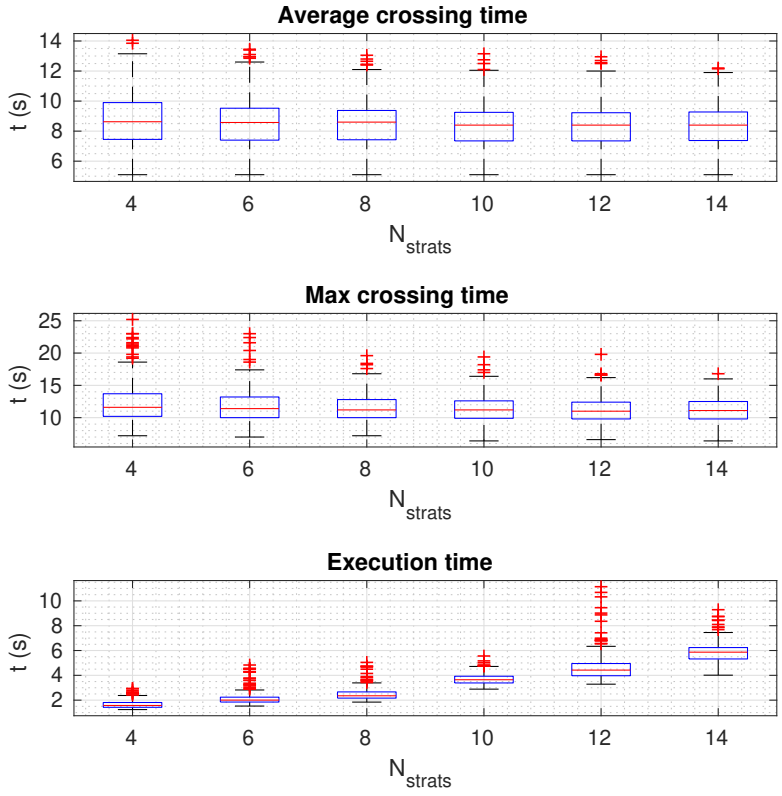

Figure 4.5: Effect of the size of the search space on the performance of the coordination algorithm.

4.4 Conclusion

A novel formulation of the Probability Collectives (PC) algorithm has been presented to apply it to ground traffic coordination. The main interests of the proposed algorithm are its

low complexity compared to usual PC applications and its flexibility to any kind of road scenario. Its capabilities have been demonstrated for an intersection crossing in which the space is highly constrained. The algorithm exhibits good exploration properties to find relevant solutions in a very short time ($0.8s$ in average for the demonstrated scenarios). It is also fully decentralized and can be applied to collaborative manoeuvre. Trade-offs between performance, consistency and execution time have been highlighted. The current performance hints towards true real-time implementation of the proposed algorithm. In order to further minimize the execution time, it is planned in the near future to switch to a ROS-based C++ implementation.

Further work will focus on practical implementation problems. For example the algorithm should re-run if a new vehicle wants to join the collaboration. In this case some continuity of the already computed speed profiles shall be enforced. Furthermore, the algorithm will be tested in situations where the cost functions of the vehicles are non-homogeneous. Non-collaborative vehicles or human-driven vehicles will also be inserted to check the robustness of the proposed algorithm to such elements.

References

[1] Daniel J. Fagnant and Kara Kockelman. “Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations”. In: *Transportation Research Part A: Policy and Practice* 77 (2015), pp. 167–181.

[2] Scott Le Vine, Alireza Zolfaghari, and John Polak. “Autonomous cars: The tension between occupant experience and intersection capacity”. In: *Transportation Research Part C: Emerging Technologies* 52 (Mar. 2015), pp. 1–14.

[3] Hironori Suzuki and Yoshitaka Marumo. “A New Approach to Green Light Optimal Speed Advisory (GLOSA) Systems for High-Density Traffic Flowe”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. 2018, pp. 362–367.

[4] Robert Hult, Gabriel R. Campos, Paolo Falcone, and Henk Wymeersch. “An approximate solution to the optimal coordination problem for autonomous vehicles at intersections”. In: *Proceedings of the American Control Conference*. Vol. 2015-July. IEEE, July 2015, pp. 763–768.

[5] Stefanie Manzinger and Matthias Althoff. “Tactical Decision Making for Cooperative Vehicles Using Reachable Sets”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. IEEE, Nov. 2018, pp. 444–451.

[6] Andreas A. Malikopoulos, Christos G. Cassandras, and Yue J. Zhang. “A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections”. In: *Automatica* 93 (2018), pp. 244–256. arXiv: 1602.03786.

[7] Jennie Lioris, Ramtin Pedarsani, Fatma Yildiz Tascikaraoglu, and Pravin Varaiya. “Platoons of connected vehicles can double throughput in urban roads”. In: *Transportation Research Part C: Emerging Technologies* 77 (2017), pp. 292–305. arXiv: 1511.00775.

[8] Anand Jayant Kulkarni and Kang Tai. “A Probability Collectives Approach for Multi-Agent Distributed and Cooperative Optimization with Tolerance for Agent Failure”. In: *Agent-Based Optimization. Studies in Computational Intelligence, vol. 456*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[9] Michalis Smyrnakis and David S. Leslie. “Sequentially updated probability collectives”. In: *Proceedings of the IEEE Conference on Decision and Control*. IEEE, Dec. 2009, pp. 5774–5779.

[10] Anand J. Kulkarni and K. Tai. “Probability Collectives: A multi-agent approach for solving combinatorial optimization problems”. In: *Applied Soft Computing Journal* 10.3 (June 2010), pp. 759–771.

[11] David Šišlák, Pemysl Volf, Michal Pěchouček, and Niranjan Suri. “Automated conflict resolution utilizing probability collectives optimizer”. In: *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* 41.3 (May 2011), pp. 365–375.

[12] Chien-Feng Huang, Stefan Bieniawski, David H Wolpert, and Charlie EM Strauss. “A comparative study of probability collectives based multi-agent systems and genetic algorithms”. In: *Proceedings of the 7th annual conference on Genetic and evolutionary computation*. 2005, pp. 751–752.

Chapter 5

Robust and Decentralized Traffic Flow Management of Autonomous Vehicles at Intersections based on Probability Collectives Algorithm

Abstract

In this chapter, the potential for the "Probability Collectives" (PC) algorithm to solve continuous multi-vehicle coordination at an intersection is evaluated. Basic application of the PC algorithm to Multi-Vehicle Coordination (MVC) has been demonstrated in [1] and in chapter 4. This chapter extends the previous work by demonstrating robustness to non-collaborative vehicles and continuous traffic flow management capabilities. The PC algorithm is a decentralized probabilistic optimization algorithm that has roots in optimization theory and Game Theory. It has been used with great success in the field of NP-hard optimization, for example for the traveling salesman problem. Following previous work establishing the possibility to use the PC algorithm for fast multi-vehicle coordination, more specific aspects are now investigated. In this chapter is detailed the study of the flexibility of the algorithm to different use cases, its ability to accommodate non-collaborative vehicles and practical aspects to enable its use in real life scenario: smoothness of the generated trajectories and ability to recompute solutions dynamically in a changing environment. The results are promising and allow to consider real life implementation. Further works will focus mainly on its compatibility to scenarios with mixed traffic (humans and autonomous vehicles).

5.1 Introduction

In the last decade, autonomous vehicles have emerged as having a great potential to reduce congestion in cities and reduce casualties on the road [2]. The most difficult phase to cope with has been foreseen as the transition between 100% of humans on the road and 100% of autonomous vehicles. Some studies also underline general public and passenger acceptance challenges [3].

5.1.1 Related works

Intersection coordination methods can be split into two main categories. The first category includes management strategies that are designed for mixed traffic (or human only traffic). In these conditions, coordination is often achieved by changing the traffic lights pattern [4]. The second category includes approaches that consider that 100% of the vehicles on the road are autonomous. A wider variety of coordination mechanisms can then be used, such as slot assignment to vehicles [5] or direct vehicle control [6]. The more conservative coordination techniques rely on mutual exclusion from a shared zone [5, 7]. More generally, the main difficulty in intersection coordination is to avoid conflicting trajectories. Mutual exclusion techniques are one way of achieving that which is compatible with human driving since it is what traffic lights achieve. This method is akin to an adaptive traffic light that lets through one vehicle at a time through the intersection area.

Optimal solutions for intersection crossing are attractive, but are often complex to obtain. They require all the vehicles to be autonomous and the environment to be under control. However, improvements of traffic flow metrics (throughput, fuel consumption, ...) can be obtained even with simple coordination mechanisms [8].

The method proposed in this chapter tries to strike a middle ground. The *mutual exclusion* techniques are deemed too conservative in their approach, leading to a slow intersection crossing. The proposed algorithm tries to keep the door open for humans to share the road while removing the restriction of mutual exclusion of the vehicles. This is achieved through the use of a decentralized Probability Collectives (PC) algorithm. The PC algorithm has been proven to be robust to *agent failure* [9] in theoretical problems: that is when an agent stops collaborating. This property is demonstrated in this chapter for the practical problem that is intersection coordination, with all its characteristics and constraints. The probabilistic nature of the PC algorithm also allows the insertion of probabilistic hypotheses about the behaviour of human driven vehicles without changing the

way it works.

The proposed approach thus fills a gap between human-compatible intersection coordination (based on traffic light management and/or mutual exclusion of vehicles from a shared space) and optimal coordination techniques based on 100% of connected autonomous vehicles on the road. Its decentralized nature allows it to be used even without dedicated infrastructure. Another benefit is the probabilistic representation of data within the algorithm, which leads to a risk-adverse behaviour. The demonstrated property of robustness to agent failure for intersection crossing will be used to develop a framework for continuous traffic flow management. Other potential benefits of the algorithm include compatibility with mixed-traffic scenarios (human drivers on the road).

To the knowledge of the author, the only similar implementation of PC for a traffic management problem is for Air Traffic Control [10]. The goal was similar (conflict solving) although the environment is different in its lack of spatial constraints (airspace) and the conflict solving process is quite heavy (several hundreds of MB of data exchanged). All the agents were connected and collaborative, and the problem was solved offline once (no continuous air traffic management). Thus, this work is more akin to [1].

5.1.2 Organization of the chapter

Following the works presented in [1], an improvement in the applicability of the PC algorithm for intersection crossing is shown in this chapter. It also pushes further the characterization and performance analysis of the algorithm. In this chapter, *intersection* is used to describe any kind of intersection (crossroad, roundabout, ...) as the algorithm has been designed to be independent from the road layout.

The remainder of this chapter is organized as follows:

- Section 5.2 is a reminder of the adaptation and improvement of the PC algorithm for intersection crossing.
- Section 5.3 brings further the analysis done in [1] on the characterization of the algorithm. Monte-Carlo simulation have been used to randomize the initial situations and draw conclusions on the flexibility of the algorithm and its scalability.
- Section 5.4 explains how a specific framework has been built to enable the use of the PC algorithm for real-life situations with continuous traffic. The behaviour of the algorithm with continuous traffic applications is demonstrated.

5.2 Proposed Intersection Coordination Algorithm

The Probability Collectives (PC) algorithm is described in [11, 12, 9]. It is a multi-agent optimization method in which different agents are playing a game iteratively. For each agent, the game consists of finding an action (among a set of possible actions) that maximizes the shared utility of all the agents. More precisely, each agent associates a probability to each of his possible actions, denoting the likelihood that this action is the best choice. These probabilities are then shared among all the agents. When this procedure is repeated, each agent gets an increasingly better knowledge over time of what the others would like to do, and how certain they are of their choice. After repeated rounds, the group of agents will have converged to a best action to undertake (one action will have a probability of 1 for each agent), and the algorithm stops.

To sum up, the PC algorithm has the following main characteristics:

- It is **probabilistic**. Each agent computes the expected utility for each of its possible actions. To do so it gets (or estimates) the probability of the actions of the other agents.
- It does not directly output a specific action, but rather a **probability distribution** $\mathbf{q}^i(\mathbf{X}^i)$ for its set $\mathbf{X^i} = (X_1^i, ..., X_N^i)$ of possible actions (for the vehicle i). An action X_k^i more likely to be the best choice will have a high probability number.
- It is **collaborative**: the agents get the information from other agents through communication. It has been shown that the algorithm properties and agents behaviour (rational players) make the algorithm converge to a Nash equilibrium, which is at least a local minimum of the utility function [10] in the case of communication between agents. It is interesting to note that the intentions of the other agents (possible actions and probabilities) could also be estimated instead of obtained through communication.
- It balances exploration of the search space with exploitation of promising solutions through **Simulated Annealing** (SA). The exploration properties have been shown to be very good in the case of multi-vehicle coordination [1].
- It has been shown to be **robust to agent failure** [9]. That means the algorithm will converge anyway if one of the agents stops collaborating. For instance, that could be a vehicle refusing to change its planned action. This property is useful for continuous traffic situations (cf. section 5.4.3) and/or to implement strong priorities

for certain classes of vehicles (emergency vehicles, police, ...)

Most of the applications of the PC algorithm are for NP-hard problems like the traveling salesman problem, the circle packing problem or others. These applications consist of one-off offline optimizations and the goal is to find a high-quality solution with a high computational time (several minutes).

The outline of the PC algorithm is reminded below in Algorithm 2 for ease of reading. For a more detailed description, the reader is invited to read [12], [10] or any reference mentioned above on the PC algorithm.

Algorithm 2 Basic outline of the PC optimization

Data: Own set of possible actions $\mathbf{X^i}$
Result: Probabilities vector $\mathbf{q}^i(\mathbf{X}^i)$
Initialize $\mathbf{q}^i(\mathbf{X}^i)$ to a uniform distribution
Initialize the SA temperature T
% Main loop of the algorithm:
while no convergence of $\mathbf{q}^i(\mathbf{X}^i)$ **do**
  % Loop to get set of possible actions and probabilities for other vehicles:
  for vehicles $j \neq i$ **do**
    if j is connected **then**
      Get set $\mathbf{X^j}$ from communication
      Get $\mathbf{q}^j(\mathbf{X}^j)$ from communication
    else
      Estimate $\mathbf{X^j}$
      Estimate $\mathbf{q}^j(\mathbf{X}^j)$
    end if
  end for
  % Loop to evaluate the cost expectancy of each action (for self):
  for each X_k^i in $\mathbf{X^i}$ **do**
    for $j \neq i$ **do**
      Randomly sample some $\mathbf{X^j}$ based on the probabilities $\mathbf{q}^j(\mathbf{X}^j)$
    end for
    Compute expected utility $E(X_k^i)$ of action X_k^i
    (based on the sampled strategies for other vehicles)
    Store $E(X_k^i)$ in a vector $\mathbf{E}(X^i)$
  end for
  Find $\mathbf{q}^i(\mathbf{X}^i)$ minimizing $f(\mathbf{E}(X^i), T)$ (f is described in Eq. (5.1))
  Update T
end while
% When algorithm has converged, apply best action:
Apply action $X_{opt}^i = argmax(\mathbf{q}^i(\mathbf{X}^i))$

From the point of view of one agent, the other agents are represented as a set of possible

actions $\mathbf{X^j}$ with associated probabilities of each action happening $\mathbf{q}^j(\mathbf{X}^j)$. This property can be used to allow the algorithm to work with non connected agents (or vehicles in our case). Instead of getting the information from the non-connected vehicle, a set of possible actions can be sampled from its observations, and likelihood estimations can be carried out. Human driven vehicles fall into that category: they are neither collaborative or connected but can still be inserted in the optimization process as "virtual failed agents".

In the PC framework, the optimization that will be done by each agent for a given iteration is of the form:

$$\mathbf{q}^i(\mathbf{X}^i) = argmin\left(f(\mathbf{E}(X^i), T)\right) \tag{5.1}$$

Where:

$$f(\mathbf{E}(X^i), T) = \sum_{k=1}^{N} q^i(X_k^i)E(J(X_k^i)) - TS\left(\mathbf{q}^i(\mathbf{X}^i)\right) \tag{5.2}$$

It means that the goal is to find a discrete probability distribution $\mathbf{q}^i(\mathbf{X}^i) = (q^i(X_1^i), ..., q^i(X_N^i))$ for the actions of the ego vehicle $\mathbf{X}^i = (X_1^i, ..., X_N^i)$ so that the utility function $f(\mathbf{E}(X^i), T)$ is minimized. This utility function has two terms. The first one is the weighted sum of the expectancy of local utilities of each action (how "good" is supposed to be each possible action). The second term is an entropy term. When T is high, this term is the most important and steers the discrete probability distribution $\mathbf{q}^i(\mathbf{X}^i)$ to a uniform distribution. This is the part that comes from the field of *Simulated Annealing* (SA). A high T (temperature) encourages exploration of all the possibilities. Typically, T starts high and will then be progressively lowered to encourage exploitation.

The overall utility function $f(\mathbf{E}(X^i), T)$ uses the **expected** utilities J of each strategy because J is not deterministic for a given action. It depends on the choice of actions of other vehicles. These choices are not known in a deterministic manner. In fact, the other vehicles communicate the probabilities of choosing each of their possible actions (their own set $\mathbf{q}^j(\mathbf{X}^j)$ for $j \neq i$). Because of the probabilistic nature of the knowledge involved, J can only be computed as its expectancy $E(J)$ for any given action.

It is interesting to note that as the optimization progresses, the vehicles will slowly become "certain" of which action to choose. In other words, one of the probabilities in $\mathbf{q}^i(\mathbf{X}^i)$ will be 1 and the rest at 0. So at the end of the optimization, the expected utilities $E(J)$ converge to the deterministic utilities J.

In this chapter, the proposed local utility J associated to a given action has several terms [1]:

- The first term rewards a good separation distance between vehicles.
- The second term penalizes the time spent in the intersection, to favorize fast crossing
- The last term penalizes the control effort. Vehicles are encouraged to keep a constant speed as much as possible.

Constraints are imposed on the separation distance. For simplicity the vehicles are represented as discs and thus the separation constraint is a distance between the vehicles centres.

For greater detail on the implementation the reader is invited to read the previous work of the author [1], and the previously mentioned references for more detail on the PC algorithm [12, 9, 13].

5.3 Algorithm Characterization

In this section, several characteristics of the algorithm are evaluated in a non-continuous vehicle flow. That means an initial situation is generated randomly with a fixed number of vehicles, and the PC algorithm is run so that the vehicles find a solution to cross the intersection. The evaluation metrics are:

- The average time for vehicles to cross the intersection (from the start of the simulation to when the vehicles exit the intersection) [1].
- The crossing time of the slowest vehicle.
- The execution time. Although it highly depends on the machine on which it is executed and the optimization of the code, it allows to draw conclusions when two simulations are run on the same machine in similar conditions.

The following characteristics of the algorithm are evaluated:

- The behaviour of the algorithm with a varying sample size to evaluate the cost expectancy $E(J(X_k^i))$ (cf. Eq. 5.1).
- The scalability properties of the algorithm depending on the number of agents $N_{vehicles}$ in the scheme.
- The influence of the intersection layout on the performance of the algorithm (crossroads versus roundabout).

Table 5.1: Monte-Carlo simulations parameters

Parameter	**Notation**	**Range**
Start distance to intersection	D_s	$[7m, 12m]$
Initial inter-vehicle distance	D^i_{sep}	$[1m, 3m]$
Index of entry road	i_{in}	$\{1, 4\}$
Index of exit road	i_{out}	$\{1, 4\} \setminus i_{in}$
Initial speed	v_{init}	$[0.1m/s, 3m/s]$
Number of simulations	N_{sims}	100

Table 5.2: Main PC parameters

Parameter	**Notation**	**Value**
Sampling time	T_s	$0.2s$
Optimization horizon	T_opt	$30s$
Number of strategies	N_s	10
Weight on control effort	$W_{control}$	0
Weight on separation distance	W_{sep}	1
Weight on avg. crossing time	W_{avg}	10
Penalty for constraint violation	J_{cons}	10^5
Samples to get expected utility	$N_{samples}$	10
Stopping criteria	N_{stop}	4
SA start temperature	T_{init}	1
SA end temperature	T_{end}	0
SA temperature step	T_{step}	0.2

To this end, Monte-Carlo simulation have been carried away with randomized initial situations. The randomized parameters are:

- The starting distances from the intersection
- The initial distance between vehicles
- The upstream road and destination of the vehicles
- The initial speeds

These parameters are summed up in Table 5.1 for the simulations carried out. If any Monte-Carlo simulation has a different spread of parameters, it will be specified. The distributions of parameters are all uniform. For a vehicle that follows closely another in the initial situation, the initial speed has been constrained to prevent an unavoidable collision because of the initial situation.

Unless specified otherwise, the main PC algorithm parameters values are defined in Table 5.2.

The intersection layouts for a crossroad and a roundabout are shown in Figure 5.1. These two use cases will be used for the Monte-Carlo simulations. As mentioned in [1], the algorithm can accommodate any situation as long as 2D paths are defined.

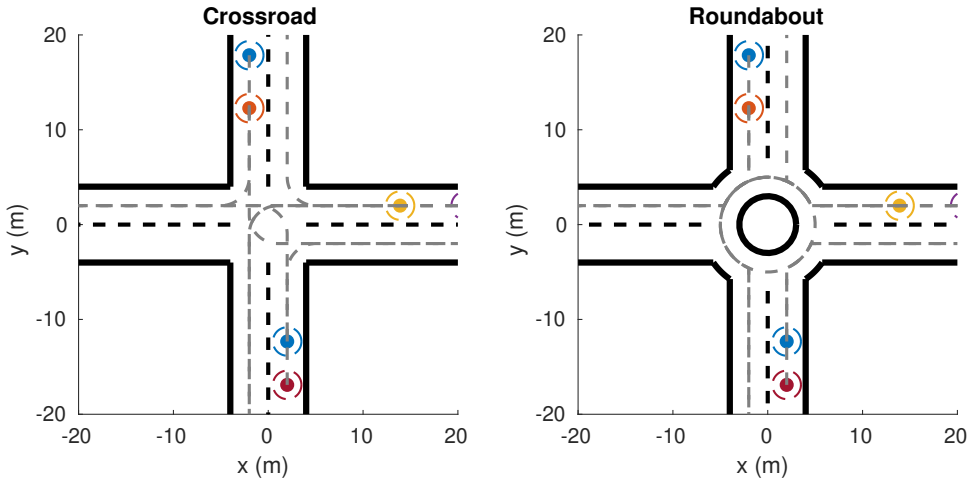

Figure 5.1: Comparison of the layout of a traditional crossroad and a roundabout. Matched upstream roads and initial position of vehicles.

In these simulations, the set of possible actions $\mathbf{X^i}$ of the vehicles is the same as in [1]. The optimization is done in two steps.

- During the first step, the possible speed profiles are composed of a constant acceleration phase to a constant speed. Each possible speed profile has a different constant speed value.
- After converging to a speed profile, the second step is started. During this second step the PC algorithm keeps running but the set of actions $\mathbf{X^i}$ changes. From the previously chosen speed profile, the different options are a reacceleration to a nominal speed to clear the intersection as fast as possible.

Examples of those sets are shown in Figure 5.2. The numerical application has been done to match the IPCar Vehicles on the PAVIN platform in Clermont-Ferrand (France). It is a scaled down city neighborhood used for tests with the IPCar vehicles whose maximum speed is about $3m/s$. The scaling down of the roads and of the vehicles dimension and speed give similar dynamics as on the road in a more compact setting and with lower

kinetic energy.

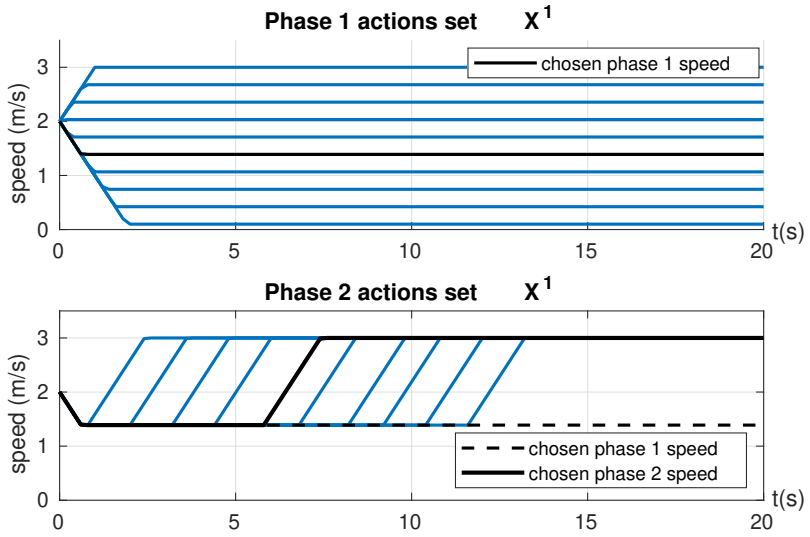

Figure 5.2: Search space representation with two-step optimization

5.3.1 Effect of sample size for cost expectancy computation

To have a clear picture of the cost expectancy: randomly sample a lot of strategies on the probability distribution. Otherwise the cost expectancy is computed from not enough cases and only reflects particular situations.

If several but not enough samples, all the relevant cases will be represented but it will give too much importance to rare cases (distribution poorly approximated).

The results in Figure 5.3 show no significant difference in the quality of the solution found among all simulations. For $N_{samples} = 1$ and $N_{samples} = 5$, the execution time is higher than for $N_{samples} = 10$ and has higher variance. The standard deviation of the execution time is reduced from $N = 10$ upwards, and the computation time increases ($N = 15$ and $N = 20$) still with a low standard deviation.

As the time spent in each iteration of the PC is lower for a lower $N_{samples}$ (quicker to compute the expected utility), it means that more iterations are needed for $N_{samples} = 1$ and $N_{samples} = 5$, i.e. the algorithm takes more time to converge. This is because at each iteration, the update of the probability distribution $\mathbf{q_i}(\mathbf{X^i})$ will be done with more scarce information: under $N_{samples} = 10$, the set of actions $\mathbf{X^j}$ associated to the probability distributions $\mathbf{q_j}(\mathbf{X^j})$ are not sampled correctly. As a remainder, the size of the set of

possible actions for each vehicle in these simulations is $N_s = 10$. From the results it appears that a number of samples of at least $N_{samples} > N_s$ is necessary for ensuring the quality of the strategy sampling for the cost expectancy computation. The optimal seems to be at $N_{samples} = N_s$. Any further increase of $N_{samples}$ increases the execution time without reducing its variance.

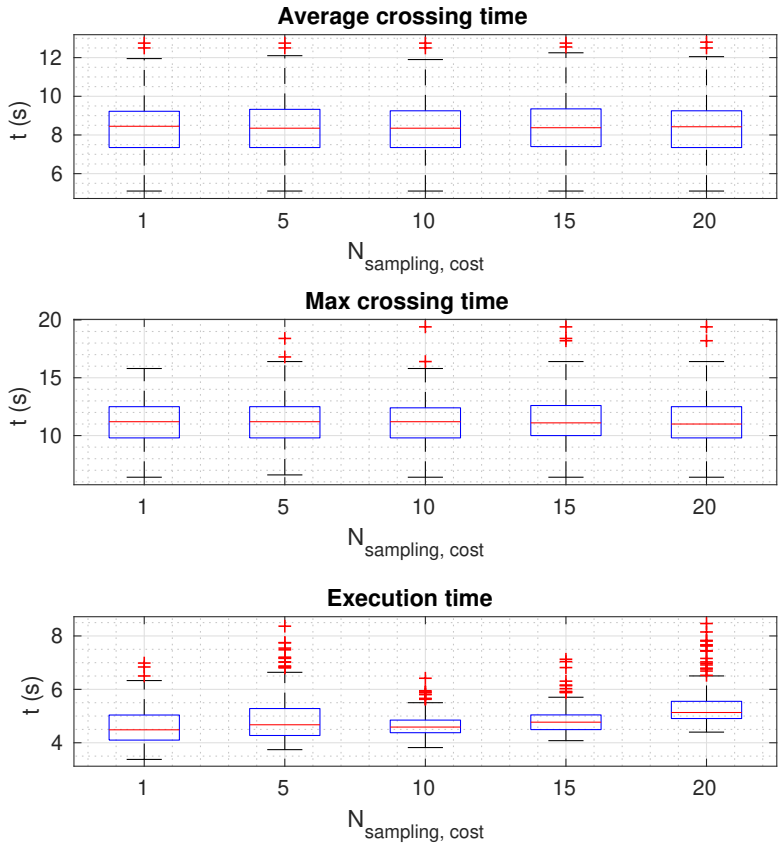

Figure 5.3: Effect of sample set size for cost expectancy computation

5.3.2 Scalability properties

In this Monte-Carlo run, the performance of the PC algorithm is evaluated with an increasing number of vehicles. The randomized parameters are within the same range, except that the initial distance from the intersection can exceed the upper bound if too many vehicles are generated on the same upstream road. The separation distance constraints take precedence.

The aim of this run is to find the limiting number of vehicles for the application of this

algorithm, and see how efficient it is at dealing with high number of vehicles in a restricted space. Figure 5.4 shows results for $N_{vehicles} = 3$ to $N_{vehicles} = 9$.

In this plot, the execution time is shown *per vehicle*, as in real life the algorithm would run in parallel on all the vehicles. With the present Matlab implementation the code has not been parallelized (neither optimized).

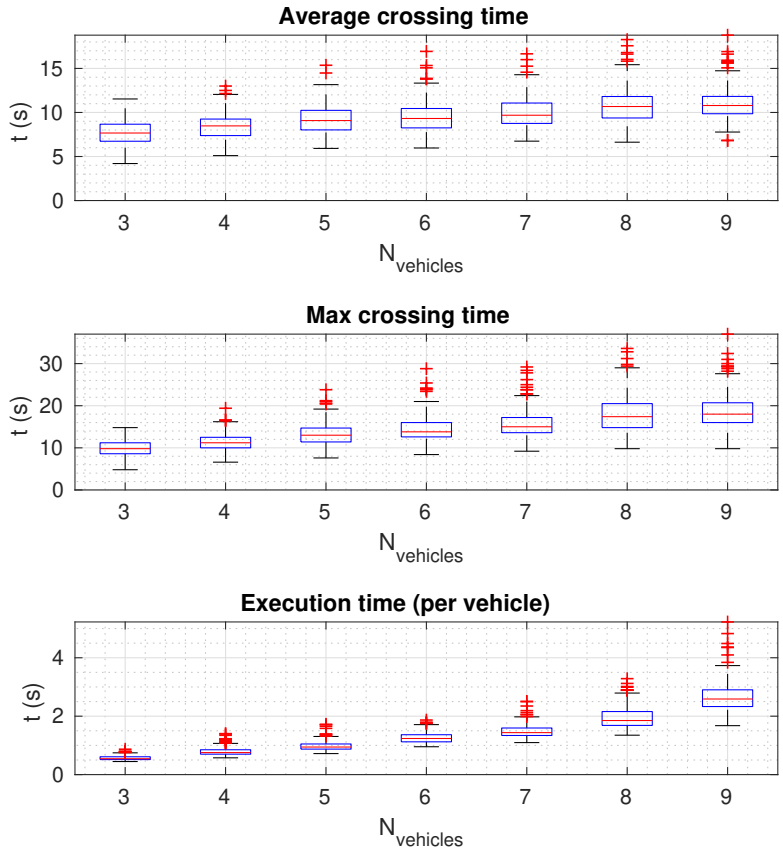

Figure 5.4: Scalability study of the PC algorithm with variable number of vehicles involved in the optimization

The results show a steady increase from $N_{vehicles} = 3$ to $N_{vehicles} = 9$ for both the average crossing time and the crossing time of the last vehicle. The change in crossing time (average and max.) is very limited (approx. 25% for the mean value) whereas the number of vehicles has been multiplied by three. This indicates that the algorithm copes well to make a high number of vehicles pass through an intersection limited in space.

The complexity trend as seen in the bottom plot shows a nonlinear increase in execution time. As the number of vehicles is multiplied by 3, the average execution time is multiplied by approx. 9. The trend up to 9 vehicles is thus polynomial of order 2. A further

analysis of complexity will be carried out in further works, and more realistic execution times will be measured after an optimization of the code and parallelization.

These complexity execution time figures can be used as a reference for the continuous flow simulations demonstrated in Section 5.4. Assuming an acceptable to find a solution, the maximum number of vehicles that can participate in the PC optimization would be given by the data in Figure 5.4. Practical aspects of the implementation of the PC algorithm for intersection crossing can allow to manage and enforce this number and will be detailed in Section 5.4.

5.3.3 Intersection layout effect

In this Monte-Carlo run, the performance of the PC algorithm is comparatively evaluated for a traditional crossroad and a roundabout. As previously, 100 initial situations are generated and are used for both intersection layouts. Figure 5.1 shows the two layouts with a matching initial situation. The vehicles have the same initial starting/destination point, initial position and velocity.

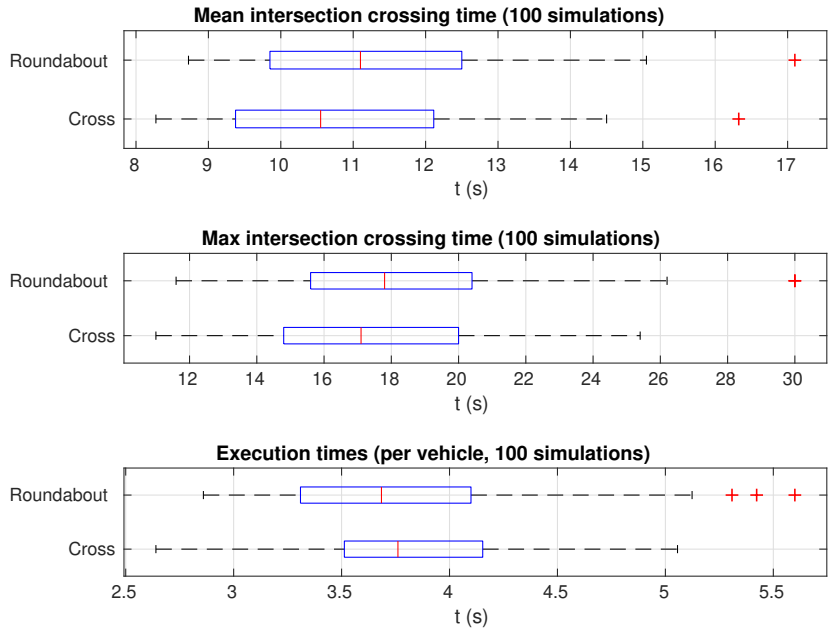

Figure 5.5: Results of Monte-Carlo simulation with 8 vehicles in a 4-way intersection versus a roundabout (matched initial positions)

The results shown in Fig. 5.5 show that the quality of the solution is slightly better in the case of a crossroad in terms of average crossing time and maximum crossing time. The

difference is quite small as the median for the average crossing times are $0.5s$ apart and for the maximum crossing time $1s$ apart. However, the execution time is slightly slower in the roundabout case, indicating a better ease of convergence (although the difference in execution time is weak).

From these results, the proposed association {crossroad, PC algorithm} seems to be (weakly) preferable to the couple {roundabout, PC algorithm} for throughput maximization. This is contrary to what has commonly been seen for roundabouts with human drivers [14]. To be more meaningful, this comparison should be extended to more complex layouts: double/triple lanes and more roads to/from the intersection.

The lack of sensitivity shown by the algorithm to the intersection layout points to good polyvalence characteristics of the algorithm. This in turn points towards its usability in a range of real-life situations without the need for significant infrastructure modification.

5.4 Towards Real-Life Implementation

This section presents a series of improvements around or centered on the PC algorithm to enable its use for continuous traffic scenarios, as opposed to a fixed initial situation with a predetermined number of vehicles.

5.4.1 Ensuring smooth speed profiles for comfortable use

The speed profiles are the possible actions the vehicles can choose from. In this section is detailed an optimal generation of speed profiles for ensuring smoothness while respecting continuity constraints with the current state of the vehicle. It is inspired by the speed profile generation presented as part of the algorithm in [5], and has been adapted to fit the needs of the PC algorithm.

The aim is to design an optimization algorithm that each vehicle can use to generate smooth speed profiles that will be its set of possible actions $\mathbf{X^i}$. For this, a cost function f for the optimization problem should be designed. Different terms are desirable in the cost function to achieve this goal:

- A cost on the input signal (jerk)

- A cost on the acceleration
- A cost on the difference to a specified reference speed v_{ref}

The state-space model chosen for optimal speed profile generation is an integrator. It has been defined in discrete form at a sampling time of Ts as:

$$X_{k+1} = AX_k + BU_k \tag{5.3}$$

Where:

$$X_k = (s_k, v_k, a_k) \quad U_k = (j_k) \tag{5.4}$$

Where the elements of X_k are respectively the curvilinear abscissa, the speed and the acceleration. The input U is the jerk. The state and input matrices A and B are defined as:

$$A = \begin{pmatrix} 1 & Ts & 0 \\ 0 & 1 & Ts \\ 0 & 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 0 \\ 0 \\ Ts \end{pmatrix} \tag{5.5}$$

For easier definition of the quadratic cost function for optimization, three different output matrices are defined for the three states respectively:

$$\begin{aligned} C_s &= \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \\ C_{vel} &= \begin{pmatrix} 0 & 1 & 0 \end{pmatrix} \\ C_a &= \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \end{aligned} \tag{5.6}$$

The vector $\bar{U} = (u_1, ..., u_{N_{opt}})$ of N_{opt} concatenated inputs is defined over a time horizon of length $N_{opt} * Ts$ for the optimization. The goal is to find an optimal input with respect to a quadratic cost function defined in the following paragraph. The model matrices A, B, C_s, C_{vel}, C_a are aggregated in matrices $\bar{\mathbf{A}}$, $\bar{\mathbf{B}}$, $\bar{C}_s$, $\bar{C}_{vel}$ and $\bar{C}_a$ as in [15] and [16]. From the concatenated input $\bar{U}$, the concatenated states over the optimization horizon are defined as:

$$\bar{X} = \bar{\mathbf{A}} * X_{init} + \bar{\mathbf{B}}\bar{U} \tag{5.7}$$

Weight coefficients Q_a and Q_{vel} are defined to penalize the acceleration and speed respectively. These coefficients are concatenated in matrices $\bar{Q_a}$ and $\bar{Q_{vel}}$. A weight coefficient R is also defined for the input (jerk) and aggregated in a weight matrix $\bar{R}$.

The cost function f depending on the input $\bar{U}$ can now be formalized in the following

form:

$$f(\bar{U}) = \sum_{k=1}^{N_{opt}} Q_a a_k^2 + R u_k^2 + Q_v (v_k - v_{ref})^2 \tag{5.8}$$

The function f can be defined as depending only of $\bar{U}$ because of Eq. 5.7.

The usual quadratic function to optimize is of the form:

$$f(\bar{U}) = \frac{1}{2} \bar{U} \mathbf{H} \bar{U} + \mathbf{f} \bar{U}^T \tag{5.9}$$

To implement the function f in Eq. 5.8 in the form of Eq. 5.9, the matrices $\mathbf{H}$ and $\mathbf{f}$ are defined as:

$$\begin{aligned} \mathbf{H} &= (\bar{C}_a \bar{\mathbf{B}})^T \bar{Q}_a (\bar{C}_a \bar{\mathbf{B}}) + (\bar{C}_{vel} \bar{\mathbf{B}})^T \bar{Q}_{vel} (\bar{C}_{vel} \bar{\mathbf{B}}) + \bar{R} \\ \mathbf{f} &= (\bar{C}_{vel} \bar{\mathbf{B}})^T \bar{Q}_{vel} (\bar{C}_{vel} \bar{\mathbf{A}}) X_{init} - Q_{vel} v_{ref} (\bar{C}_{vel} \bar{\mathbf{B}})^T \end{aligned} \tag{5.10}$$

The first term in $\mathbf{H}$ implements the cost on the acceleration, and the last term $\bar{R}$ is for the cost on the input signal. The remainder of the terms in $\mathbf{H}$ and $\mathbf{f}$ are implementing the quadratic cost to the difference to the reference speed v_{ref} (explained in detail in [5].

For the usual quadratic problems, constraints are inserted as:

$$\begin{cases} U_{min} \leq \bar{U} \leq U_{max} \\ A_{ineq} \bar{U} = b_{ineq} \\ A_{eq} \bar{U} = b_{eq} \end{cases} \tag{5.11}$$

The previously defined matrices are used to insert inequality constraints on the input $\bar{U}$, on the acceleration and on the speed. Equality constraints are used to enforce the speed and acceleration at specific points.

Figure 5.6 shows the result of the generation of $N = 10$ speed profiles from an initial state of $v_{init} = 2.2 m/s$ and $a_{init} = -1 m/s^2$. Constraints have been inserted to force the speeds at $t = 4.8s$ to a spread going from $0.1 m/s$ to $2.9 m/s$ through equality constraints. The acceleration has been constrained at the same instant to $0 m/s^2$. The reference speed v_{ref} in Eq. 5.9 has been defined for each speed profile to be equal to the constrained value at $t = 4.8s$. The cost function weights are $Q_a = 1$, $Q_{vel} = 0.5$ and $R = 1$.

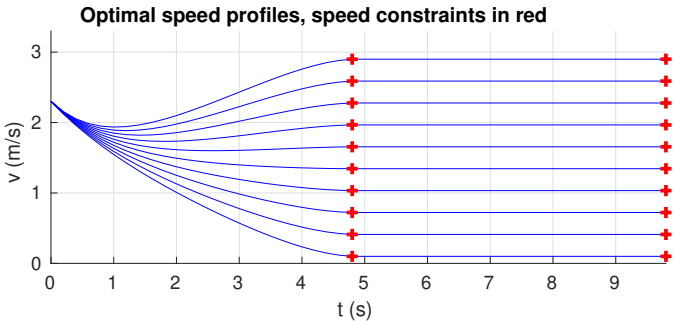

Figure 5.6: Generated smooth speed profiles

5.4.2 Optimization with non-collaborative connected agents

Non-collaborative connected agents are defined as vehicles that communicate their speed profile but will not change it. This could for example be the case of high kinetic energy vehicles (trucks) or emergency vehicles (firefighters, police, ambulances). In section 5.4.3, this property is also used to limit the number of vehicles partaking in the optimization at any given instant. It is one of the tools used to keep a low complexity of the optimization scheme.

Due to the nature of the PC algorithm, the information for a non-collaborative agent j can be inserted in the same way than for collaborative agents, through:

- The set of possible actions $\mathbf{X^j}$
- The associated probability distribution $\mathbf{q}^j(\mathbf{X}^j)$

These two variables completely define the interaction between vehicles within the PC algorithm framework. With a non-collaborative vehicle, these variables just take particular values:

- The set $\mathbf{X^j}$ collapses to one element, that is the planned speed profile of the connected non-collaborative vehicle.
- The associated probability distribution collapses to $q_i(X_i) = \{1\}$.

That means the non collaborative vehicle only has one possible action which probability of execution is 1. The work flow of the algorithm presented in Algorithm 2 is otherwise unchanged.

The capability of the algorithm to cope with connected but non-collaborative agents is highlighted in Fig. 5.7. In this figure, the blue vehicle is connected and non-collaborative. The only possible option it advertises to other agents is that it will keep a constant speed

(cf. blue line on the top plot). As a result, the four other agents do the normal PC optimization taking into account the non-collaborative vehicle. They will either pass before or after the blue vehicle, so that it does not need to modify its speed profile. The blue agent comes from the left and holds a constant speed. It has nonetheless communicated its intentions to the other agents. The vehicles coming from the top lane (yellow and orange) slowed down just enough. Two vehicles coming from the bottom lane (green and purple) found a solution that allowed them to go before the blue vehicle. The last one (pale blue) reduced his speed just enough to go after the blue non-collaborative vehicle.

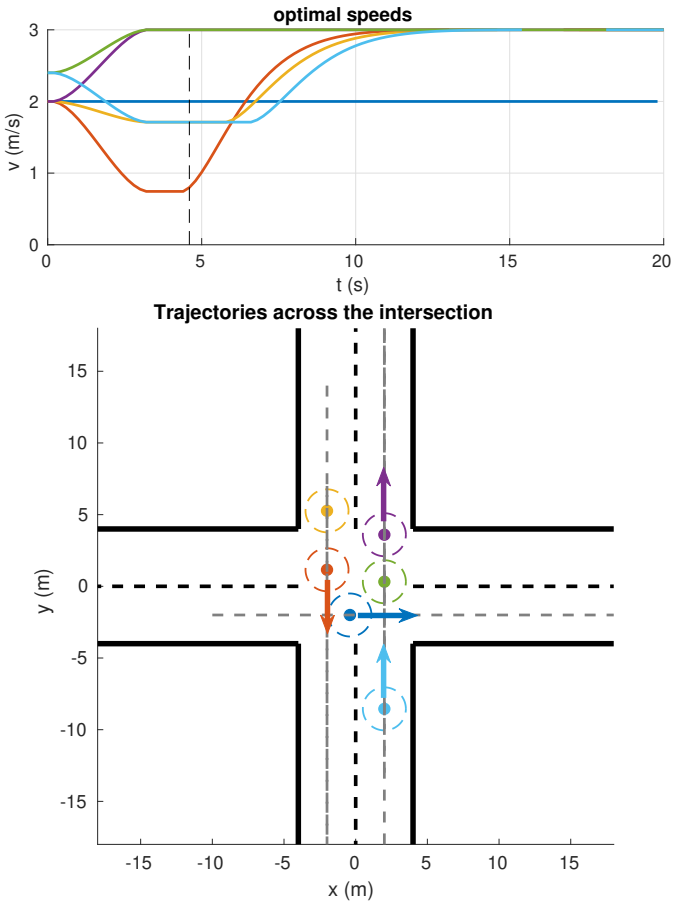

Figure 5.7: Snapshot of the collaborative solution accounting for non-collaborative connected agent in blue, taken at 4.7s. Link to video at `https://youtu.be/bpxXq9G4RTk`

The seamless integration of collaborative agents means that at any moment during the optimization a collaborative agent can become non-collaborative and vice-versa. This property has been demonstrated (albeit for a very different class of problems) in [9]. In this chapter the non-collaboration of agents is not seen as a ”failure” as in the cited paper

but as a feature of the algorithm and framework built around it.

5.4.3 Adaptation to continuous flow at an intersection

To the knowledge of the author, the PC algorithm has not been used yet to do repeated optimizations, in our case to deal with a continuous flow of vehicles. The closest application was [10] for air traffic management, but the algorithm was demonstrated on fixed initial situations.

The first point to address is when to run the optimization and with which vehicles. Fig. 5.8 shows the intersection layout that has been defined to manage the repeated optimizations. Vehicles can enter the distributed optimization scheme only if they are within the synchronization zone. They must have agreed on an admissible solution before entering the shared zone. As the vehicles are not allowed to enter the shared space of the intersection (red) without a valid collaborative solution, the default speed profile planned by the vehicles upon entering the synchronisation zone should be to slowly come to a complete stop just before the shared zone. This condition ensures feasibility when only connected vehicles are present. Within the shared zone, vehicles are allowed to reevaluate their solutions to improve the situation for new incoming vehicles. The length of the synchronization zone shall be enough to allow for a safe stopping before entering the shared zone. It should also be short enough as not to contain too many vehicles for the algorithm to cope with (cf. execution time study on Fig. 5.4)

Assuming some of the vehicles present in the synchronization zone already have an admissible solution, rules have to be defined as to whether these vehicles effectively participate again or just keep their speed profile. Three possibilities have been implemented:

- Single Vehicle optimization (denoted "*single*" mode)
- Full optimization ("*full*")
- Batch optimization ("*batch*")

In the *single* mode, a vehicle runs the PC optimization as soon as it enters the synchronization zone. All the other vehicles are considered to be connected but non-collaborative (cf. section 5.4.2). In this mode, the coordination is sequential more than collaborative since the vehicles decide what to do one by one. This is a great option to reduce complexity in situations with low uncertainty and with only autonomous vehicles. Of course,

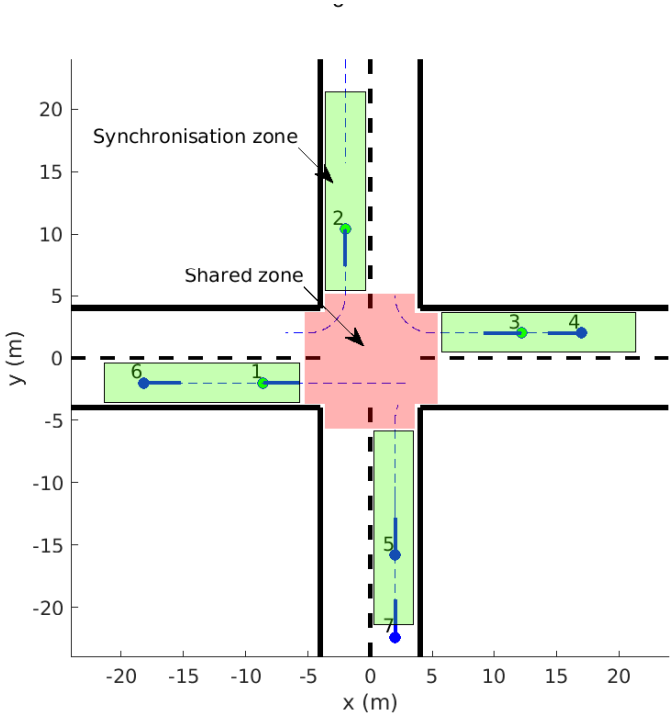

Figure 5.8: Illustration of intersection layout for continuous traffic coordination

if there is an unforeseen event or uncertainty about the action of a vehicle in the intersection, this mode will not work: all the vehicles should recompute what is the best course of action. This is the goal of the *full* optimization mode. This is to be used if such an event occurs, or if any problem arises with the current solution(s). Also, the intersection crossing performance with the *single* mode could be suboptimal as not all the vehicles coordinate with each other. A *full* optimization from time to time would allow to find better solutions. The *batch* mode is a compromise between the two. In this mode, rules are defined regarding which vehicles participate in an optimization. For instance, the optimization could be done in waves of several vehicles once a threshold number is reached (5 vehicles not coordinated yet in the synchronization zone for example). This mode has not been implemented yet, the author relying mostly on the first two for the simulations presented in this chapter.

It is of importance to note that in the case of a human driven vehicle present in the intersection or near it, all the connected autonomous vehicles would have to continuously reevaluate their solutions to account for behaviour of the human driver.

The second point to address is to ensure continuity through the repeated optimizations. If a vehicle has already participated in an optimization and has determined a speed profile to follow, any new speed profile recomputed later should comply with continuity con-

straints for the speed profile. To ensure continuity, the set of possible actions $\mathbf{X_i}$ is defined with the algorithm described in section 5.4.1. This algorithm allows to insert constraints on the speed profile for the initial conditions. Thus, all the possible actions in $\mathbf{X_i}$ will ensure continuity by design, and so will the action $X_{i,k} \in \mathbf{X_i}$ chosen at the end of the optimization.

The second point to address is the time horizon of the optimization. A finite time horizon of $10s$ has been implemented for each vehicle participating in the optimization. it is low enough to reduce complexity, and long enough to ensure that the planning has a high chance of spanning until when the vehicle exits the intersection. If the time horizon is too short, another optimization can be run later. Alternatively, the optimization can be re-run right away with a longer time horizon. The first solution has been implemented to with success so far, as it does not lead to adverse effects.

5.4.4 Demonstration of continuous traffic capabilities

In this section is shown a simulation with all the elements explained through Section 5.4. Vehicles generate their speed profiles with the optimization algorithm described in section 5.4.1. Among the PC optimization modes described in section 5.4.3, the *single* mode has been tested.

In the case of the *single* mode, all the vehicles are non-collaborative except the last vehicle to have entered the synchronization zone. This mode is useful to demonstrate the capabilities of the algorithm to deal with a high number of non-collaborative connected agents. It also demonstrates the use and performance of the algorithm in the special case of single vehicle optimization. On the other hand, the *full* mode shows the exact opposite as it forces all the vehicles to re-run the optimization every time a new vehicle enters the synchronization zone. This mode has been used in the previous section when an optimization is done on a fixed initial situation with all the vehicles. That is why the *single* mode has been preferred here.

Vehicles are spawned on the road in a random manner, similarly to the simulations in Section 5.3. A probability distribution has been defined for the spawning time between two vehicles. The main simulation parameters are summed up in Table 5.3. The sampling time within the PC algorithm is decoupled from the sampling time of the simulation.

In the example simulation, no conflicts happened between any of the vehicles present, even though they did the optimization one by one. Figure 5.9 shows a snapshot of the

Table 5.3: Continuous traffic simulation parameters

Parameter	**Notation**	**Value**
Simulation duration	T_{sim}	$120s$
Sim. sampling time	$T_{s,sim}$	$0.2s$
Start distance to intersection	D_s	$25m$
Spawn time distribution parameters (Normal law))		
Min time between spawns	Δt_{min}	$0.5s$
Min time (same lane)	Δt^0_{min}	$1s$
Avg. time	Δt_{avg}	$1.5s$
Standard deviation	σ_t	$2s$
PC parameters		
PC time horizon	$T_{horiz,PC}$	$10s$
PC sampling time	$T_{s,PC}$	$0.2s$

continuous traffic situation during the simulation with a link to the full video. Green vehicle cores indicate that the vehicle has successfully found a speed profile to cross the intersection with the PC algorithm. The red vectors show the direction and speed of the vehicles. The chosen parameters for the distribution of the spawning times lead to a quite dense traffic situation. Even like that, the *single* optimization mode is enough to ensure that there are no conflicts between vehicles and no jamming. With the ability to perform continuous traffic simulations, it will be possible to evaluate the throughput properties of the algorithm.

5.5 Conclusion

A study of the characteristics of the *Probability Collectives* algorithm for intersection crossing has been presented in this chapter, with the aim of evaluating its applicability to real-life intersection management. The study has explored the behaviour of the algorithm in terms of scalability, sensitivity to road layout, and convergence. Overall the presented PC algorithm for intersection shows sound convergence properties, exhibits a polynomial complexity trend and its performance is mainly insensitive to the road layout. These characteristics show no critical flaw so far for application of the PC to continuous traffic management.

To go further towards such application of the PC algorithm, several steps carried out to make it compatible with real-life scenarios and continuous traffic management. An optimal smooth speed profile generator has been designed, also ensuring continuity of

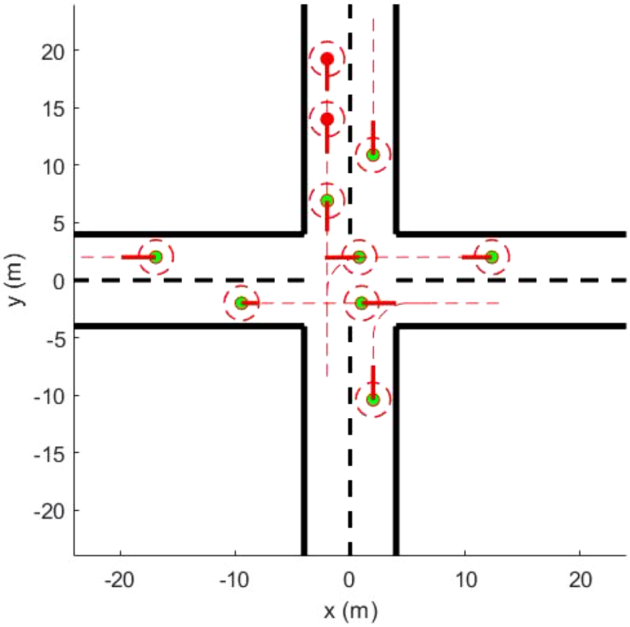

Figure 5.9: Snapshot of the continuous traffic simulation. Link to video at `https://youtu.be/9k_J8E0x_-M`

.

the overall planned speed of the vehicles through repeated PC optimizations. Robustness of the algorithm to non-collaborative connected vehicles has been demonstrated, and an overall framework for real-time traffic management has been designed and demonstrated with Matlab simulations.

Further works will include optimization of the code for faster execution and insertion of human drivers in the intersection.

References

[1] Charles Philippe, Lounis Adouane, Antonios Tsourdos, Hyo-Sang Shin, and Benoît Thuilot. "Probability Collectives Algorithm applied to Decentralized Intersection Coordination for Connected Autonomous Vehicles". In: *Intelligent Vehicles Symposium* (2019).

[2] Daniel J. Fagnant and Kara Kockelman. "Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations". In: *Transportation Research Part A: Policy and Practice* 77 (2015), pp. 167–181.

[3] Scott Le Vine, Alireza Zolfaghari, and John Polak. “Autonomous cars: The tension between occupant experience and intersection capacity”. In: *Transportation Research Part C: Emerging Technologies* 52 (Mar. 2015), pp. 1–14.

[4] Hironori Suzuki and Yoshitaka Marumo. “A New Approach to Green Light Optimal Speed Advisory (GLOSA) Systems for High-Density Traffic Flowe”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. 2018, pp. 362–367.

[5] Robert Hult, Gabriel R. Campos, Paolo Falcone, and Henk Wymeersch. “An approximate solution to the optimal coordination problem for autonomous vehicles at intersections”. In: *Proceedings of the American Control Conference*. Vol. 2015-July. IEEE, July 2015, pp. 763–768.

[6] Stefanie Manzinger and Matthias Althoff. “Tactical Decision Making for Cooperative Vehicles Using Reachable Sets”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. IEEE, Nov. 2018, pp. 444–451.

[7] Changliu Liu, Chung-Wei Lin, Shinichi Shiraishi, and Masayoshi Tomizuka. “Distributed Conflict Resolution for Connected Autonomous Vehicles”. In: *IEEE Transactions on Intelligent Vehicles* 3.1 (2018), pp. 18–29.

[8] Jennie Lioris, Ramtin Pedarsani, Fatma Yildiz Tascikaraoglu, and Pravin Varaiya. “Platoons of connected vehicles can double throughput in urban roads”. In: *Transportation Research Part C: Emerging Technologies* 77 (2017), pp. 292–305. arXiv: 1511.00775.

[9] Anand Jayant Kulkarni and Kang Tai. “A Probability Collectives Approach for Multi-Agent Distributed and Cooperative Optimization with Tolerance for Agent Failure”. In: *Agent-Based Optimization. Studies in Computational Intelligence, vol. 456*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[10] David Šišlák, Pemysl Volf, Michal Pěchouček, and Niranjan Suri. “Automated conflict resolution utilizing probability collectives optimizer”. In: *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews* 41.3 (May 2011), pp. 365–375.

[11] Michalis Smyrnakis and David S. Leslie. “Sequentially updated probability collectives”. In: *Proceedings of the IEEE Conference on Decision and Control*. IEEE, Dec. 2009, pp. 5774–5779.

[12] Anand J. Kulkarni and K. Tai. “Probability Collectives: A multi-agent approach for solving combinatorial optimization problems”. In: *Applied Soft Computing Journal* 10.3 (June 2010), pp. 759–771.

[13] Anand Jayant Kulkarni, Kang Tai, and Ajith Abraham. “Constrained probability collectives with feasibility based rule I”. In: *Intelligent Systems Reference Library* 86 (2015), pp. 73–93.

[14] Srinivas Mandavilli, Margaret J. Rys, and Eugene R. Russell. “Environmental impact of modern roundabouts”. In: *International Journal of Industrial Ergonomics* 38.2 (2008), pp. 135–142.

[15] F Künhe, J Gomes, and W Fetter. “Mobile Robot Trajectory Tracking Using Model Predictive Control”. In: *II IEEE latin-american robotics* (2005), pp. 1–7.

[16] Charles Philippe, Lounis Adouane, Benoît Thuilot, Antonios Tsourdos, and Hyo Sang Shin. “Safe and Online MPC for Managing Safety and Comfort of Autonomous Vehicles in Urban Environment”. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. 2018, pp. 300–306.

Chapter 6

Unified Probabilistic Multi-Vehicle Coordination (UP-MVC): Extending the P-MVC for Real-Life Application and Mixed-Traffic Scenarios

Abstract

Some potential benefits of the PC algorithm that have previously been demonstrated by the author include robustness to agent failure and the possibility to accommodate non-connected vehicles (human drivers). This latter property would be very important with a market penetration of autonomous vehicles < 100%, which includes all short and medium term scenarios for road use. Only the robustness to a specific kind of agent failure has been demonstrated so far [1]. In this chapter are presented improvements to the previously developed PC based *Probabilistic Multi-Vehicle Coordination* (P-MVC) algorithm in order to make it robust to *non-collaborative, non-connected* vehicles. These are classic vehicles driven by humans and without Vehicle-to-Vehicle (V2V) communication technologies. Robustness of the PC algorithm to such agents has never been demonstrated. In this work, it has been achieved by representing human drivers in a probabilistic manner similar to that of the usual PC algorithm.

The resulting algorithm has been called *Unified Probabilistic Multi-Vehicle Coordination* (UP-MVC). It provides a decentralized risk-adverse algorithm to manage mixed-traffic at intersections. Its decentralized nature allows to use it without dedicated infrastructure. The risk-adverse characteristic comes from its probabilistic nature and makes the algorithm choices easy to understand and monitor. These characteristics are demonstrated in the presence of human drivers on the road with Matlab simulations both offline and online (dealing with continuous traffic flow). CAV market penetration rates from 50% to 100% have been tested.

Further works will include testing the UP-MVC on the full range of CAV market penetration (from 0% to 100%) and using it for other maneuvers than intersection crossing as the algorithm in essence does not depend on the intersection layout. These maneuvers can include highway insertion or platooning.

6.1 Introduction

In the field of intersection coordination, several approaches coexist depending on the hypotheses considered by the authors. For instance, coordination can be achieved by changing the traffic lights pattern [2], by assigning slots to vehicles [3] or by direct vehicle control [4]. Among these, only some focus on shorter-term hypotheses in which some human drivers are present on the roads [2]. In this case when not all vehicles are autonomous it is more difficult to find and enforce truly optimal solutions regarding the specific problem objectives (fuel consumption, time for crossing the intersection, ...). Traffic lights pattern optimization is a popular technique in this case.

Other coordination techniques aim to be compatible with mixed traffic while directly controlling the autonomous vehicles. One such category of works rely on mutual exclusion from a shared zone with humans [3] (the centre of the intersection for example). Mutual exclusion techniques are one way of achieving a collision free traffic that is compatible with human driving at the cost of more loss of time, because of the conservative "exclusion" hypothesis. This paper does not work under this hypothesis, while still providing compatibility with mixed-traffic scenarios. This is achieved through the use of a Probability Collectives (PC) based algorithm.

The proposed approach thus provides the mixed traffic compatibility that was reserved until now to traffic lights based human-compatible intersection coordination or exclusion based methods, while having performance real optimal coordination algorithms not compatible with mixed traffic. Another significant benefit of the proposed algorithm is its explicit and easily understandable risk-averse behaviour, based on its probabilistic data representation [5].

This chapter describes the algorithmic framework and improvements built upon the P-MVC algorithm of Chapter 4 and 5 in order to develop a unified algorithm for multi-vehicle coordination. Several big modifications to the existing algorithm were needed.

The remainder of this chapter is organized as follows:

- Section 6.2 shows the generalization of the PC framework to mixed traffic scenarios including human drivers.
- Section 6.3 presents simulation results demonstrating the abilities of the PC algorithm to deal with continuous traffic management and human drivers on the road. The behaviour of the algorithm is analyzed and the main performance metrics of

the UP-MVC are discussed.

6.2 UP-MVC: Extension of the Algorithm to Mixed-Traffic Scenarios

In this section are presented the developments that made the proposed PC-based algorithm a really unified solution for intersection crossing with all types of vehicles:

- Connected collaborative vehicles
- Connected non-collaborative vehicles, also described as "stubborn" (Section 5.4.2)
- And now, also with non-connected, non-collaborative vehicles. This class of vehicles has unknown intentions from the point of view of the connected vehicles, as they do not communicate them and may not collaborate with the other vehicles. In real life, those characteristics correspond mostly to classic vehicles driven by humans.

The last class of vehicles will be referred to as "human driven vehicles" in the remainder of this chapter as it is more easily relatable.

The resulting algorithm have been called Unified Probabilistic Multi-Vehicle Coordination algorithm (UP-MVC). Its aim is to provide an optimization engine working on probabilistic data to come up with decisions on which the level of risk can be understood by humans and monitored. As previously stated, it is a polyvalent algorithm with respect to the types of vehicles on the road. It is also flexible with regards to the intersection layout. The UP-MVC has been demonstrated on simulations containing all three types of vehicles, both offline and online.

6.2.1 Computational aspects of dealing with non-collaborative, non-connected agents

This section is about using the algorithm with human driven vehicles in the intersection or more generally in the zone shared by all vehicles. The human drivers are characterized by the fact that:

- They do not collaborate (not taking part in the collaborative decision).

- They are not connected (no information or very little is available regarding their intentions).

The PC algorithm has the potential to deal with this type of situation because of its probabilistic formulation. Its robustness to a specific type of failed agents has been demonstrated in [1], although for an offline optimization of a mathematical problem. Integrating human-drivers in the optimization goes beyond that for two reasons:

- It is dynamic (rapidly changing)
- The action of human drivers are inherently unknown

For the second point, educated guesses can be built through observation of the current and past states of the agent (speed, turn signals, ...) and assumptions about the behaviour (is the vehicle arriving at a stop? A green light?).

To make the PC algorithm compatible with the presence of such agents, a set of possible actions $\mathbf{X^i}$ has to be generated along with a probability distribution $\mathbf{q_i}(\mathbf{X^i})$ on $\mathbf{X^i}$.

The focus of this work is not on the set generation per se. It focuses on characterizing what quality of information is needed for the algorithm to come up with relevant solutions. Thus, the output of this work can be seen more as a set of design requirements for:

- The intersection layout (will condition the knowledge and assumptions about human drivers)
- The algorithm that will effectively generate those sets
- The adaptations required for continuous traffic flow optimization

6.2.2 Possible action prediction for human-driven vehicles

The work here is focused on the characteristics of the prediction necessary for the PC algorithm to work. It can be seen as a list of design requirements for the conception of the predictor of possible actions.

As designed, each possible action is a full trajectory composed of a 2D path and a speed profile. For now the 2D paths have been set to be fixed and known. This hypothesis is realistic because a lot of visual clues are available to know the intended path of a vehicle such as the lane it is in and whether the turn signals are on. With the possible 2D paths fixed, the notations for the possible actions of humans can be kept simple and consistent.

For a human driven vehicle i, the set of N possible actions considered is $\mathbf{X^i} = (X_1^i, ..., X_N^i)$ where each X_k^i is a speed profile over a given time horizon in the future.

The possible speed profiles have been designed for continuity of speed and acceleration. The targeted human driven vehicle is assumed to have been perfectly observed. The range of possibilities considered is shown in Fig. 6.1.

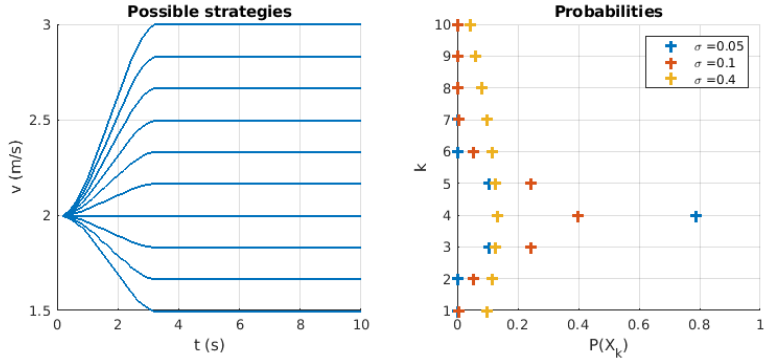

Figure 6.1: Example of strategies hypotheses and discrete probability distribution

.

The probabilities shown on Fig. 6.1 have been generated through a discretization of a normal law of known standard deviation. The distribution has been centered on the speed profile that makes the vehicle keep its initial speed. The vehicle is deemed unlikely to either accelerate much or decelerate much.

The dimensionless parameter σ is akin to a standard deviation and is an input of the behaviour prediction function. It is used to artificially change the degree of certainty assumed about a human driver's behaviour to see the effect it has on the algorithm. It is expected that a higher standard deviation (more uncertainty on the human driver behaviour) would lead to the UP-MVC algorithm onto a more conservative behaviour (for example by giving more space to the human drivers). The probabilities of each discrete action are computed from a continuous normal probability distribution:

- Centered on $i_{central}$, the index of the strategy that is deemed the most likely.
- Of standard deviation $N_s\sigma$, where N_s is the number of possible strategies considered.

The probabilities are then normalized so that their sum is equal to 1, in order to give the discrete probability distribution $\mathbf{q}^i(\mathbf{X}^i)$. The $\mathbf{q}^i$ notation has been kept to be consistent with the previous definition of the PC algorithm.

This action prediction technique has been kept consistent through this section so that the results are more easily understandable, and any effect observed will be a result of the PC algorithm behaviour.

6.2.3 Criteria for comparing solutions

Assuming an ego-vehicle is i, the sets of possible actions $\mathbf{X^j}$ and associated probabilities $\mathbf{q}^j(\mathbf{X}^j)$ are:

- **known** for all $j \neq i$, j describing a CAV.
- **estimated** for all $j \neq i$, j describing a human driven vehicle.

As described in Algorithm 3, the possible actions of each vehicle j will be sampled according to their probability distribution. These samples will serve to compute the expectancies $\mathbf{E}(X^i)$ and ultimately the favourite self-strategy of the ego-vehicle [5].

The PC algorithm is used here in a semi-centralized formulation presented in Algorithm 3. Solutions found by the connected vehicles are compared with each other to find out which vehicle found the best overall strategy at the current step. This formulation allows to more closely monitor the feasibility of the solution at each step instead of letting behaviours emerge as in [1]. The main contribution for dealing with human drivers in the optimization algorithm is related to how the costs are compared between vehicles to know which connected vehicle has found the best solution.

The preferred self strategy $X^i_{opt,tmp}$ is computed based on cost expectancies associated with each self possible action $X^i_k \in \mathbf{X^i}$. Those cost expectancies are computed using the expected actions of other vehicles, including human driven vehicles.

Even though the preferences of human driven vehicles and connected vehicles are both expressed through the same format $(\mathbf{X^j}, \mathbf{q}^j(\mathbf{X}^j))$, they mean very different things. For connected vehicles those preferences are actually a degree of freedom of the optimization, and the favourite action in $\mathbf{X^j}$ will be the one actually carried away. For human driven vehicles, the couple $(\mathbf{X^j}, \mathbf{q}^j(\mathbf{X}^j))$ represents a mere guess of the future behaviour of the vehicle: the most likely action in $\mathbf{X^j}$ may not even happen and the connected vehicles have no influence on it.

Thus, a joint strategy Y can only be defined for the set of connected vehicles that participate to the optimization. With human drivers inserted in the optimization, the cost of a joint strategy Y is composed of two very different terms:

Algorithm 3 Semi-centralized PC optimization outline

Data: Own set of possible actions $\mathbf{X^i}$
Data: Current joint strategy proposed by vehicle i: $Y^i_{opt,tmp}$
Data: Best joint strategy found so far by any vehicle: Y_{opt}
Result: Joint strategy Y_{opt}
Initialize $\mathbf{q}^i(\mathbf{X}^i)$ to a uniform distribution
Initialize the SA temperature T
while no convergence of $\mathbf{q}^i(\mathbf{X}^i)$ **do**
  for vehicles $j \neq i$ **do**
    if j is connected **then**
      Get set $\mathbf{X^j}$ from communication
      Get $\mathbf{q}^j(\mathbf{X}^j)$ from communication
    else
      Estimate $\mathbf{X^j}$
      Estimate $\mathbf{q}^j(\mathbf{X}^j)$
    end if
  end for
  for each X^i_k in $\mathbf{X^i}$ **do**
    for $j \neq i$ **do**
      Randomly sample some $\mathbf{X^j}$ based on the probabilities $\mathbf{q}^j(\mathbf{X}^j)$
    end for
    Compute expected utility $E(X^i_k)$ of X^i_k
    (based on the sampled strategies for other vehicles)
    Store $E(X^i_k)$ in a vector $\mathbf{E}(X^i)$
  end for
  Find $\mathbf{q}^i(\mathbf{X}^i)$ minimizing $f(\mathbf{E}(X^i), T)$
  Store action $X^i_{opt,tmp} = argmax(\mathbf{q}^i(\mathbf{X}^i))$
  Build $Y^i_{opt,tmp}$ by concatenating $X^i_{opt,tmp}$ with known preferrences for vehicles $j \neq i$
  if $Y^i_{opt,tmp}$ is better than Y_{opt} **then**
    Propose $Y^i_{opt,tmp}$ to other vehicles
    Compare $Y^i_{opt,tmp}$ to other new proposals $Y^j_{opt,tmp}$ $(j \neq i)$
    Store best proposal as Y_{opt}
  end if
end while

- A **deterministic cost** cost $J_D(Y)$ related to the quality of the joint strategy if no human driven vehicles were there. This cost is the same as described in [5] for a PC application without human driven vehicles. The cost includes a weighted sum of terms related to: the average speed through the intersection, the separation distance with the other vehicles and the control effort that the solution requires.
- A **cost expectancy** $E(J_H(Y))$ related to the interaction of connected vehicles with human driven vehicles. The notation J_H is used because the cost function is slightly modified from J that was defined in [5]. The subscript H is used because it is related to "humans".

The objectives to fulfil during the optimization are very different with respect to human driven vehicles versus connected vehicles. Among connected vehicles, the aim of a strategy is threefold as described in [5]. A good solution is a balance of separation with other vehicles, high average speed through the intersection and low control effort for the passengers comfort. With respect to human driven vehicles, the single major objective is to ensure safety through keeping a safe distance. It is logical to expect that the more uncertain the behaviour of the human driver is, the bigger the separation distance should be kept. Concepts such as average speed through the intersection and control effort do not apply to the human driven vehicles in the scope of the collaborative optimization. Thus, the resulting cost is denoted J_H and expressed as:

$$J_H(X) = W_{sep} \sum_{j \in C_{indexes}} \sum_{k=1}^{N} \frac{1}{d_k(j,i)^2} \tag{6.1}$$

Where N is the prediction horizon and $C_{indexes}$ is the set of indexes of the connected vehicles present in the intersection.

To account for constraints violation, a high cost penalty J_{cons} is added to the cost for each $d_k(j,i) < d_{safe}$ (where d_{safe} is a minimal separation distance to the human driven vehicles). For a number of violated separation constraints $N_{collisions}$, the cost $J_H(X)$ becomes:

$$J_H(X) = W_{sep} \sum_{j \in C_{indexes}} \sum_{k=1}^{N} \frac{1}{d_k(j,i)^2} + N_{collisions} J_{cons} \tag{6.2}$$

For each human driven vehicle V_i present in the intersection, N_s strategies are sampled among the possible strategies set $\mathbf{X^i}$ of V_i. The separation cost J_H is then evaluated. As each tested strategy X_k^i for the human driven vehicle V_i is associated with a probability

q_k^i, the partial expectancy $E^i(J(Y))$ can be computed. The term partial is used because $E^i(J_H(Y))$ is the cost expectancy due only to the human driven vehicle V_i with respect to all connected vehicles.

When computed for all vehicles, the cost expectancy $E(J_H(Y))$ is then defined as:

$$E(J_H(Y)) = \sum_{i \in H_{indexes}} E^i(J_H(Y)) \tag{6.3}$$

For a joint strategy Y of the connected vehicles. The set $H_{indexes}$ denotes the indexes of human driven vehicles.

The penalty J_{cons} in Equation 6.2 is usually several orders of magnitude bigger than W_{sep}. Thus, when monitoring the cost expectancy, it becomes easy to see if some outcomes for the human driver's behaviour lead to collisions with the current test joint strategy Y.

It is important to remember that the cost expectancy $E(J_H(Y))$ due to humans depends on the actual random sampling of human driven vehicle's strategies. When comparing $E(J_H(Y))$ between two solutions, the sampling needs to be exactly the same as not to favour unfairly one strategy against another one.

The following criteria have been implemented to decide which of the proposed $Y^i_{opt,tmp}$ is best in Algorithm 3:

- First of all, the feasibility of $Y^i_{opt,tmp}$ is computed as if only connected vehicles existed. The strategy with strictly less constraint violations wins.
- For strategies $Y^i_{opt,tmp}$ with the same number of constraints violations, the deterministic cost $J_D(Y)$ and human related cost expectancy $E(J_H(Y))$ are used to determine which is better.
- To strike a balance between optimality and robustness during the comparison of solutions, the sum of $J_D(Y)$ and $E(J_H(Y))$ has been used as a criteria.

With these adaptations, the transformation of the P-MVC algorithm into the UP-MVC has been fully carried away. The UP-MVC is able to consider human driven vehicles in the optimization in a probabilistic manner. In Section 6.3, The algorithm will be demonstrated for offline situations to check and explain its behaviour. An extension to online use will be presented.

6.2.4 Probability of disruption

One of the criteria of success of the algorithm (and any optimization algorithm) is to find a solution that minimizes the cost function that has been defined.

Another success metric has been developed for the UP-MVC algorithm in order to evaluate the quality of the solution with respect to the human drivers potential actions.

When the algorithm stops, the final cost expectancy $E(J_H(Y))$ defined in 6.2.3 can be memorized, as well as the following quantities for each human driven vehicle V_j ($j \in H_{indexes}$):

- For each possible strategy X_k^j of V_j, a boolean b_k^j denoting the feasibility of X_k^j with respect to separation constraints with any of the connected vehicles.
- The sum of all probabilities of the strategies of V_j that lead to such violation. This is the probability that V_j does something the joint strategy Y has not been planned for. This probability is denoted $q_v^j = \sum_{k \in I_{collision}} q_k^j$, where q_k^j is the probability of X_k^j and $I_{collision}$ is the set of all k so that $b_k^j = 1$, for $X_k^j \in \mathbf{X^j}$ (in other words, all the X_k^j that lead to a separation constraint violation).

Assuming that the behaviour of the human driven vehicles is independent (which is a pessimistic hypothesis, more on that later), the total probability that the joint strategy Y "goes wrong" with respect to the separation constraint with human drivers is then:

$$Q_v = \sum_{j \in H_{indexes}} q_v^j \tag{6.4}$$

Of course, humans will do their best to avoid collisions so this probability should be seen as the likelihood that the humans are set on a colliding trajectory at the current time, and thus the likelihood that they will have to reconsider their actions at a future time. In other words, this can be seen as a probability of "disruption" of the human drivers. The lower this probability is, the better. With an accurate enough knowledge of the intentions of the human drivers, this probability would actually be very close to zero.

This would be an approximation of the likelihood of collision if the humans were "blind" and did not ever change their course in reaction to the situation. As a consequence, this probability should be seen as no better than a coarse upper bound of the probability of collision at any given time.

Parameter	Notation	Value
Number of strategies	N_s	10
Sampling time	T_s	$0.2s$
Weight on control effort	$W_{control}$	0
Weight on separation dist.	W_{sep}	1
Weight on avg. crossing time	W_{avg}	10
Penalty for constraint violation	J_{cons}	10^5
Samples to get expected utility	$N_{samples}$	10
Stopping criteria	N_{stop}	4
SA start temperature	T_{init}	10
SA end temperature	T_{end}	0
SA temperature step	T_{step}	0.2
Degree of certainty for human drivers	σ	0.5

Table 6.1: Main parameters for UP-MVC proof of concept

6.3 UP-MVC Demonstrations and Experiments

6.3.1 UP-MVC for mixed traffic: proof of concept

The simulation presented here is a proof of concept of the integration of human driven vehicles in the optimization algorithm. A set of three vehicles has been defined. The vehicle V_1 in blue is human driven, comes from the left and goes to the right. Vehicles V_2 (orange) and V_3 (yellow) are autonomous and have opposite vertical trajectories. They both participate to the offline situation.

The main PC parameters are shown in Table 6.1.

The assumptions for the human driven vehicles are as follows:

- The initial speed of $2m/s$ is perfectly observed
- The range of possible speeds is assumed to go from $1.5m/s$ to $3m/s$: that means the human is assumed to not stop (i.e. he has the priority)
- The most likely action is for him to keep a constant speed. The provided video of the solution will show this action.
- The standard deviation that describes the quality of the knowledge is $\sigma = 0.5$, which is a quite poor distribution as shown in Figure 6.2

On Figure 6.3 is shown a snapshot of the solution when the human driver (in blue) actually applies what was deemed the most likely hypothesis by the autonomous vehicles (i.e.

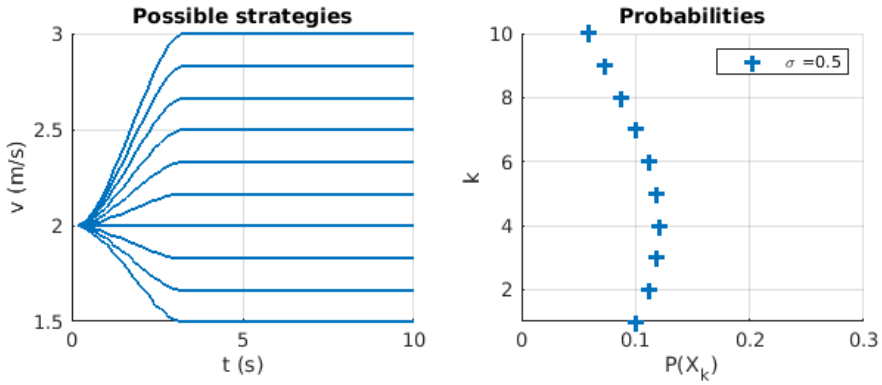

Figure 6.2: Possible strategies for V_1 and associated probabilities

constant speed, as shown on Fig. 6.2). The full video is available at `https://youtu.be/wNOiwajG4tA`.The autonomous vehicles leave some space to the human driven vehicle in case their prediction had been wrong. If V_1 had gone slower than the hypothesis deemed most likely, the vehicles V_2 and V_3 would still have had a valid solution with enough space left as they did not rush behind the blue vehicle. As V_1 was deemed to have a possibility to accelerate to $3m/s$, both vehicles decided not to go in front of it.

For this simulation, the total probability of disruption is $Q_v = 0.101$. So there is an estimated 10% chance that the human driver has to change its behaviour. This quantifies the degree of acceptability of the solution found by the algorithm, assuming the behaviour estimation is correct. The majority of optimizations with human drivers lead to $Q_v < 0.1$, meaning that the algorithm has been implemented in a way that favorizes low chances of disruption of the human drivers.

For comparison purposes, Fig. 6.4 shows a solution generated with a much higher degree of certainty on the actions of the human driven vehicle (video available at `https://youtu.be/q9dQPt0LflQ`). In this case the yellow vehicle (V_3) decides it is safe enough to go in front of the human driven vehicle. The orange vehicle V_2 leaves much less space at the back of V_1. In this case the probability of disruption computed by the optimization algorithm is $Q_v = 0$. However, the certainty has been set very high. In real life, this level of certainty is unrealistic, and the solution applied by the autonomous vehicles would actually influence its behaviour (braking when the yellow goes in front). This latter effect would actually be mitigated by enforcing a higher minimal separation distance to humans such as their driving would not be disrupted even when the knowledge is considered certain.

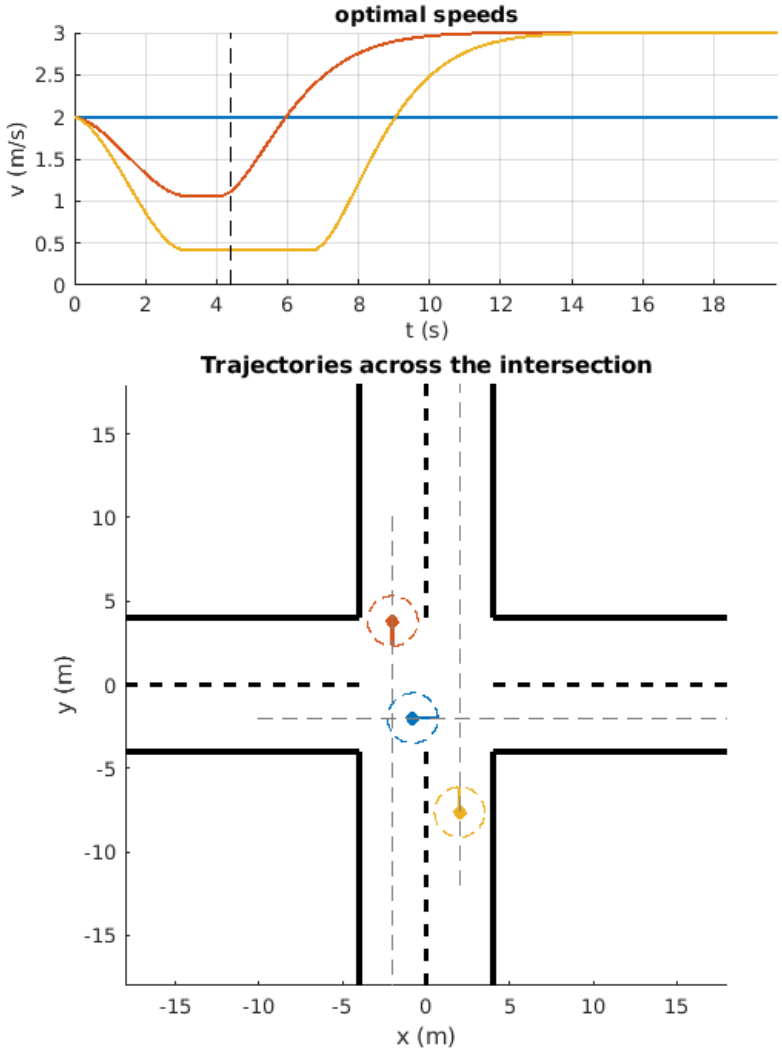

Figure 6.3: Snapshot of the application of the best solution when $t = 4.3s$

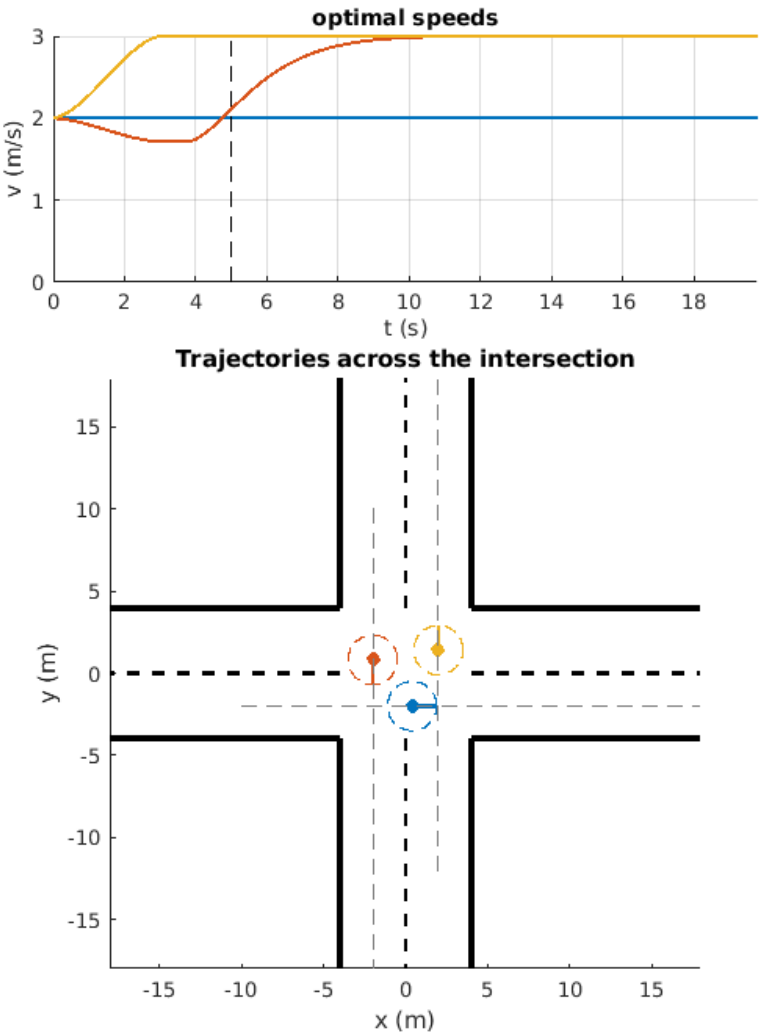

Figure 6.4: Snapshot of the application of the best solution when $t = 5s$ (high certainty)

6.3.2 Study of probability distribution standard deviation

As previously mentioned, the actual solution found by the algorithm is heavily influenced by the degree of certainty associated with the behaviour prediction for the human drivers. This section investigates this relation. It is expected that the less sure the knowledge is about the human drivers, the more separation the autonomous vehicles will observe.

More precisely:

- If the predictions are very broad (lots of actions possible with equal probability), the algorithm will be overly conservative and tend to lead to slow solutions.
- If the predictions are very specific (one action considered certain, probability 1), the solution will be very optimal. However, in this case the UP-MVC algorithm would not really be needed: the non collaborative vehicle behaves as if it was connected because its actions are known (cf section 5.4.2).

To evaluate the behaviour of the algorithm with respect to the degree of certainty in the human driver's behaviour prediction, a series of simulations have been carried away. The initial situation has been fixed to the one presented in Section 6.3.1. For each simulation, the standard deviation σ defining the spread of probabilities for the strategies set of the human driver has been picked randomly in $[0.02, 0.5]$. The chosen output metric is the average distance left by the autonomous vehicles to the human driver when at their closest point. The expected result is that the more uncertain the knowledge is, the more space will be given to the human driven vehicle V_1. Figure 6.5 shows the results obtained for 100 simulations.

It appears that the distance left to the humans is closely correlated to σ up to $\sigma = 0.15$. Then, a saturation phase appears where the distance is constant. It means the algorithm saw no additional benefit to give more space to the human driver, because the probability of collision was low enough. In terms of the distances shown on the figure, the reader should remember that the test case is the "IPCar" vehicles on the PAVIN platform. These vehicles go at speeds of up to 3m/s, are 2m in length and 1m in width. Thus, round vehicles of radius of 1.5m have been implemented in the simulations. A separation distance of 4m (centre to centre) means that 1m has been left between the vehicles.

It is also interesting to note that there are outliers to the top (high certainty and high separation distance) but not to the bottom of the curve which means the algorithm is sometimes too conservative but has not exhibited a "reckless" behaviour. This, and the correlation of σ with the separation distance are very important and novel properties that

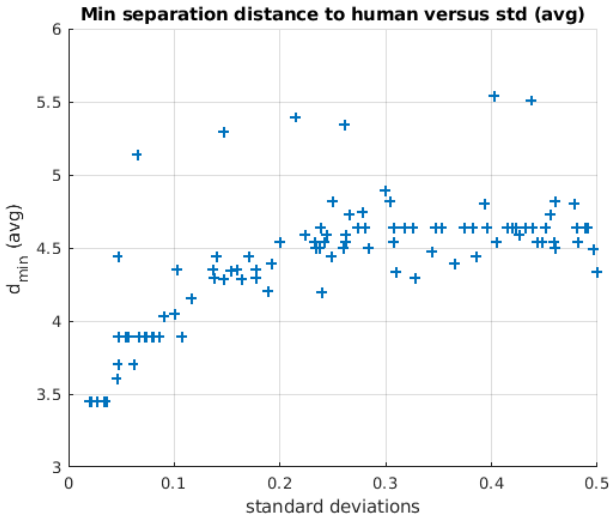

Figure 6.5: Relation between standard deviation of behaviour estimation and space given to human drivers.

show the suitability of the UP-MVC algorithm to mixed traffic scenarios.

6.3.3 Insertion of human-driven vehicles in online continuous flow simulations

In this section are shown simulations with continuous traffic management with the UP-MVC algorithm. This time, the simulations include a proportion of human drivers to demonstrate the capabilities of the UP-MVC algorithm to deal with human drivers in a continuous traffic optimization.

For the simulation, vehicles are spawned randomly with a given probability of being human driven. A rolling time horizon has been implemented for the optimization. A vehicle will recompute its solution at regular intervals if it does not reach the intersection exit within the optimization horizon.

An Adaptive Cruise Control (ACC) has been implemented as a crude way to emulate the behaviour of human drivers. Their path has been fixed in advance but the actual speed profile depends on what happens during the simulation. The ACC implemented comes from [6]. The main ACC equation is as follows:

$$a_{ref} = K_1(\Delta d - t_h v_{self}) + K_2 \dot{\Delta} d \tag{6.5}$$

Where a_{ref} is a reference acceleration, Δd is the distance from the ego vehicle to the

Parameter	**Notation**	**Value**
Simulation duration	T_{sim}	$60s$
Sim. sampling time	$T_{s,sim}$	$0.2s$
Start distance to intersection	D_s	$25m$
Spawn time distribution parameters (Normal law)		
Min time between spawns	Δt_{min}	$0.5s$
Min time (same lane)	Δt^0_{min}	$1.5s$
Avg. time	Δt_{avg}	$1.5s$
Standard deviation	σ_t	$2s$
PC parameters		
UP-MVC time horizon	$T_{horiz,PC}$	$10s$
UP-MVC sampling time	$T_{s,PC}$	$0.2s$
ACC desired headway	t_h	1s
ACC coefficients	$[K_1, K_2]$	$[1, 3]$
ACC acceleration bounds	$[a_{min}, a_{max}]$	$[1, -3]$

Table 6.2: UP-MVC parameters for continuous traffic simulation with human drivers

vehicle in front, t_h is the desired headway time, v_{self} is the ego vehicle speed and $\dot{\Delta}d$ is the time derivative of Δd.

The main simulation parameters and ACC parameters have been summed up in Table 6.2.

For the simulation, a priority rule has been set up for the human drivers. Vehicles who want to enter the intersection have to yield to the vehicles already in it. This has been implemented through the ACC formula, by projection the position of vehicles in the intersection onto the path of the ego-vehicle. The projection is then considered as a real vehicle and used to compute a reference acceleration with the ACC.

Of course, the behaviour of autonomous vehicles is not bound to those rules as they optimize the way they cross the intersection. In addition to that, the hypothesis deemed most likely for the human drivers is that they keep a constant speed. The consequence of choosing this hypothesis is that autonomous vehicles will tend to be careful about human drivers entering the intersection, as they expect them not to observe the priority. This choice has been done for two reasons:

- It allows to demonstrate the behaviour of the algorithm in a dynamic environment when the predictions happen to not match the reality.
- It helps to generate a very conservative behaviour for the autnomous regarding the human drivers, which can be considered as a "priority by default" for the humans.

For a more realistic behaviour of the connected vehicles, a reactive collision avoidance has been used within the intersection when interacting with human driven vehicles. The ACC has been used, as for the human driven vehicles when they have to yield before the intersection.

The following rules have been set up for when and which autonomous vehicles should participate in the collaborative optimization:

- By default, the *single* mode is used (defined in Section 5.4.3). When a vehicle enters the synchronization zone upstream of the intersection, it runs the UP-MVC alone while considering autonomous vehicles already in the synchronization zone as stubborn.
- A fixed time duration has been implemented after which a vehicle will participate again in an optimization. This is to make sure that an optimization has been done with recent predictions for the human drivers. In the simulations presented, this duration has been set to $1s$. This is also a form of *single* mode optimization.
- When a human driven vehicle enters the synchronization zone, an optimization is run for all vehicles present in the intersection (*full* optimization). These optimizations are more computationally intensive.
- When no feasible solution has been found between a vehicle that participated in the optimization and a stubborn vehicle (a vehicle which does not want to change its trajectory), a "soft" conflict solving mechanism is applied. The optimization is re-run for the two conflicting vehicles, with neither of them being granted the "stubborn" status. Usually this increases enough the degree of freedom of the optimization to come up with a solution. In case this *batch* recomputation did not come up with a feasible solution, a *full* optimization is run.

A snapshot of the simulation is shown in Figure 6.6. In this figure, connected vehicles are represented in blue, with a green core when they have successfully finished one or more optimization. Human driven vehicles are in red. The full video of the continuous traffic simulation with 20% of human drivers is available at `https://youtu.be/h3H57iAFvXw`.

The speed profiles of some of the connected vehicles present in the optimization are shown in Figure 6.7. It can be seen that V_2 and V_3 keep a constant speed equal to $3m/s$ to cross the intersection. The vehicles V_4 and V_5 participated to one round of optimization and thus their speed profile is very typical: slowing down to a constant speed that solves the

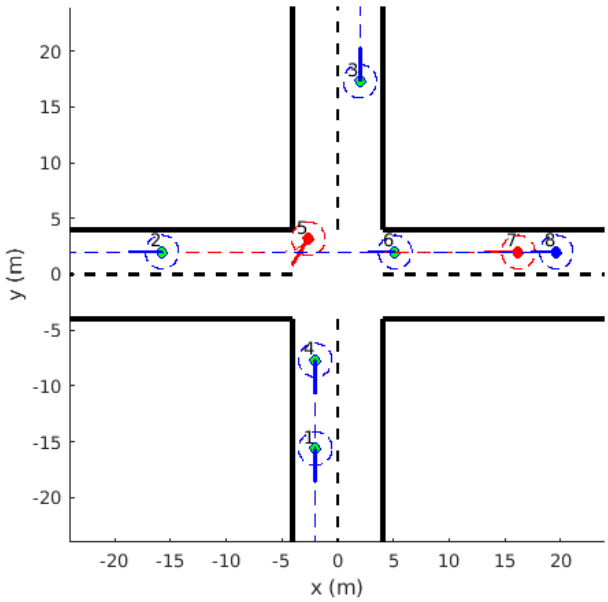

Figure 6.6: Snapshot of a continuous traffic flow simulation with 20% of human drivers

conflicts that could arise during the crossing, and then reacceleration when possible to v_{max}. The vehicle V_6 has a more tortured speed profile. It participated to collaborative optimizations at $t = 13s$, $t = 16s$, $t = 18s$, and $t = 25s$. It is noticeable on the figure because these durations correspond to changes in accelerations. The pattern observed between $18s$ and $25s$ is the result of a single optimization. As a reminder, each round of the UP-MVC algorithm is done in two steps: a first optimization that finds a target speed to solve the conflict, and second step in which the algorithm looks for a reacceleration in order to clear the intersection faster. The target speed found at the first optimization step is reached at $t \approx 20s$, and a reacceleration to $1m/s$ is then applied. There is a discontinuity in the acceleration at $t = 32s$ for V_6 which is not due to the UP-MVC algorithm: it happens when the vehicle switches back to a pure ACC behaviour upon exiting the intersection. As what happens after the exit is of little interest for the present research, continuity constraints have not been enforced there.

Overall, this figure demonstrates the continuity of the speed profiles applied by the autonomous vehicles, whether they participate in a single UP-MVC round or several due to the rules mentioned above. Even in the case of V_6, there are no ”changes of mind”, in which the vehicle would repeatedly decelerate/accelerate/decelerate. Such a behaviour is highly desired for the maneuvers to be understandable for the passengers.

An histogram of the intersection crossing times for all vehicles is shown in Figure 6.8.

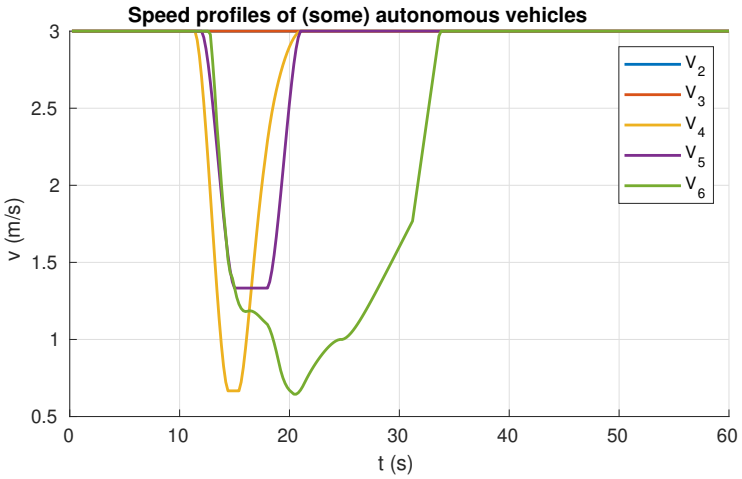

Figure 6.7: Speed profiles of some of the autonomous vehicles

The crossing time here has been computed between the entrance of a vehicle in the synchronization zone, and its exit from the intersection. The synchronization zone starts $10m$ upstream of the intersection entrance, and the path length to cross the intersection is in average $8m$. So a crossing time of $6s$ would be achieved in average for a vehicle that keeps its maximal speed of $3m/s$. Shorter times are achieved for some vehicles that do a right turn at full speed and thus have less distance until the intersection exit. It can be seen that most of the vehicles have a crossing time between $4s$ and $11s$. The average is $7.6s$ and the standard deviation $3.7s$. These results are on par with those presented in [5] for an initial situation with 6 vehicles (connected vehicles only). In the present case, the environment is changing and presents stubborn vehicles as well as non-collaborative vehicles (human drivers). It shows that the performance of the PC algorithm holds well even for highly dynamic environments.

For comparison and information purposes, the proportion of human driven vehicles has been tested up to 50%. However, since the human drivers behaviour has been implemented in a very simplistic way, collisions start to arise between human drivers. The connected vehicles however did not have any collisions either between them or with the human drivers. A video is available at `https://youtu.be/rL8Wn8s--i8`. While the UP-MVC algorithm is still coping well, the demonstration would be better served by a real human behavioural engine at those proportions of human drivers on the road.

With higher proportions of human drivers on the road, the collaborative aspect of the algorithm will be less important as most of the optimizations will be done for a single autonomous vehicle. Nevertheless, the UP-MVC is still a suitable algorithm for these situations because of its polyvalence, its probabilistic nature and its explicit success metrics

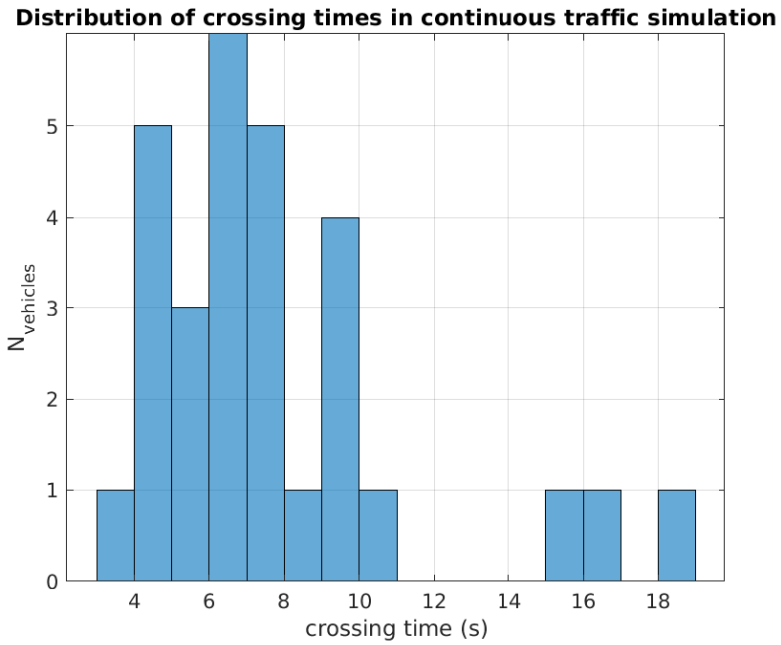

Figure 6.8: Repartition of intersection crossing times

such as the *probability of disruption*.

6.4 Conclusion

A unified algorithm for optimal intersection crossing compatible with mixed-traffic scenarios has been proposed. It is based on the Probability Collectives algorithm, which is a decentralized optimization algorithm that was designed for solving NP-hard problems.

The UP-MVC brings the performance of pure optimization algorithms to mixed traffic scenarios (scenarios with both human drivers and CAVs on the road). Mixed traffic scenarios were until now dealt with conservative methods: traffic lights management or mutual exclusion of vehicles. The algorithm has been demonstrated to have good performance and predictable behaviour on offline and online simulations (continuous traffic simulation). Further works will include testing the algorithm on the full range of CAV market penetration with a better human behaviour engine. It will also be extended to other maneuvers than intersection crossing, such as highway insertion or platooning in mixed traffic environments.

References

[1] Anand Jayant Kulkarni and Kang Tai. "A Probability Collectives Approach for Multi-Agent Distributed and Cooperative Optimization with Tolerance for Agent Failure". In: *Agent-Based Optimization. Studies in Computational Intelligence, vol. 456*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[2] Hironori Suzuki and Yoshitaka Marumo. "A New Approach to Green Light Optimal Speed Advisory (GLOSA) Systems for High-Density Traffic Flowe". In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. 2018, pp. 362–367.

[3] Robert Hult, Gabriel R. Campos, Paolo Falcone, and Henk Wymeersch. "An approximate solution to the optimal coordination problem for autonomous vehicles at intersections". In: *Proceedings of the American Control Conference*. Vol. 2015-July. IEEE, July 2015, pp. 763–768.

[4] Stefanie Manzinger and Matthias Althoff. "Tactical Decision Making for Cooperative Vehicles Using Reachable Sets". In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*. Vol. 2018-Novem. IEEE, Nov. 2018, pp. 444–451.

[5] Charles Philippe, Lounis Adouane, Antonios Tsourdos, Hyo-Sang Shin, and Benoît Thuilot. “Probability Collectives Algorithm applied to Decentralized Intersection Coordination for Connected Autonomous Vehicles”. In: *Intelligent Vehicles Symposium* (2019).

[6] Chi-Ying Liang and Huei Peng. “Optimal adaptive cruise control with guaranteed string stability”. In: *Vehicle system dynamics* 32.4-5 (1999), pp. 313–330.

Chapter 7

General Conclusion and Perspectives

7.1 Conclusion

In Chapter 2 has been proposed a cascade architecture for autonomous vehicle navigation. This architecture seamlessly fills the tasks of trajectory/path tracking as well as dynamic target following and thus can cope with multi-vehicle scenarios. The architecture is divided into a robust low-level yaw stabilization controller that focuses on the vehicle's dynamics and a high-level tracking MPC controller that focuses mainly on the kinematics. This architecture shows an improvement in tracking performances, safety and flexibility compared to usual kinematic controllers for trajectory tracking. It is not intended to have an edge on performances compared to integrated approaches for trajectory tracking but to improve robustness and implementability.

In Chapter 3, the control architecture's capabilities have been extended for explicitly ensuring comfort levels, and management of tracking-related risk for an urban vehicle. This has been achieved by using a linear MPC based approach with appropriate formulation and constraints. It is also robust to both noise on the reference trajectory and model uncertainties due to the low level control design.

The safety monitoring has been achieved through a probabilistic evaluation of the risks linked to the trajectory tracking. The main risk considered is to overshoot the lateral tracking constraints. A performance monitor associated with a probabilistic analysis of the performance has been able to foresee future dangerous situations and prevent them by reducing the speed of the vehicle. Ultimately this provides a guaranteed tracking performance and reduces the speed if necessary. This tracking technique has been tested on the 4DV simulator as well as on real vehicles on the PAVIN platform.

The probabilistic approach has been carried in Chapter 4 by developing a Multi-Vehicle Coordination (MVC) algorithm based on the Probability Collectives (PC) algorithm. The PC algorithm is a multi-agent based optimization technique that was until now used for solving NP-hard problems. Its principle has been heavily modified to make its core compatible with fast optimization in highly constrained environments such as an intersection. The algorithm naturally uses a probabilistic way of representing the data so was a good candidate for further developments in this thesis. The main interests of the proposed modified PC algorithm are its low complexity and its flexibility to any kind of road scenario. That makes it a polyvalent algorithm able to fulfill the objectives stated in Section 1.3. The algorithm exhibits abilities to quickly explore a search space for feasible solutions ($0.8s$ in average for the demonstrated scenarios). The optimality metrics achieved are

satisfying compared to the short time allocated to finding a solution, and it inherently performs better than algorithms based on mutual exclusion from a shared zone.

In Chapter 5 has been carried out an extensive study of the capabilities of this algorithm for intersection crossing. The algorithm exhibits a polynomial complexity which means it is easily scalable (there is no combinatorial complexity explosion). It also shows very little sensitivity to the road layout, meaning that it is generalizable to -at least- various intersection layouts. The convergence speed has been shown to be satisfactory even with a high number of vehicles.

Chapter 5 also describes the development of a framework around the proposed algorithm to make it compatible with real-time continuous traffic management. This framework includes a smooth speed profile generator, a specific intersection layout to ensure that vehicles cannot enter the intersection without a valid collaborative solution, and actual rules for when and which vehicles should participate in a collaborative optimization. This goes towards the objectives of ensuring applicability of the algorithm in the short to medium term as stated in Section 1.3.2. This ability has been validated on Matlab simulations.

The applicability of the proposed algorithm to short term scenarios has been brought even further in Chapter 6. The capabilities of the algorithm have been extended to scenarios with human drivers. It successfully bridges the gap between optimal collaborative planning methods working for 100% of CAV on the road only, and human compatible methods mostly based on traffic lights management. This finalized version of the algorithm has been called the *Unified Probabilistic Multi-Vehicle Coordination* algorithm. In addition to filling the research gap, it brings a natural probabilistic representation of information perfectly suited for representing human behaviours. Additional benefits of this representation include easy to understand risk metrics such as the *Probability of disruption* described in Section 6.2.4.

7.2 Further work

7.2.1 Vehicle Planning and Control

This thesis opens a lot of perspectives for further work, in order to close in on real-life application of all the presented techniques. A Linear Parameter Varying (LPV) model

parametrized by speed could be used for the low-level controller to improve its performance and operational envelope. This would make the controller less conservative while keeping its robustness characteristics. A robustification of the MPC controller could also be carried out in addition to the current characteristics implemented. Such techniques exist in the form of the *Generalized Predictive Control* [1]. The speed monitoring developed in Section 3.2.3 could also be used to feed the probabilistic UP-MVC coordination algorithm with a probability that each vehicle respects its tracking constraints (and at what speed). This could help prune the candidate trajectories in the optimization that are too fast to ensure a given level of safety.

7.2.2 Multi-Vehicle Coordination

The UP-MVC itself has room for more developments. A proper C++ implementation on the 4DV simulator and test vehicles would serve to evaluate more finely the speed of the algorithm and communication requirements. If needed, complexity could be further reduced by grouping vehicles in platoons to cross the intersection and considering a platoon as a single entity. More testing of the UP-MVC could specifically be done in pathological situations such as CAV ratios on the road near 0% (At the very beginning of the introduction of autonomous vehicles on the roads) and specific road layouts such as merging roads. The development of a better human behaviour engine would also be required to explore CAV ratios closer to 0%. For the behavioural prediction of other road users, several works propose such an estimation [2, 3, 4]. However, only [2] proposes an estimation of the uncertainty of the prediction that is required by the proposed UP-MVC algorithm. Current algorithms usually focus on accurate predictions.

Some other pure implementation aspects of the MPC will be interesting to investigate. The properties of the separation function (isotropy, threshold distance). In particular, the separation cost depends at the moment only of the distance between two vehicles. It should also depend on their relative speed as two vehicles approaching each other will appear more dangerous to the human passenger than two vehicles at the same distance with similar directions. The communication requirements imposed by the algorithm could also be evaluated in more depth, even though a very low communication volume requirement has been demonstrated. The algorithm properties could also be studied for a network of intersections, to understand how the properties scale up.

References

[1] Pedro Rodríguez and Didier Dumur. “Generalized predictive control robustification under frequency and time-domain constraints”. In: *IEEE Transactions on Control Systems Technology* 13.4 (July 2005), pp. 577–587.

[2] Michael Goldhammer, Sebastian Köhler, Stefan Zernetsch, Konrad Doll, Bernhard Sick, and Klaus Dietmayer. “Intentions of Vulnerable Road Users—Detection and Forecasting by Means of Machine Learning”. In: *IEEE transactions on intelligent transportation systems* 21.7 (2019), pp. 3035–3045.

[3] Dimia Iberraken, Lounis Adouane, and Dieumet Denis. “Multi-level bayesian decision-making for safe and flexible autonomous navigation in highway environment”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 3984–3990.

[4] Songyang Han, Jie Fu, and Fei Miao. “Exploiting beneficial information sharing among autonomous vehicles”. In: *2019 IEEE 58th Conference on Decision and Control (CDC)*. IEEE. 2019, pp. 2226–2232.