ReliablE in-Vehicle pErception and decisioN-making in complex environmenTal conditionS

Grant Agreement Number: 101069614

D4.3 Behavioural decision-making

Document Identification			
Status	Final	Due Date	31/08/2024
Version	1.0	Submission Date	23/12/2024
Related WP	WP4	Document Reference	D4.3
Related Deliverable(s)	D4.1 and D4.2	Dissemination Level	PU
Lead Participant	CRF	Document Type:	Other
Contributors	All WP4 partners	Lead Authors	Fabio Tango, CRF
		Reviewers	Barys Shyrokau (TUD)
			Kostas Koufos (WMG)

Funded by the European Union

This project has received funding under grant agreement No 101069614. It is funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or European Commission. Neither the European Union nor the granting authority can be held responsible for them.

Document Information

Author(s)		
First Name	**Last Name**	**Partner**
Leonardo	Gonzalez	TECN
Asier	Arizala	TECN
Fabio	Tango	CRF
Oscar	de Groot	TUD
Alberto	Bertipaglia	TUD
Emmmanouil	Gkigkilinis	ICCS
Anastasia	Bolovinou	ICCS
Bill	Roungas	ICCS
Thomas	Griebel	UULM
Anthony	Ohazulike	HIT

Document History			
Version	**Date**	**Modified by**	**Modification reason**
0.1	15/07/2024	CRF	Empty document with section structure
0.2	03/10/2023	TUD & UULM	Received contributions from TUD and UULM
0.3	20/11/2024	TECN & ICCS	Received contributions from TECN and ICCS
0.4	04/12/2024	CRF	Merged contributions of all partners in a single document. Version ready for review.
0.5	15/12/2024	WMG	Received review from WMG
0.6	17/12/2024	TUD	Received review from TUD
0.7	20/12/2024	CRF	Merged reviews. Document ready for submission
1.0	23/12/2024	ICCS	Last check and submission

Quality Control		
Role	**Who (Partner short name)**	**Approval Date**
Deliverable leader	Fabio Tango (CRF)	20/12/2024
Quality manager	Panagiotis Lytrivis (ICCS)	23/12/2024
Project Coordinator	Angelos Amditis (ICCS)	23/12/2024

Executive Summary

This deliverable highlights the behavioural decision-making strategies, including the strategies for identifying the optimal manoeuvre in a given situation. This is the outcome of EVENTS task T4.2 on behavioural decision-making for all experiments involved in this task. Note that such activities are still work in progress, meaning that their full evaluation will be detailed in the deliverables of WP6.

Not every experiment in the project has the goal of handling the decision-making of the vehicle. Therefore, only the ones for which a full navigation system will be designed are mentioned here. Considering the current stage of development related to the decision-making inside the project, this document contains at least the system architectures and the algorithmic designs with the first results depending on each experiment.

Table of Contents

List of Tables

List of Figures

Abbreviations & Acronyms

Abbreviation / acronym	Description
ACC	Adaptive Cruise Control
AD(F)	Autonomous Driving (Function)
AI	Artificial Intelligence
AL	Alert Limit
AV	Automated Vehicle
BP	Behavioural Planning
CA	Consortium Agreement
CAM	Cooperative Awareness Message
CAV	Connected Automated Vehicle
CPM	Collective Perception Message
DDT	Dynamic Driving Task
DENM	Decentralized Environmental Notification Message
DM	Decision-Making
EC	European Commission
EXPs	Experiments
FIS	Fuzzy Inference System
FoV	Follow opposing vehicle
FTP	Fail-degraded Trajectory Planning
GA	Grant Agreement
IR	Integrity Risk
ISO	International Organization for Standardization
I/O	Input(s) / Output(s)
LiDAR	Light Detection and Ranging
MDP	Markov Decision Process
MPC	Model Predictive Control
MRM	Minimum Risk Manoeuvre
MOP	Moving Object Prediction

Abbreviation / acronym	Description
MOT	Multi-Object Tracking
ODD	Operational Design Domain
PE	Position Error
PL	Protection Level
PP	Perception Platform
RADAR	RAdio Detecting And Ranging
REQs	Requirements
RL	Reinforcement Learning
SAE	Society of Automotive Engineers
SMD	Safety-mode Decision
SoTA	State of the art
SPaT message	Signal Phase and Timing message
SPECs	Specifications
TTC	Time to Collision
TOR	Take Over Request
TP	Trajectory Planner
TSs	Target Scenarios
UCs	Use Cases
VRU	Vulnerable Road User
WP	Work Package

1. Introduction

This document follows the structure used for all EVENTS project deliverables. It begins with an introduction outlining the aims of the project and the scope of this document, followed by the main body, which, in this deliverable, focuses on behavioural decision-making strategies.

1.1 Project aim

Driving is a challenging task. In our everyday life as drivers, we are facing unexpected situations we need to handle in a safe and efficient way. The same is valid for Connected and Automated Vehicles (CAVs), which also need to handle these situations, to a certain extent, depending on their automation level. The higher the automation level is, the higher the expectations for the system to cope with these situations are.

In the context of this project, these unexpected situations where the normal operation of the CAV is close to be disrupted (e.g., ODD limit is reached due to traffic changes, harsh weather/light conditions, imperfect data, sensor/communication failures, etc.), are called “events”. EVENTS is also the acronym of this project.

Nowadays, CAVs are facing several challenges (e.g., perception in complex urban environments, Vulnerable Road Users (VRUs) detection, perception in adverse weather and low visibility conditions) that should be overcome to be able to drive through these events in a safe and reliable way.

Within this project scope, and to cover a wide area of scenarios, these kinds of events are clustered under three main use cases: a) Interaction with VRUs, b) Non-Standard and Unstructured Road Conditions and c) Low Visibility and Adverse Weather Conditions.

Our vision in the EVENTS project is to create a robust and self-resilient perception and decision-making system for Automated Vehicles (AVs) to effectively manage different kinds of “events” on the horizon. These events result in reaching the AV’s Operational Design Domain (ODD) limitations due to the dynamically changing road environment (VRUs, obstacles) and/or due to imperfect data (e.g., sensor and communication failures). The AV should continue to operate safely under all circumstances. When it cannot handle a situation, an improved minimum risk manoeuvre should be put in place to ensure safety.

1.2 Deliverable scope and content of the Document

In the EVENTS project, WP4 main goals are to design, implement and test in a lab environment (or prototype vehicle), the on-board decision-making and control algorithms of EVENTS use cases, considering complex traffic and environmental conditions, especially around critical zones, where potential contradictions to existing traffic rules can emerge, or the environment is unstructured. This Deliverable (D4.3) reports the complete design and algorithms for behavioural decision-making, as developed in Task 4.2, as well as the initial results of testing. The complete integration of the WP4 modules will be then described in the deliverable D5.2 of WP5.

In detail, this deliverable highlights the behavioural decision-making strategies, including strategies for identifying the optimal manoeuvre in a given situation, as implemented in T4.2. Moreover, Behavioural Planning (BP) supports motion planning (MP), providing the necessary information to compute the trajectories (MP is described in the deliverable D4.2). BP considers both collision risk assessment and manoeuvre selection.

The information does not necessarily flow from the behavioural planner to the motion planner. For instance, some MP methods generate motion candidates, which are evaluated by the behavioural planner. The information flow is depicted in Figure 1 (as derived by D4.1 [1]), where some elements are highlighted, such as occupancy grids and the values of time to a collision, that are sent to the MP model or the trajectory candidates that can be sent back to the BP.

The output of the perception models developed in WP3 of the project (e.g., obstacle information) serve as input for the BP module. Additionally, further details about the surroundings of the vehicle can be obtained from the communication modules, e.g., through vehicle-to-everything (V2X)-enabled collaborative (or collective) perception.

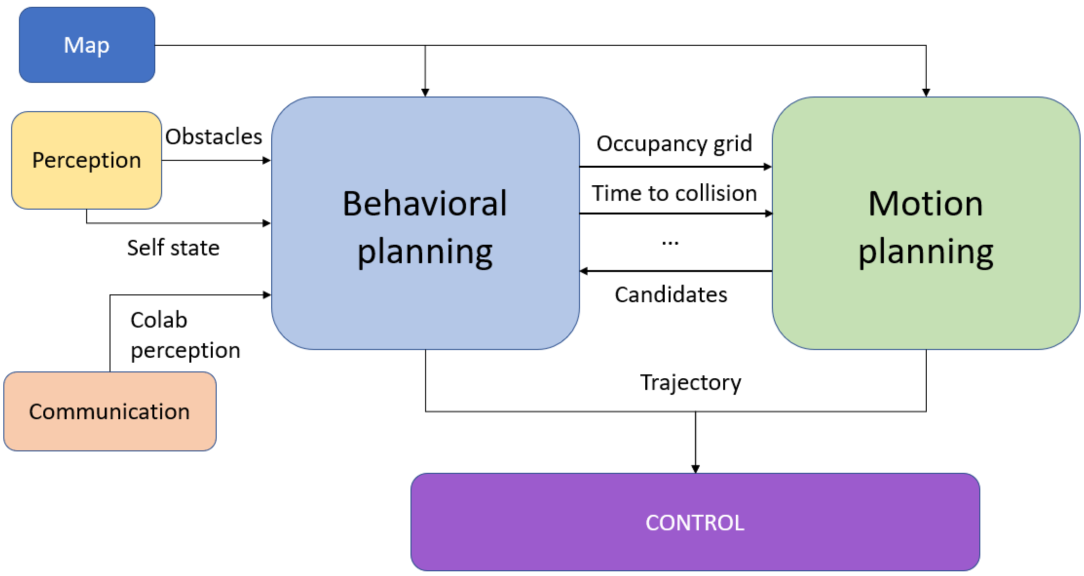

Figure 1: Behavioural Planning and Motion planning generic information flow

The document is structured as follows. After this introduction, Chapter 2 describes the research and progress made in the behavioural planning for each of the relevant experiments, as well as future works to be included in following deliverables. Finally, Chapter 3 presents the conclusions of the progress mentioned in the previous chapters, including a summary of the document and possible lessons learnt.

2. Behavioural decision-making

In the context of WP4, T4.2 focuses on “behavioural decision-making”. Using the inputs provided by WP3 (e.g., tracked objects and their associated forecasted short-term trajectory, lane marking detection, self-assessment of the perception, etc.) the goal is to ***generate behavioural decisions***, that is, to determine the “**best action**” to perform (e.g., brake, lane-keeping / car-following, lane-change, return to the left lane, etc.). Figure 2 illustrates the various connections among the tasks within WP4, pointing out the inputs and outputs specific to T4.2:

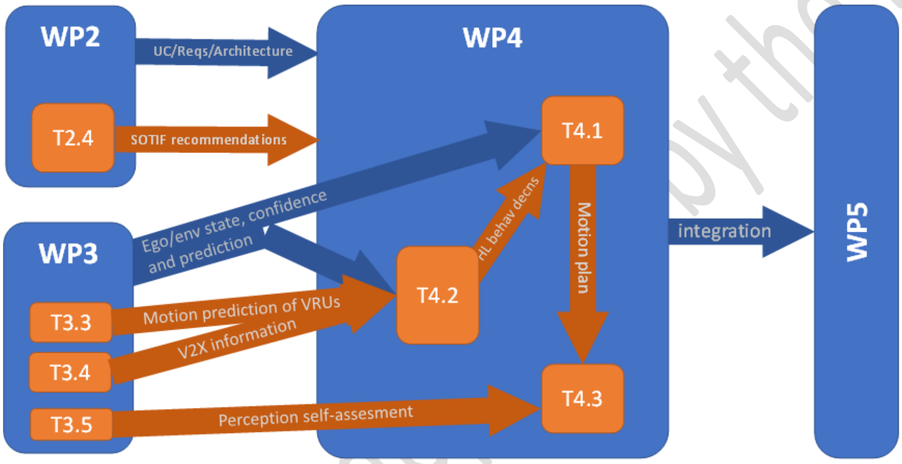

Figure 2: Graphical sketch of the WP4 interactions

The outputs of the behavioural decision-making module can be used by a human driver or an automated system to enhance safety significantly, as well as to increase comfort and efficiency and create new solutions for individual transport in cities.

As mentioned in the deliverable D4.1 [1], the algorithms used in T4.2 leverage state-of-the-art (SOTA) machine learning (ML) and probabilistic methods, such as [2], [3]. While developing such algorithms, it is important to consider various technical challenges. Firstly, the perception of an AV is imperfect due to noise, range limitations of sensors, as well as occlusions/blocks in the environment. Secondly, in order to generate safe trajectories for the ego vehicle, the motion of the other traffic participants (TPs) needs to be predicted, by taking into consideration the uncertain information of their current state, including hidden variables (such as unknown goals and destinations). Thirdly, since the motion of the ego vehicle must be collision-free, the probabilistic optimization framework has to meet its kinematic and dynamic constraints and follow the traffic rules [4]-[5].

In the following chapters, the developed algorithms will be presented experiment by experiment (EXP, in short). The only exception is EXP6, “Small object detection at a far range in adverse weather conditions", led by the APTIV partner, which focuses only on the perception aspects.

2.1 EXP1

Its title is "Interaction with VRUs in complex urban environment", and it is under the responsibility of TUD. The objective of Experiment 1 (EXP1) is to achieve safe, comfortable, and time-efficient automated driving in a complex urban environment while engaging with VRUs such as pedestrians and cyclists.

To handle interactions with VRUs, the behavioral decision-making component is tightly coupled with the local motion planners described in D4.2. This chapter details the architecture of the decision-making components.

2.1.1 Architecture

When interacting with VRUs, the behavioural decisions that the planner can take depend on the exact position and predicted intentions of the VRUs. Decisions, therefore, do not only pertain to the discrete level (e.g., overtake or brake) but must explicitly consider what behaviours are dynamically feasible. For this purpose, the motion planner and behavioural decision-making components are closely coupled (see Figure 3). It consists of a Guidance Planner that computes high-level guidance trajectories with distinct behaviours and the decision-making, after locally optimizing trajectories in each behaviour, decides what is the best course of action, given the quality of these trajectories.

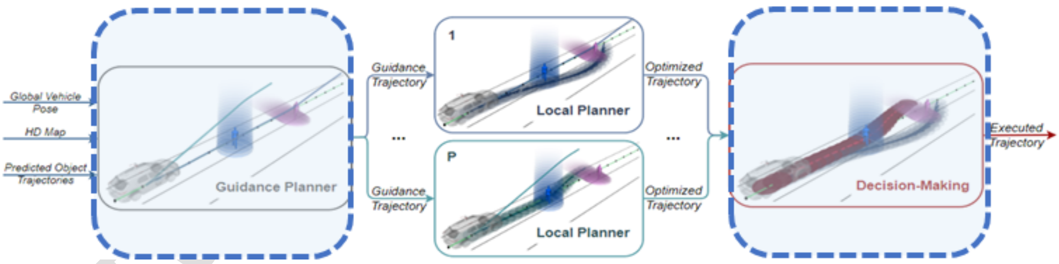

Figure 3: Behavioural decision-making component architecture of EXP1

Guidance Planner. The guidance planner is a modified sampling-based planning algorithm based on Probabilistic Roadmaps (PRM). In particular, it uses Visibility PRM [6], which results in a sparse graph through the environment. To incorporate dynamic objects, the environment is considered in 3D, consisting of 2D position (x, y) and time (t), allowing the guidance planner to incorporate their expected motion. The guidance planner differentiates between different behaviours by analysing the homotopy class of the trajectory in the underlying collision-free space. The homotopy class of a trajectory captures a set of trajectories that can be continuously transformed into each other, which holds for trajectories that pass obstacles on the same side but does not hold for trajectories that pass obstacles on opposite sides. Filtering on the homotropy class allows the guidance planner to filter out equivalent behaviours, ultimately leading to a set of distinct collision-free behaviours. Figure 4 illustrates how the homotopy classes are compared using the H-signature [7] by computing whether a loop of trajectories (in blue/red) encloses obstacle motion predictions in 2D space with time (3D).

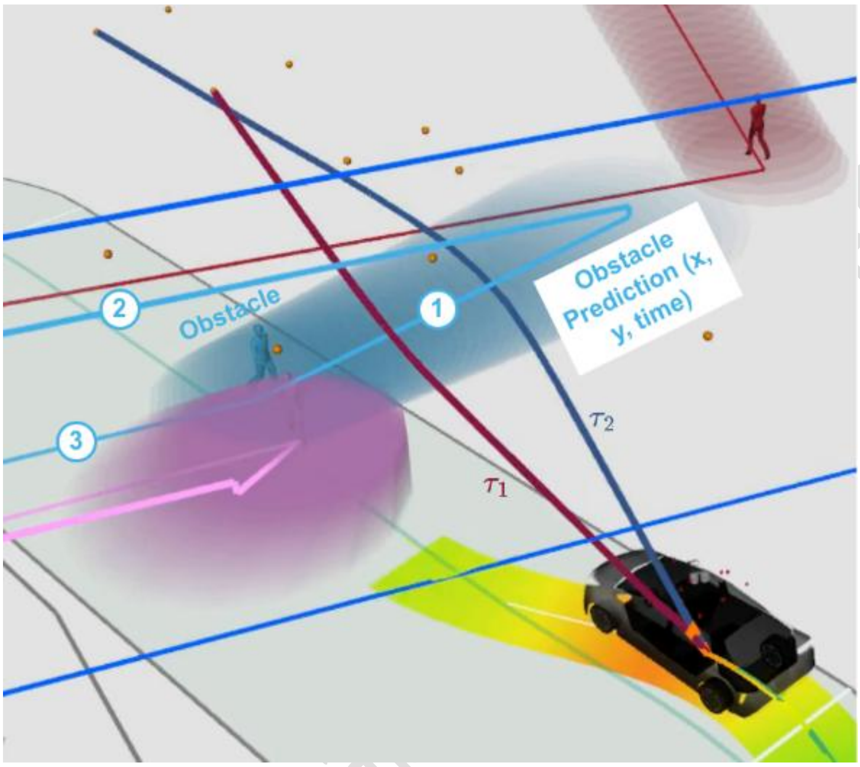

Figure 4: Homotopy classes are compared using the H-signature

These motion predictions consist of (1) the motion predictions for 0≤t≤T, where T is the final time, (2) a line at t=T+ϵ, going outside of the workspace, with ϵ is a small value, resulting in different trajectories τ_1 and τ_2. There, it connects down to (3) and moves back to the obstacle at t= -ϵ. In this example, the H-signature is different for the blue and red trajectories as they enclose the light blue motion prediction of the blue obstacle.

Decision-Making. It compares each of the refined trajectories. Since the local planners described in D4.2, all have the same objective function, the quality of each trajectory can be compared by their optimal costs. It is likely that each local planner computes a locally optimal solution to the underlying optimization problem, and by comparing the optimal costs, the decision-making module can pick the best local optimum, improving on the standard planning approach that identifies a single arbitrary local optimum. In addition, given that multiple trajectories are optimized jointly, it is more likely that at least one trajectory is computed, improving the robustness of the optimization-based planner.

While selecting the lowest-cost solution leads to the theoretically best solution, it is possible that this solution rapidly switches between iterations when the environment is difficult to estimate accurately. A consistent decision-making scheme is therefore also considered. This scheme is facilitated by the guidance planner that, in each iteration, reidentifies trajectories to determine if trajectories in previously computed homotopy classes have reappeared. By assigning them an identifier, it is possible to determine which trajectory the vehicle was previously executing and that trajectory preference by lowering its cost by a factor. This makes it more likely that the previously followed trajectory is executed again.

Local Motion Planning consists of multiple MPC planners that each refine one of the guidance trajectories, incorporating dynamic constraints and a more detailed objective. Its detailed description is in D4.2.

2.1.2 Algorithmic Approach

Guidance Planner. The implementation of the guidance planner differs from existing Visibility PRM implementations in several key aspects:

1. It considers not only one but multiple goal positions, allowing it to find a trajectory even if its desired position is not reachable (e.g., because another vehicle is driving in front of the ego-vehicle).
2. Homotopy classes of trajectories are compared to distinguish between equivalent and distinct trajectories.
3. The graph in each iteration is propagated to the next iteration by dropping the time coordinate of each node by the planning time.
4. Trajectories are propagated to following iterations by comparing homotopy classes of newly computed trajectories against the trajectories from the previous iteration.

These features ultimately allow the guidance planner to maintain a consistent set of trajectories with different passing behaviours. The guidance planner must be initialized with the vehicle position and a set of goal positions. For urban driving, given the reference path $[0, S] \to \mathbb{R}^2$ and adaptive reference velocity $[0, S] \to \mathbb{R}$, five path positions are sampled along the reference path starting from the current path position of the vehicle and considering the velocity along the path. For each of these longitudinal positions, three lateral goals are added. Similarly, to populate the graph, samples are drawn from a uniform distribution over support of $\Delta = [s_0, s_N]$, that is, the path parameter from the current to the expected final path position while additionally sampling between $[b_l(s_k), b_r(s_k)]$ that are the road boundaries at s_k, and between $[0, N]$ in time axis. The resulting sampling and graph are shown in Figure 5, where goals (circled, orange) and samples (blue) of the guidance planner follow the road centerline and consider the road width to determine feasible driving behaviours.

If there are more trajectories than the maximum specified number of paths, then the trajectories are singled out and move the furthest over time.

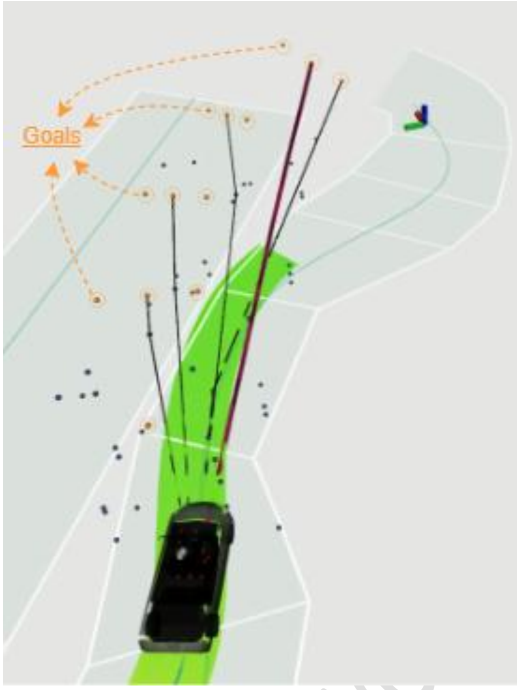

Figure 5: The goals and samples of the guidance planner

In the vicinity of VRUs, the guidance planner generally computes more than one passing behaviour, as shown in Figure 6, where in the left, it identifies that the blue (passes in front) and red (passes behind) trajectories are different. In this case, it finds that the blue trajectory requires high acceleration and decides to pass behind the pedestrian, as indicated by the yellow (slow speed) vehicle plan. In the right part of Figure 6, instead of stopping (blue trajectory), it decides to speed up slightly to pass in front indicated by the green (normal speed) trajectory. These two cases illustrate the advantages of computing multiple local optima. A more standard pipeline would decide, based on limited information, what behaviour the vehicle should follow, and the local planner would refine this behaviour. This could easily lead to the worst of the two options being executed by the vehicle and would require significant tuning effort to make the best decision, while in the proposed method, the decision-making is left to the objective function of the MPC.

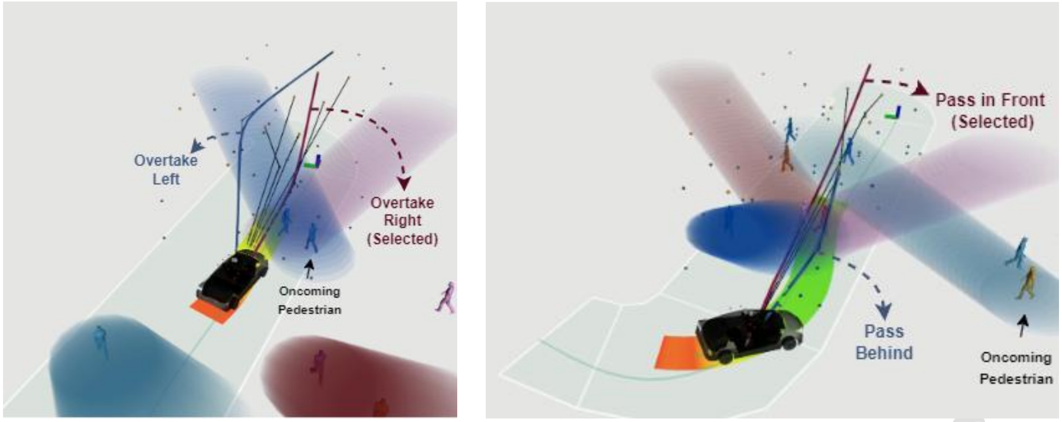

Figure 6: The guidance planner computing more than one trajectories

Decision-Making. Two decision-making strategies are considered. The first is denoted the *minimal cost decision* and compares the optimal cost of all locally optimized trajectories to select the lowest cost option. That is, with J_i^* denoting the optimal cost of trajectory i, the trajectory is selected with:

$$\tau_i^*, i = argmin_i \, J_i^*, \tag{1}$$

This strategy results in the theoretically best solution (with the lowest cost). It is likely that this solution is closer to the global optimal solution than a single locally optimized trajectory would be. Overall, this improves planner performance. The downside of using the minimal cost decision is that over successive iterations of the planner and when two plans are of comparable quality, the optimal trajectory may switch rapidly. This induces oscillations that are not desirable. Therefore, an alternative decision-making strategy is proposed, denoted as the *consistent decision*. To accommodate this decision, we note that the homotopy class of trajectories in the previous planner iteration may be compared to those of the new trajectories $\tau_i^*, \forall i$. When a new trajectory exists in the homotopy class of the trajectory that the vehicle is already following, then choosing that trajectory should be preferred over other options unless the other options are much better. To quantify this in practice, the consistent decision is defined as:

$$\tau_i^*, i = argmin_i \, w_i J_i^*, \tag{2}$$

where $w_i = c_i$ if this trajectory was previously selected, $0 \leq c_i \leq 1$, and $w_i = 1$, otherwise. That is, the previously selected trajectory is discounted with a factor $1 - c_i$. For $c_i = 0$, the previous trajectory is always selected if it exists. For $c_i = 1$, the lowest cost decision is obtained. Therefore, the value of c_i can be tuned to encode the desired consistency of the planner. In preliminary experiments, it was found that $c_i = 0.25$ to be a good starting point. With this value, oscillations are eliminated while the planner is still able to switch the passing direction of obstacles when the scenario requires it.

The guidance planner and decision-making components are implemented in ROS2/C++ and Autoware. Its parameters are listed in Table 1.

Table 1: Planner parameters, preliminary values and their description

Parameter Name	Parameter Value (preliminary)	Description
$n_{samples}$	50	Number of samples in each iteration
P	2	Number of distinct trajectories
c_i	0.25	Factor with which the cost of the previously followed trajectory is discounted.
N	35	Planning horizon
Δt, s	0.2	Time between discrete time steps of the prediction
T, s	7.0	Planning time horizon
t_{loop}, s	0.1	Planning time
$n_{obstacles}$	4	Number of obstacles
G	5 x 3	Goal grid (longitudinal x lateral)

Because the samples of the guidance planner are reused in following iterations, only a few samples are necessary in each iteration, and the guidance planner is typically executed under 5ms.

2.1.3 Future work

Future work includes quantitative evaluation of behavioural decision-making in randomized simulation scenarios with multiple VRUs via simulations and real-world tests.

2.2 EXP2

The EXP2 "Re-establish platoon formation after split due to roundabout", is being led by Tecnalia with support from ICCS as a partner. In this scenario, a platoon of AVs approaches a roundabout in an urban environment with heavy traffic. As the lead vehicle navigates through the roundabout, it faces the challenge of maintaining cohesion with the following vehicles in the platoon. However, due to the density and flow of external traffic within the roundabout, gaps form between the vehicles, causing the platoon to split temporarily. This separation disrupts the intended formation and introduces the risk of losing coordination among the platoon members, which could compromise the efficiency and safety of the crossing.

To address this, three decision-making methods based on different technologies are proposed in this work. First, a fuzzy logic-based decision-making method is proposed for the following vehicles to decide whether they should follow the preceding vehicle in the platoon or an outside vehicle inside the roundabout. Second, an MPC-based trajectory generator is proposed with obstacle consideration for speed planning.

Finally, an AI-based solution is proposed for the generation of the speed profiles trained using Reinforcement Learning.

This case is specific enough not to have been touched on the SOTA previously. Therefore, three different solutions are going to be tested to get the desired result.

2.2.1 Fuzzy logic based selection

Fuzzy logic-based algorithms aim to mimic human decision-making processes, making them well-suited for handling complex driving scenarios like roundabouts. When a human driver approaches a roundabout, they must decide whether to continue following the vehicle directly ahead or wait for other vehicles already in the roundabout. This solution adopts a similar approach, using fuzzy logic to help the AV make safe, human-like decisions about when to follow or yield to other vehicles in dynamic environments.

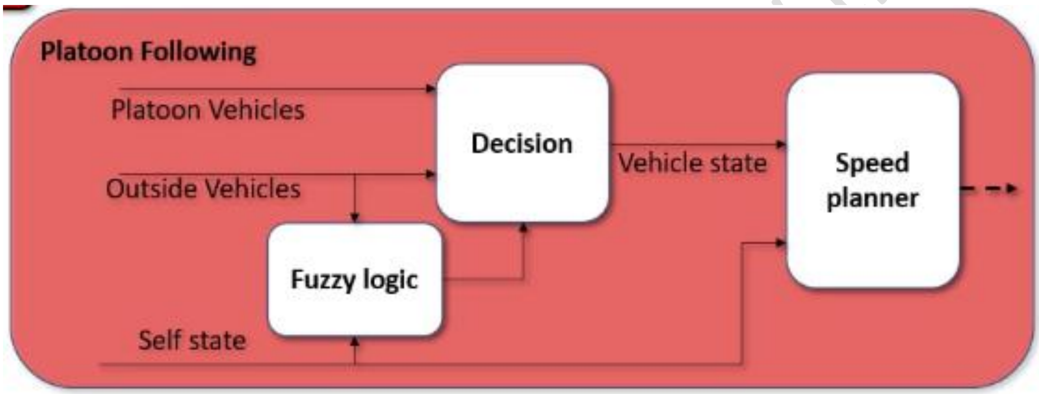

Figure 7: Fuzzy logic-based decision architecture

In Figure 7 a scheme of the decision-making architecture is presented, illustrating how the AV uses inputs from platoon vehicles, outside vehicles, and its own state to make safe and efficient speed adjustments. The architecture incorporates fuzzy logic to handle uncertainties in vehicle detection and positioning, allowing the decision module to determine which vehicle to follow. This decision is then passed to the speed planner, which adjusts the ego vehicle's speed profile to maintain safe distances and avoid collisions within a roundabout scenario.

The selection of the variables considered by the Decision-Making algorithm has been carried out by considering human reasoning when it comes to making this same decision. Vehicles outside the platoon will be considered external vehicles from this point onwards. Hence, distance to an external vehicle, self-speed and the external vehicle speed are used as input variables. The following fuzzy sets have been defined for the aforementioned variables:

- **Distance** to the external vehicle, calculated as the Euclidean distance to an external vehicle: Very close (VC), Close (C), Far away (FA), Very far away (VFA).

– **Speed** of the controlled vehicle/ Speed of an external vehicle: Slow(S), Medium (M), Fast (F).

Finally, the output of the fuzzy logic algorithm is defined by an action variable. It can get two possible values that have been separated in trapezoidal fuzzy sets: "Follow opposing vehicle" (FOV) and "Stay in platoon" (SIP).

Distance	VC		
Oposed / Controlled	S	M	F
S	FOV	FOV	FOV
M	FOV	FOV	FOV
F	FOV	FOV	FOV

Distance	C		
Oposed / Controlled	S	M	F
S	FOV	FOV	SIP
M	FOV	FOV	SIP
F	SIP	SIP	SIP

Distance	FA		
Oposed / Controlled	S	M	F
S	SIP	SIP	SIP
M	SIP	SIP	SIP
F	SIP	SIP	SIP

Distance	VFA		
Oposed / Controlled	S	M	F
S	SIP	SIP	SIP
M	SIP	SIP	SIP
F	SIP	SIP	SIP

Table 2: Rules of the fuzzy logic

Based on these inputs, the algorithm outputs either **FOV** or **SIP**, with each output corresponding to a floating-point value between 0 and 1. The value indicates the confidence or strength of the decision, with values closer to 1 representing a higher confidence. When SIP is suggested, the algorithm advises the ego vehicle to continue following the vehicle ahead, while FOV suggests that the ego vehicle should follow the external vehicle, prioritizing safety in response to the detected distances and speeds. The fuzzy rules allow the vehicle to make nuanced and adaptive decisions based on the specific conditions in the roundabout. The set of rules has been selected so it is consistent with the decisions a human driver would take in this exact scenario.

Vehicle-to-vehicle (V2V) communication enables the use of Cooperative Adaptive Cruise Control (CACC) within the platoon, as it allows vehicles to share detailed information about their speed, position, and intentions. However, information obtained from external vehicles through perception sensors is limited and lacks the depth needed to implement the same cooperative approach. Consequently, for following non-platoon vehicles, the system relies on standard Adaptive Cruise Control (ACC), which adjusts speed based solely on the perceived distance and relative speed of the external vehicle.

2.2.2 MPC trajectory generator

Model Predictive Control (MPC) is arguably the most popular trajectory generation method when it comes to AVs. The possibility of optimizing a path based on specific vehicle dynamics with physical constraints in a single method is appealing for smooth collision-free trajectory generation. With the lack of a higher-level reference generator or behavioural planner, this method can be used for continuous decision-making in an AV architecture. Moreover, it can be considered both, a decision-making and path planner module if well designed.

In this alternative method, an MPC approach is employed to generate a safe and efficient trajectory for the ego vehicle as it navigates the roundabout. Unlike traditional ACC, which relies primarily on maintaining distance from a preceding vehicle, MPC allows for more sophisticated trajectory planning by predicting and optimizing over a future time horizon. The MPC generates a speed profile for the ego vehicle that adapts to the dynamic conditions within the roundabout, ensuring smooth entry, navigation, and exit while maintaining safe distances from both platoon and non-platoon vehicles (Figure 8).

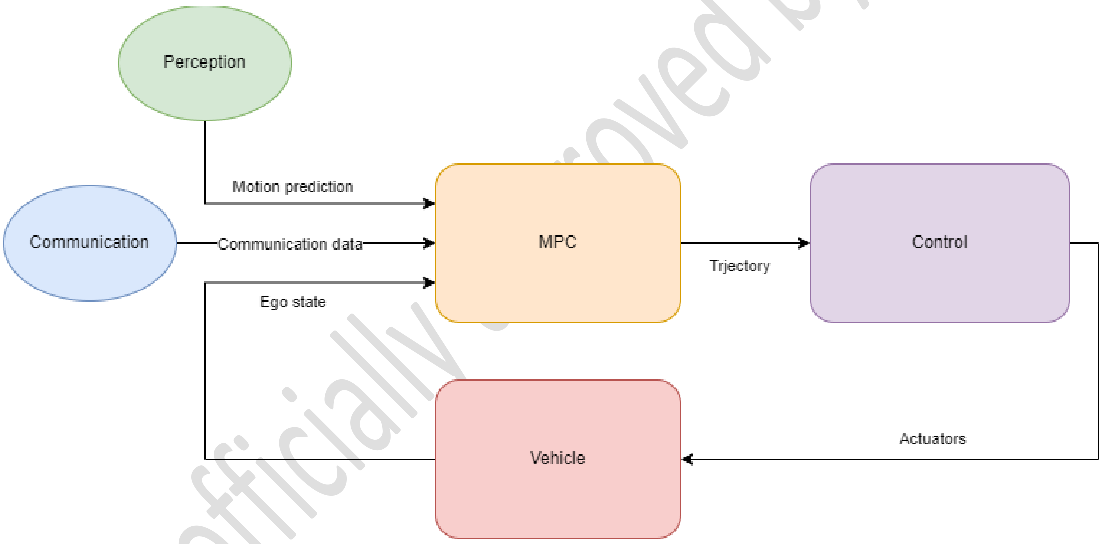

Figure 8: EXP2 architecture

A key advantage of this MPC-based approach is its ability to incorporate the predicted trajectories of surrounding obstacles. By forecasting the movements of nearby vehicles, the MPC can adjust the ego vehicle's speed and path to proactively avoid potential collisions. This predictive aspect is crucial in a roundabout scenario, where vehicles may merge, exit, or change lanes unpredictably.

As seen in several works [7]-[8], the model used for the MPC planning is a kinematic model of a two-axle vehicle with a displaced control point. Since one of the objectives of the MPC is to create smooth trajectories and speed profiles the following states and inputs are used:

$$\bar{x} = \{x, y, \theta, v, \delta\};\ \bar{u} = \{a, \dot{\delta}\} \tag{1}$$

Where, x, y and θ are the positions and orientation in the frame of the initial position, v is the longitudinal speed and δ is the road wheel angle represented in the center of the front axle. The inputs of the model are the longitudinal acceleration (a) and the steering rate $(\dot{\delta})$.

The first derivate of the state can be represented by the following equations:

$$\dot{x} = \begin{cases} v\cos(\theta+\beta) \\ v\sin(\theta+\beta) \\ v\cos(\beta)\dfrac{\tan(\delta)}{L} \\ a \\ \dot{\delta} \end{cases} \tag{2}$$

Where, L is the wheelbase of the vehicle and β the sideslip angle is calculated as:

$$\beta = \operatorname{atan}\left(\frac{l_r\tan(\delta)}{L}\right) \tag{3}$$

Where, l_r is the distance from the rear axle to the center of the mass of the vehicle.

The optimization problem can be defined in the following way:

$$\begin{aligned} \min \quad J(\bar{x},\bar{u}) = & \int_{i=0}^{H} w_{dist} d_{ref}^2 + w_{orient}\theta_{dif} \\ & + w_{steer}\delta^2 + w_{acc}a^2 + w_{strate}\dot{\delta}^2 \\ & + \sum_{j=0}^{N} \frac{w_{obst}}{d_{obst}} \end{aligned} \tag{4}$$

$$\text{s.t.} \quad \begin{aligned} \delta_{\min} & < \delta < \delta_{\max} \\ 0 & < v < v_{\max} \\ a_{\min} & < a < a_{\max} \end{aligned} \tag{5}$$

The cost function tries to minimize the distance and orientation to the reference trajectory (d_{ref}, θ_{dif}), generated by the past positions and prediction of the platoon leader), the road wheel angle change (δ) and the input variation $(a, \dot{\delta})$, which should lead to smoother and more comfortable paths.

In MPC trajectory generation, obstacles are typically represented using Artificial Potential Fields (APF) if the objective is to avoid them [11],[10]. However, in this particular case, the ego vehicle should keep driving behind the obstacles. Therefore, obstacle consideration is added to the cost function through the last element, rewarding the vehicle for keeping a safe distance from them.

Specific parameter values, such as prediction horizon, weights or constraint values, will be specified in a future deliverable for the best result obtained with this method.

2.2.3 Reinforcement learning for speed planner

Automated driving is an extremely complex task encompassing a wide variety of challenges, from motion planning to individual decision-making. These challenges are often addressed from a traditional technical perspective, e.g., rule-based methods. However, Reinforcement Learning (RL), powered by deep neural networks, has been applied to efficiently solve these challenges over the past decades. In what follows, a thorough analysis of the SOTA RL techniques applied to adverse traffic scenarios (e.g., intersections, roundabouts) is performed, along with a review of some works related to platooning orchestration.

RL has been used to approach various scenarios within the scope of automated driving, with path planning and decision-making being two of the main focuses within academia. While the former determines the path the vehicle should follow in the next few seconds, the latter allows for decisions on whether to accelerate, decelerate, or stop to avoid collisions.

RL can be defined as a machine learning technique that defines how decisions (actions, A) made by agents guide them toward achieving a specific goal. To accomplish this, agents are provided with and trained on observations, which encode the current state of the simulated scenario, and rewards (R), which measure the appropriateness of taking action given the current environment configuration (state, S), as shown in Figure 9:

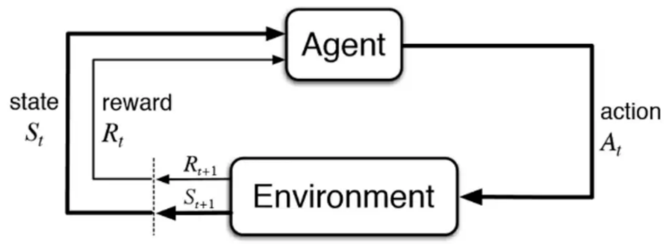

Figure 9: Traditional RL agent/environment interaction

Under the scope of RL, there are plenty of algorithms and configurations suited for solving a wide variety of automated driving scenarios, which will not be covered in this report; however, readers interested in this subject are encouraged to read this survey [9]. In [9], the authors provide a review of Deep RL techniques addressing the challenge of automated driving at unsignalized intersections, such as roundabouts and 4-way intersections. Key issues include predicting the intentions of other drivers and planning movements in conditions of uncertainty, both of which are crucial for safe navigation. To address these challenges, cooperative, heuristic, and game-theory-based methods are explored.

Automated driving at intersections (signalized or not) provides an excellent scenario for RL due to their inherent uncertainty. Specifically, in the case of roundabouts, the main concern of the experimentation proposed in this document, the community has focused its efforts on decision-making and controlling agents when approaching a roundabout. In [5], this problem is approached by proposing a Q-Learning algorithm to train an AV agent. Using the CARLA simulation environment, the algorithm is trained and tested in traffic and non-traffic scenarios. Experimental results show that the Q-Learning-trained agent successfully learns smooth and efficient roundabout manoeuvres. The action space includes vehicle control commands: steering angle, acceleration, braking, and handbrake activation, while the observation space comprises vehicular data such as GPS position, velocity components, distance to the next GPS point, and binary indicators for lanes to the left or right of the vehicle.

Usually, mimicking human decisions can help RL systems learn the right policy under extreme circumstances. The work conducted in [12] introduces a decision-making system based on imitation learning to help AVs safely and efficiently merge into roundabouts. The system captures observations and employs deep policy networks, trained with human expert data (GO and WAIT signals), which are used as reward scalation for the agent when it takes the same action as the human expert. The method is finally evaluated against traditional supervised learning models (Support Vector Machines and k-Nearest Neighbours) and other deep learning methods, showing superior decision accuracy. However, these algorithms lack adaptability when the action space shifts from a discrete action set to a continuous action space, such as velocity or yaw angles, which are used in [13] to make AVs learn how to turn left securely at unmarked intersections. The training is conducted using Soft Actor-Critic (SAC), a model-free off-policy RL algorithm, which generates reference speed and yaw angle signals focused on safety and collision avoidance, while the MPC system optimizes these signals within vehicle dynamics constraints. Other studies [14] expand the action space and the actor-critic-based algorithm to allow agents to decide whether they should change lanes.

However, complexities may not only arise from the curvature of the roundabout or the absence of signalling but also from traffic flows, different vehicle structures (e.g., tractor-trailer vehicle [15]) or vehicle compositions (e.g., platooning). Maintaining automated platooning requires precise, reliable sensing to track vehicle positions and respond to environmental factors like rain or fog. Real-time decision-making is essential, as control algorithms must rapidly process information and adjust spacing to ensure stability. Balancing driving efficiency with safety is crucial, and the system must adapt to varied traffic and road conditions.

A platoon can be guided by an RL system, as done in [16], where the position of each vehicle and traffic density is analysed to reduce traffic congestion and CO2 emissions through the routes derived from the proposed methodology. A similar approach is presented in [13], where cooperative driving strategies are used to generate offline collision-free trajectories in a roundabout, while a real-time cooperative decision-making and motion planning method using Adaptive Monte Carlo Tree Search (AMCTS) is employed for movement coordination.

More elaborate solutions can be found in [17] and [18]. Starting with [17], the RL algorithm used in this work (i.e., comPPO) enables each agent to adjust its acceleration to reduce fuel consumption based on the shared platoon state, which includes a shared set of variables such as traffic state, speed differences, vehicle speed, gap between vehicles, and the ordinal number of the vehicle. Next, the work in [18] introduces the FH-DDPG-SS (Finite-Horizon DDPG with Sweeping through reduced state space using Stationary policy approximation) algorithm, which enhances the efficiency of reinforcement learning for platoon control. One key innovation is the transfer of network weights over time, where weights from networks trained in later time steps are transferred to earlier ones. This technique improves sampling efficiency by leveraging the knowledge gained from more advanced stages of training to accelerate the learning process at the beginning, thus optimizing the overall training and performance of the system.

In summary, the capabilities and potential of RL in the field of automated driving are undeniable. Its strengths enable it to address problems ranging from simpler tasks using small neural networks to complex scenarios requiring multi-agent cooperation. It is important to note that, while many RL studies focus on the application of RL methods to solve issues related to automated driving in roundabouts or other intersections, a few research has been conducted on how to restore the formation of a vehicle platoon when only a subset of the vehicles has successfully entered the roundabout. The next section will detail the RL system designed to tackle this challenge.

2.2.4 Reinforcement Learning Solution

In the proposed scenario, as illustrated in Figure 10, one of the vehicles in the platoon faces the challenge of entering a roundabout at the optimal moment. The complexity arises from the presence of a third vehicle, referred to as Vehicle X, which obstructs Vehicle 3's entry into the roundabout, creating a dynamic and constrained decision-making problem.

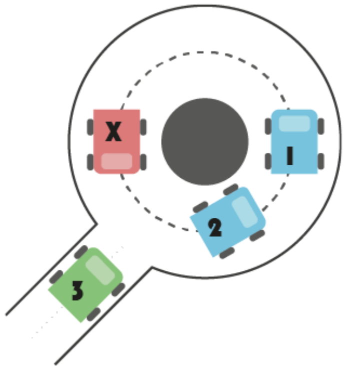

Figure 10: Proposed scenario in EXP2

To address this, the observation space must first be clearly defined. Since the environment is considered fully observable, the observation space can be defined as:

$$\{(p_i, v_i, o_i, \varepsilon, \delta_{i,j}) \mid i,j \in N, i \neq j\} \quad (6)$$

where N is the number of vehicles (i.e. 4 in this experiment) and $p_i, v_i, o_i, \varepsilon$ *and* $\delta_{i,j}$ represent the position of the whole set of vehicles, their velocities, identification (1,2,3 or X), controlled vehicle's lateral error and the distances between any pairs of cars, respectively.

The primary objective of this experiment is to regulate the speed of the green vehicle (the action), which can be modelled as:

$$a \in \mathbb{R}(0, 1), \quad (7)$$

allowing it to seamlessly rejoin the platoon and reach its intended destination. Note that this action is scaled to the maximum and minimum speed conditions defined in CARLA. This action space, even if continuous in nature could be modelled as a discrete space and solved using traditional RL algorithms, as Deep Q-Network (DQN), which can be considered a baseline to contrast with Proximal Policy Optimization (PPO), an SOTA, an on-policy algorithm renowned for its sample efficiency and model-free architecture.

PPO operates by collecting sufficient experience from the environment to populate a buffer, leveraging this data to iteratively refine its policy. After each learning iteration, the buffer is refilled with new experiences to ensure continuous improvement. PPO adopts an Actor-Critic framework, wherein the critic evaluates the value function for a given state. This evaluation informs updates to the actor, which is responsible for determining the optimal actions at each timestep. This iterative process enables PPO to effectively navigate complex, dynamic environments such as the one described here.

One of the main limitations of PPO is the requirement for a stable connection to the environment. In our experience, working with CARLA and RL presents significant challenges, not only in terms of simulator stability but also in the achievable sample rate, which can delay training. CARLA offers some features, such as the ability to run the simulator offscreen, however, these do not fully mitigate the issue.

Another important aspect of PPO is the reward function, which evaluates the quality of the decisions made by the agent. To enhance PPO's stability and accelerate its training, all rewards are normalized. The logic encoded in the reward function is summarized in the following table:

Event	Reward
Reach Goal	+++
Entering roundabout	++
Correct position in the platoon	+
Following the platoon	+
Collision detected	--
Staying still	-

Table 3: Reward decomposition

A well-designed reward function is essential for the algorithm to learn effectively, here, each event in **Error! Reference source not found.** results in a variation of the r eward obtained, representing the effect with the amount of positive (+) or negative (-) reward. Next, a brief description of each entry is provided:

- **Reach goal (+++)**: Vehicle numbered as 3 (v3) is near the destination along with the other platoon members.
- **Entering roundabout (++)**: Vehicle numbered as v3 reaches and passes away the roundabout entrance.
- **Correct position in the platoon (+)**: $d_{3,1}$ and $d_{3,2}$ situates V3 behind V1 and V2. We compute this by means of the orientation of V3 and the distances in X and Y between all vehicles. This reward will only work if vehicle X is not in-between the rest of vehicles.
- **Following the platoon (+)**: Keeping a minimum gap in the platoon and similar velocities returns a small positive reward. This encourages V3 to keep distances with the rest of the vehicles in the platoon.
- **Collision detected (--)**: CARLA informs of a collision occurring in the simulation between vehicle 3 and any other element triggering collisions.
- **Staying still (-)**: If at any point of the simulation the vehicle decides to stop a small negative reward is returned. This way, the vehicle is encouraged to move, and only stop if the situation demands it.

The source code for this project is developed in Python, one of the most popular programming languages for Artificial Intelligence development, especially in the context of this project, as CARLA provides an API to work with Python. Additionally, specific tools for RL development are utilized. Among them, Stable-Baselines3 should be noted, since it offers baseline implementations for many SOTA RL algorithms, including the PPO implementation adopted for this project. Some of the key tuned hyperparameters are listed below in Table 4:

Variable	Value
Normalize	true
N_envs	1
Batch_size	32
N_steps	512
Gamma	0.99
Learning_rate	5e-5
N_epochs	50
Vf_coef	0.871923
Policy	MlpPolicy

Table 4: Key hyperparameters and their values

- **Normalize**: As mentioned above, the rewards are normalized for stability on training.
- **N_envs**: The number of environments trained in parallel, we can only connect to 1 CARLA server.
- **Batch_size**: The number of training samples processed at once during each gradient update
- **N_steps**: The number of timesteps (or steps) to collect in each environment before performing a single update.
- Gamma: The discount factor used to compute the present value of future rewards.
- **Learning_rate**: The step size used in the optimization process.
- **N_epochs**: The number of times the entire batch of data is passed through the model during training.
- **Vf_coef**: The coefficient for the value function loss, which controls how much the value function (used by the critic) impacts the total loss function.
- **Policy**: The network type used for the actor and critic (i.e. Multilayer Perceptron). For this first approach we use the default networks' configuration for Actor and Critic.

In summary, the proposed RL solution aims to optimize the entry of a vehicle into a roundabout within a platoon, taking in account the obstacles (i.e. other vehicles) inside the roundabout. Using PPO, the system can iteratively improve its decision-making process. Also, carefully designing the reward function is crucial to ensure the agent learns adequately.

2.3 EXP3

EXP3 title is "Self-assessment and reliability of perception data with complementary V2X data in complex urban environments", under the responsibility of UULM partner.

It aims at demonstrating safe automated driving in complex urban environments with occlusion.

This will be achieved through the usage of self-assessment methods in the onboard perception system, which will produce reliable assessment outputs. Additionally, V2X data from an infrastructure pilot site will be utilized. More information about EXP3 can be found in Deliverable D2.1 "User and System Requirements for Selected Use Cases" **Error! Reference source not found.**. In the context of EXP3, the perception s ystem is not the only essential component, but behavioural decision-making and trajectory planning are also crucial. Behavioural decision-making responds to various influences, such as the environment or internal states, and can control the vehicle's actions. The trajectory planning calculates the vehicle's path based on the output of the behavioural decision-making. In this section, the focus is set on the behavioural decision-making component in the context of EXP3. As the concrete scenario for behavioural decision-making, the vehicle turns right at the intersection (see Deliverable D2.1 for an illustration of the intersection) and merges without hindering other traffic participants. The difficulty in this scenario is that the view to the left side is impeded, and the vehicle cannot see if another vehicle is approaching. The data from an infrastructure pilot site sent to the vehicle with V2X communication is used to resolve the occlusion. The decision if the vehicle should enter the intersection and merge between the other vehicles is taken with the behavioural decision-making presented in this section.

2.3.1 Architecture

The data flow and components in the overall architecture of EXP3 are first presented in Deliverable D2.2 "Full Stack Architecture & Interfaces" Error! Reference source not f ound.. The architecture is visualized in **Error! Reference source not found.**, which is a modification from **Error! Reference source not found.**, and in which the tracking and the CPMs deliver the data for behavioural decision-making and trajectory planning. The processing pipeline gets raw sensor data from the vehicle sensors, such as camera, LiDAR, or RADAR sensors. The sensor data is pre-processed on the ego vehicle. The pre-processing could be a rectification for the camera data or an ego-motion compensation for the LiDAR data. The pre-processed data is used to extract single objects, wherefore neural network object detection models are used. The detections of the sensors are associated and filtered over time with a tracking algorithm. A novelty of the tracking approach used in the onboard tracking is the integrated self-assessment procedure, which is developed in WP3 of the project. The output of the onboard tracking algorithm, containing a track list and a self-assessment score, is given to the fusion module. Here, the onboard track list is fused with data delivered by Collective Perception Messages (CPMs) from the external roadside units (RSUs). In the track list fusion, the onboard track list is fused with the infrastructure track list to a fused track list. The fusion applies the self-assessment score to resolve objects detected by both sensor suits. The additional information is a track list that contains the data for the objects detected by the infrastructure and is sent over a V2X communication in the form of a CPM to the vehicle. In contrast to the track list generated onboard, the object list from the infrastructure does not contain a self-assessment score. The fused onboard information and the infrastructure data determine the vehicle's behaviour in behavioural decision-making, where the self-assessment score and the infrastructure's availability determine the behaviour.

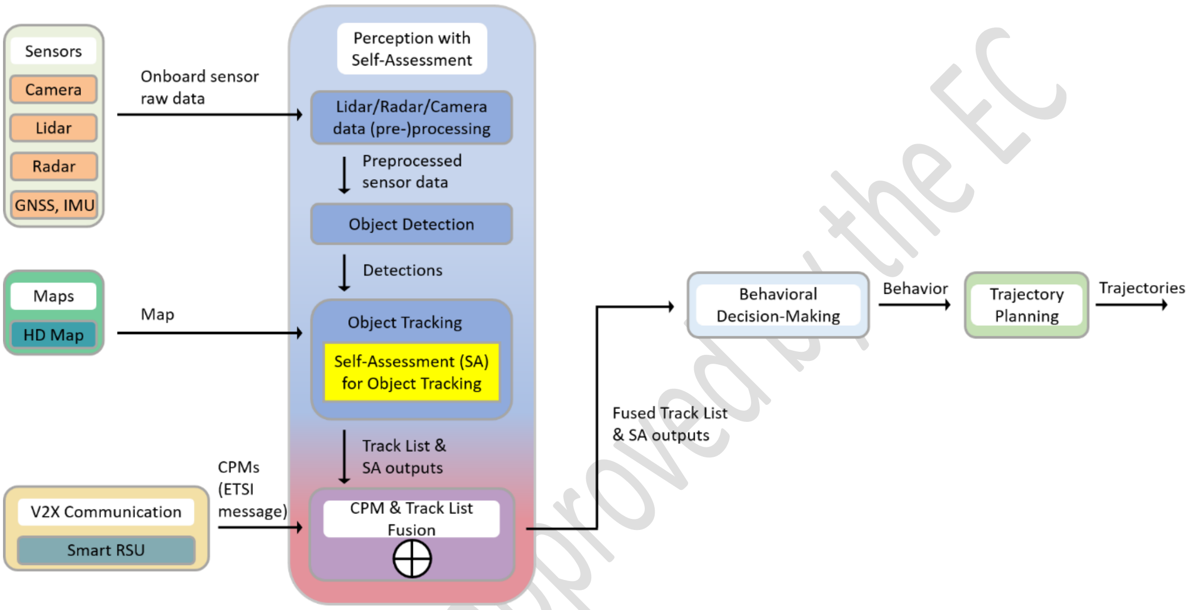

Figure 11: Architecture of EXP3

On top of the behavioural decision-making, the trajectory planning step is performed, which uses the fused track list and the planned behaviour to plan the trajectories.

2.3.2 Algorithmic Approach

The algorithmic approach of UULM partner for EXP3 is described in this chapter, with a first section dedicated to the state-of-the-art (SOTA) and the other two sections to the real developed algorithms.

State of the Art

Automated driving moves the whole driving responsibility from the driver to the vehicle. This means that the vehicle has to make decisions like passing another vehicle Error! Reference source not found. or driving into an intersection Error! Reference so urce not found.. To model the human-like behaviour of the vehicle, different driver models are available, like the intelligent driver model [23] or Error! Reference source n ot found.. For lane changes on highways, a behavioural decision-making model is proposed in MOBIL Error! Reference source not found., relying only on the a ccelerations of the different surrounding vehicles and the ego vehicle. A more advanced approach for behavioural decision-making is POMDP Error! Reference s ource not found., which uses Markov decision processes in an environment only partially observable. A possible way to solve the POMDP is investigated by González et al. Error! Reference source not found., using a Monte-Carlo-based algorithm Error! Re ference source not found. to determine the vehicle's behaviour. Furthermore, in this approach Error! Reference source not found., the other vehicles' physical state and the lane-changing intentions of the other vehicles can be considered, wherefore it is possible to imitate human-like behaviour for highway scenarios. Lenz et al. Error! Reference source not found. also work on solving the highway scenario and, therefore, apply a Monte-Carlo tree search to the decision-making problem formulated as a Markov decision process. Another method for behaviour decision-making Error! R eference source not found. creates a geometric model with a high-precision map and estimates the motion of other vehicles with a Dynamic Bayesian network Error! R eference source not found.. With the former model and the motion, the behaviour decision-making is done with a POMDP algorithm. In the real world, the POMDP algorithm can be computationally intractable. To overcome this limitation, Zhang et al. Error! Reference source not found. introduce an efficient uncertainty-aware d ecision-making framework, which is real-time capable and divided into two parts. First, a domain-specific closed-loop policy tree is created, which determines the behaviour of the ego-vehicle. Second, conditional-focused branching identifies potentially dangerous situations with other vehicles, which are excluded from the possible behaviour. Another possible scenario in automated driving is passing a truck on a single-lane road. Here, the passing vehicle must pass a gap between the oncoming vehicles. Zhang et al. Error! Reference source not found. use a forward h idden set to include all possible locations of a hidden car in behaviour planning and extrapolate the forward hidden set to the future. The car's behaviour plan is safe because of a viable fallback strategy as long as the future forward hidden set does not overlap with the danger zone of the ego vehicle. For behaviour decision-making, a state machine can also be applied Error! Reference source not found. Error! Reference so urce not found.. Noh and Kyounghwan Error! Reference source not found. apply in the first step a situation assessment, which includes all surrounding vehicles and assesses the possibility of collisions to avoid them. The situation assessment is determined with Bayesian networks and is propagated to the strategy decision component. The strategy decision component is realized with a state machine and determines a goal-directed and collision-free behaviour. Besides normal state machines, hierarchical state machines can be used for behavioural decision-making Error! Reference source not found., which can have several states in parallel. While the state machine limits the possible manoeuvres, Hubmann et al. Error! Reference source not found. propose a non-rule-based decision-making algorithm that can decide between an infinite number of possible manoeuvres. It is applicable in different scenarios, e.g., cruise control or decision-making at traffic lights, and is based on the A* algorithm Error!

Reference source not found.. Together with behavioural decision-making, the trajectory can be determined so that both tasks benefit from each other Error! Reference source not found.. Ma et al. Error! Reference source not found. detect violations of dynamic and static traffic rules in real-time with a four-stage algorithm, which predicts with the MPC algorithm Error! Reference source not found. a new trajectory that does not violate any traffic rule.

Behavioural Decision-Making

The decision-finding process of the overall behavioural decision-making is realized as a state machine in Error! Reference source not found.. At the start, the state machine i s initialized to the *Drive Mode*, where the vehicle is limited by the standard traffic regulations and the surrounding environment in its behaviour. The vehicle's start is only possible if the self-assessment (SA) score is above the threshold *thr*. If the SA is below *thr*, the state machine changes the state from the *Drive Mode* into the *Fallback Mode*. This happens because, with a low SA score, the vehicle's own environmental perception is inaccurate, and therefore, it must be planned with unforeseeable environmental conditions. Changing the state from the *Fallback Mode* back to the *Drive Mode* is only possible after an expert check. The expert check is necessary because the worse SA score can indicate a serious issue like a miss-calibrated sensor, which can lead to dangerous situations. To lower the effort for the expert check, a remote check could also be possible. Notable is that the behaviour in both modes can differ between various situations. The vehicle can stop from both states.

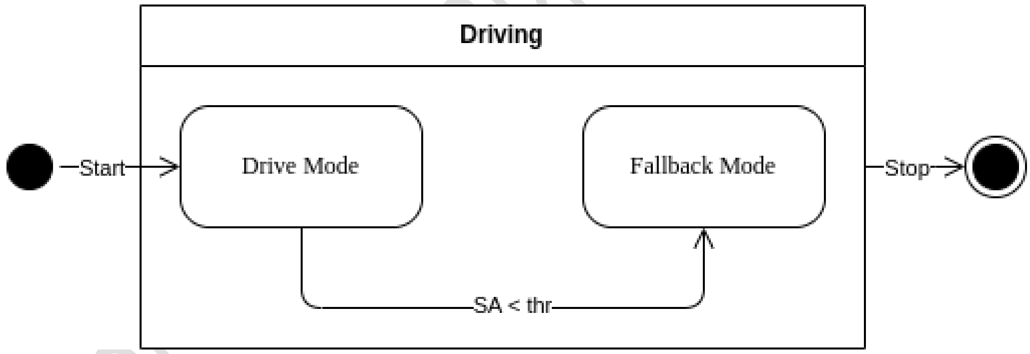

Figure 12: State machine diagram of the behavioural decision-making

The following sections describe the behaviour at an intersection with infrastructure sensors and a road segment without infrastructure support.

Behavioural Decision-Making with Infrastructure Sensors

If external infrastructure data is available, the behavioural decision-making gets a fused track list, which contains information from both the onboard track list and the track list from the infrastructure. **Error! Reference source not found.** gives a s chematic overview of the vehicle's behaviour at the intersection. A high SA score means that the vehicle is in the *Drive Mode*. In this mode, the vehicle can enter the intersection without stopping when no other vehicle is arriving (possible if the database is reliable). When an arriving vehicle is detected, the ego vehicle must stop at the intersection and wait until the other vehicle passes.

If the SA score is below *thr,* the vehicle only relies on external data, no data from the onboard perception is used, and the ego vehicle enters the *Fallback Mode*. This means that the vehicle plans to stop because after leaving the intersection area, no external data is available, and therefore, driving solely with the external data is not possible. For stopping the vehicle, safe stop points are defined in the intersection area so that the vehicle can use the external data to reach a safe state. The location of the defined safe points can be seen in Figure 14. One safe stop point is before the vehicle enters the intersection, and another is after the vehicle enters the intersection. Which of the stop points is used depends on how far the vehicle has entered the intersection before the SA score falls below *thr.* Using external information has the advantage that even with an uncertain ego perception, the vehicle does not block the intersection but can leave the dangerous intersection zone in a safe manner.

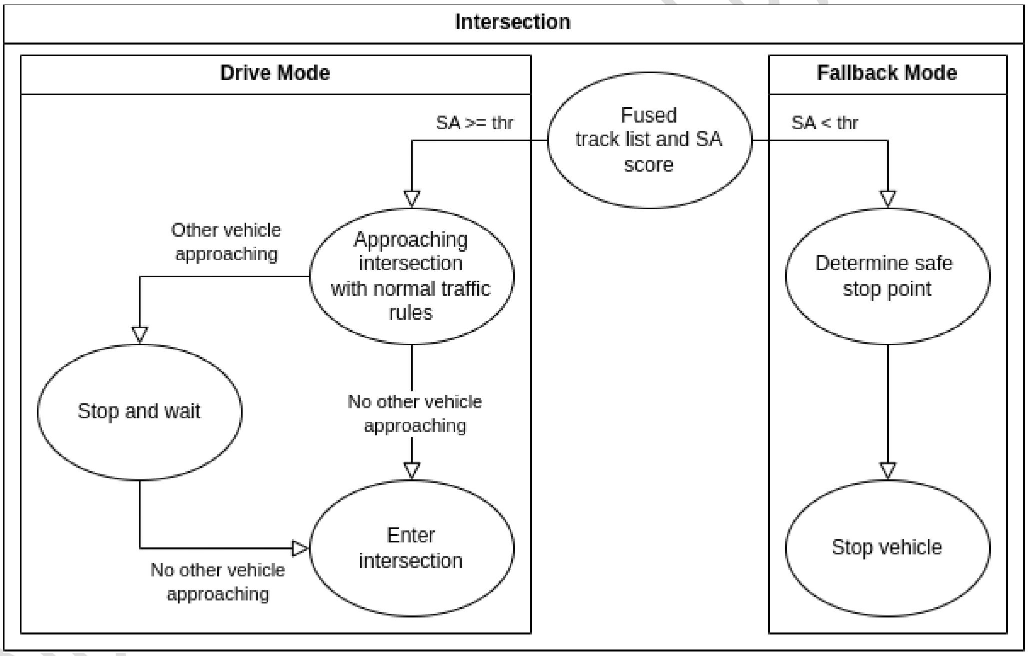

Figure 13: Schematic overview of the behavioural decision-making at an intersection with infrastructure sensors

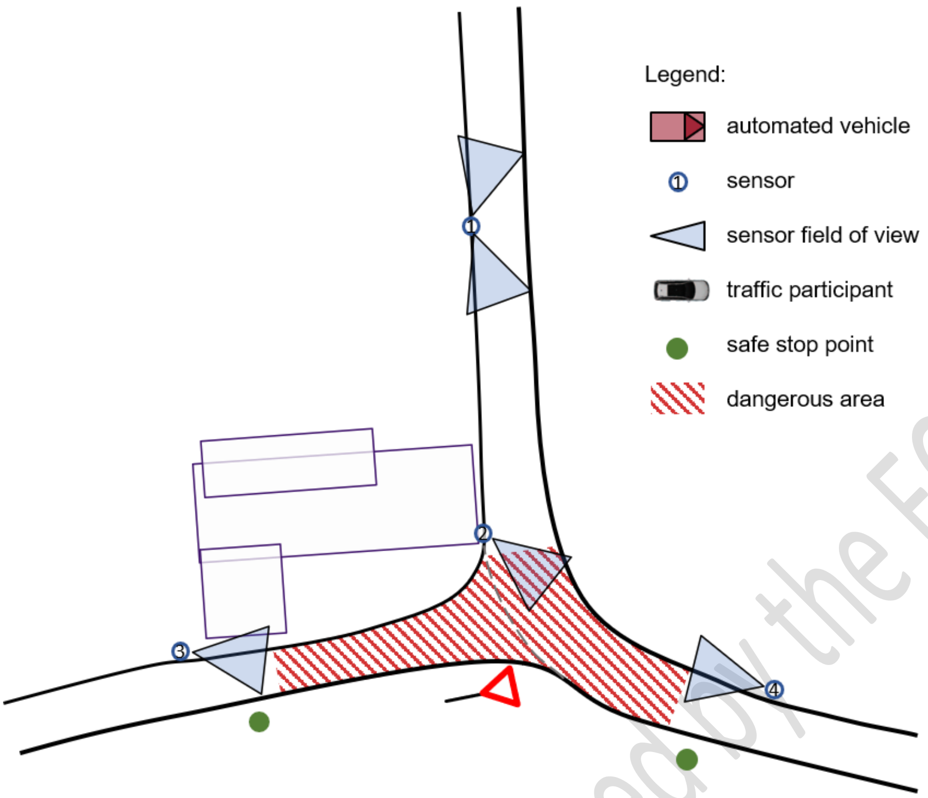

Figure 14: Visualization of the dangerous area and the safe stop point of the intersection

After reaching the safe stop point, a human driver has enough time to oversee the situation and can hand over the driving task.

Behavioural Decision-Making Without Infrastructure

While the former behaviour requires additional external data, the vehicle behaves differently if no additional data from infrastructure is available, as visualized in Figure 15. In the case the SA score is above *thr*, the vehicle operates in the normal driving mode. This means the vehicle operates only with the standard traffic rules and the environmental limitations as restrictions. An SA score below *thr* changes this behaviour because the onboard perception can have errors that can lead to dangerous situations. Therefore, the vehicle enters the safe mode, which means in this condition that the vehicle drives slower and holds a higher safety distance to the leading vehicle. Reducing the velocity from 50 km/h allowed in urban areas to only 30 km/h and doubling the safety distance to the leading vehicle gives more time to react to unexpected situations, which can happen with an uncertain ego perception.

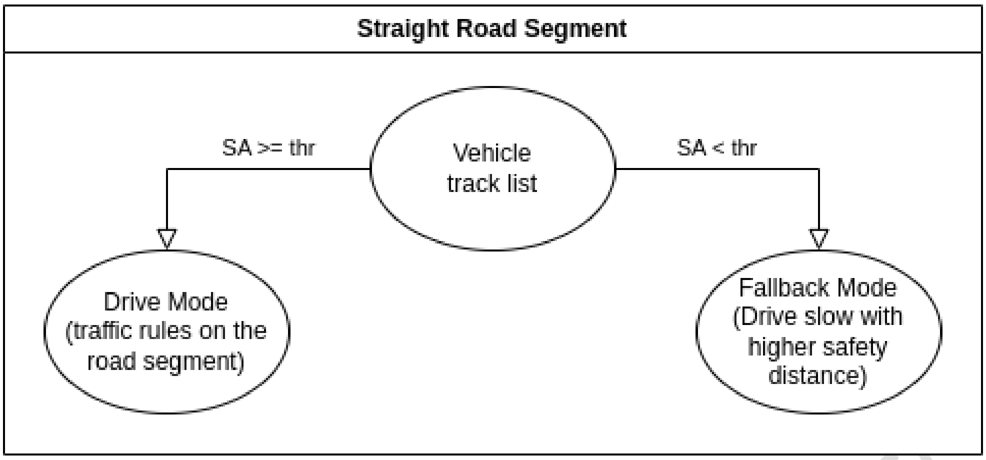

Figure 15: Schematic overview of the behavioural decision-making without infrastructure sensors

Finally, the diver is informed about the uncertain ego perception so that he can prepare for a potential takeover.

2.3.3 Conclusion and Future Work

The behavioural decision-making presented in this section is implemented as a state machine, which divides between the *Drive Mode* and the *Fallback Mode*. In the *Drive Mode*, the vehicle behaves according to the current traffic regulations and the surrounding environment. In the *Fallback Mode*, some issues with the vehicle's perception are detected, and the vehicle adapts its behaviour. So, without any infrastructure support the vehicle reduces the velocity and increases the safety distance. With infrastructure support, the automated vehicle determines a safe stop point and stops at the stop point. This prevents the intersection from being blocked by an automated vehicle with a malfunction.

The behavioural decision-making will be integrated into the ROS2 software stack of Ulm University's test vehicle, between the fusion of the different track lists and the trajectory planning in the scope of the project's WP5. In this context, the whole approach will also be tested in simulations and real-world tests.

2.4 EXP4

EXP4 is carried out by HITACHI, Tecnalia and WMG partners and its title is "Decision making for motion planning when faced with roadworks, unmarked lanes and narrow roads with assistance from perception self-assessment". The objective of the EXP4 is to navigate in an unstructured road wide enough for at least two vehicles driving in parallel and with occasional disturbances, like roadworks. The challenge regarding behavioural planning is the selection of a collision free path within the boundaries of the roads in real time.

2.4.1 Architecture

Motion planning on unstructured roads for automated vehicles presents a set of complex challenges due to the lack of well-defined lane markings, traffic signs, and other standard infrastructure. Unlike structured environments, where clear road geometry and traffic rules guide decision-making, unstructured roads require the vehicle to rely heavily on perception and contextual understanding. This uncertainty forces motion planners to operate with incomplete or noisy data, making real-time decision-making and trajectory generation significantly more challenging.

A common approach for both, behavioural and motion planning on unstructured roads is to generate a simple reference trajectory and then run a MPC to apply vehicle dynamic constraints [42]. However, most of the works assume a stable knowledge of the lane edges. In this work a novel reference generation with Bezier curve smoothing is proposed alongside an MPC for optimization for it to be tested with real world data.

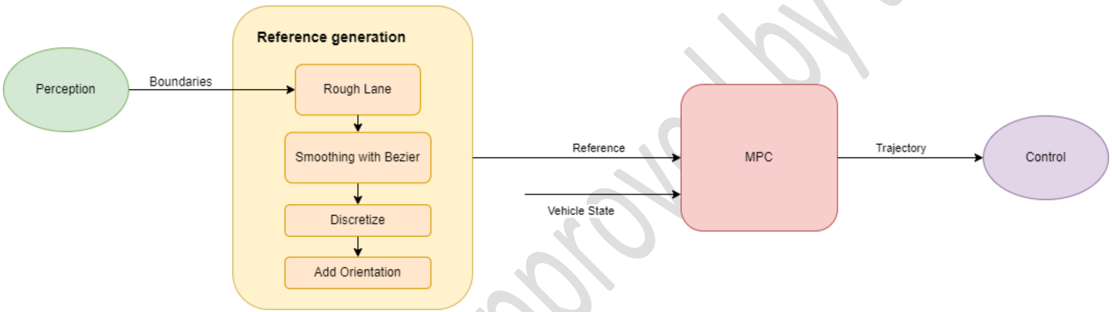

Figure 16: Behavioural planning architecture

Error! Reference source not found. shows the data flow of this method. First, a rough t rajectory is calculated using the lane boundaries as reference. After that a smoothing is applied to that path to be more feasible using 5th order Bezier curves. The output needs to be discretized to match the step time used in the MPC and the relative orientation added. Finally, the result will be used by the MPC as the state reference in the cost function.

2.4.2 Algorithmic approach

In this case the road where the data has been recorded is two lanes wide. One of the premises in this experiment is that the vehicle should stay on the right lane. Therefore, the points of the first rough lane meet the following condition:

$$\begin{cases} d_{right} = \frac{w_L}{4}, if\ w_L > 4 \\ d_{right} = \frac{W_L}{2}, if\ w_L < 4 \end{cases} \quad (8)$$

where d_{right} is the distance of the point to the right edge and w_L is the road width.

This means the reference will stay closer to the right unless the road shortens due to roadworks. The calculation of the lane width presents several problems. To start, the number of points in each edge might not be the same, so it is not possible to calculate the width by Euclidean distance between points with the same index. First, we get the bisector of the first two points of the borders. For each point in the array, we calculate the distance to the borders by calculating the minimum distance to each segment of two consecutive points in the borders. The width of the lane for a certain point is the sum of the distances to the right and left border. This approximation is considered good enough, since no sharp turns are contemplated in the EXP4. Either way, a maximum length of 20m is considered in the trajectory generation. However, these points alone often result in sharp angles or irregularities, which are unsuitable for many applications, especially in automated vehicle navigation or robotics.

To address this, Bézier splines are employed. Bézier splines are mathematical constructs that allow smooth interpolation between points by leveraging control points. After segmenting the path, a quintic Bézier curve is applied to the first segment. The process is repeated for all path segments, ensuring smooth transitions at junctions by maintaining curvature continuity, where the first or second derivatives of adjacent curves match. An example of the curve is provided in Figure 17.

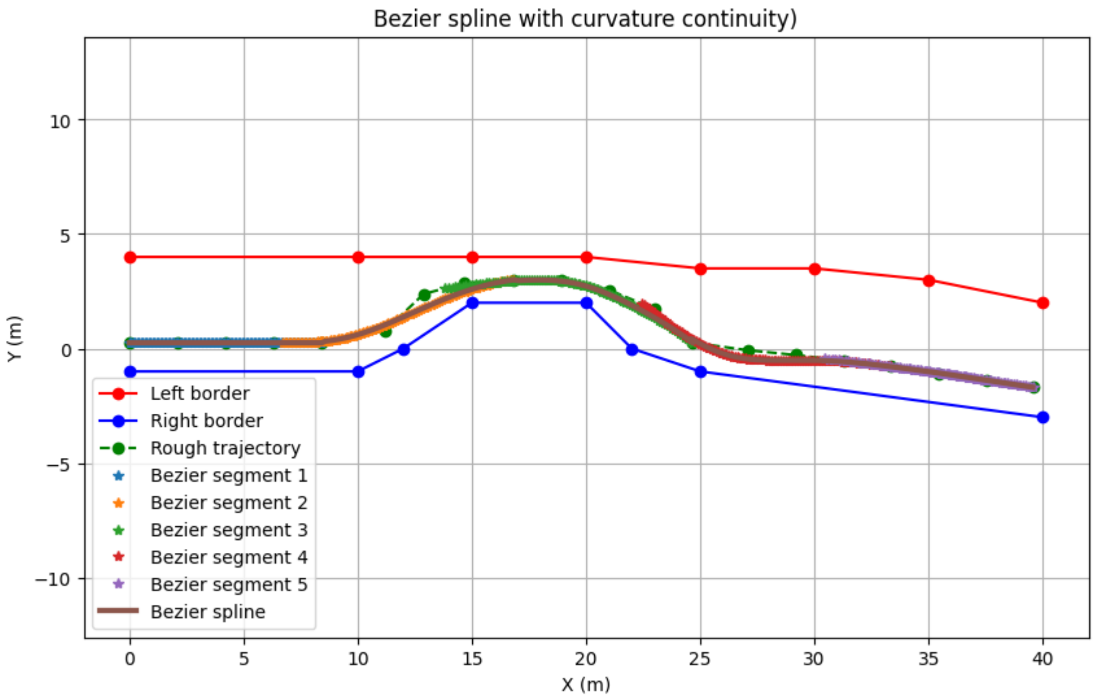

Figure 17: Bezier spline trajectory generation for unstructured roads

The orientation for each point is calculated using the vector between that point and the next in the path, except for the last, which has the same orientation as the previous point.

For the MPC a well-known kinematic model has been chosen. The state of the vehicle is represented by three variables, position in cartesian coordinates (x, y) and heading orientation relative to the initial state (θ). The derivation of the state is represented in the following equation:

$$\dot{\bar{x}} = \begin{cases} v\cos(\theta + \beta) \\ v\sin(\theta + \beta) \\ v\cos(\beta)\dfrac{\tan(\delta)}{L} \end{cases} \tag{9}$$

Where, parameter *L* is the wheelbase of the vehicle and β the sideslip angle is calculated by the following formula, where l_r is the distance from the rear axle to the center of the mass of the vehicle:

$$\beta = \operatorname{atan}\left(\frac{l_r\tan(\delta)}{L}\right) \tag{10}$$

The optimization problem can be described as:

$$min\ (J(\bar{x},\bar{u})) = \int_{i=0}^{H} w_{dist}d_{ref}^2 + w_{orient}\theta_{dif} + w_{steer}\delta^2 + w_{acc}v^2 \tag{11}$$

$$\text{s.t.} \qquad \begin{aligned} \delta_{min} < \delta < \delta_{\max} \\ 0 < v < v_{\max} \\ d_{left} < w_L \\ d_{right} < w_L \end{aligned} \tag{12}$$

where distance (d_{ref}) and orientation (θ_{dif}) to reference is minimized. Heavy change in both inputs (δ and v) are punished too.

State constraints (12) are used to consider the vehicle's physical restrictions, as well as road restriction, imposing distance to the left and right borders (d_{left}, d_{right}) not to be above the lane width (w_L).

2.5 EXP7

EXP7 "*Localization/perception self-assessment for advanced ACC and other vehicles' behavior prediction under adverse weather or adverse road conditions*", leaded by ICCS partner, has the goas to develop and evaluate a novel multi-agent motion prediction module, which predicts both the intention of vehicular and VRU traffic participants, as well as their short- and long-range trajectories.

2.5.1 Manoeuvre Classification

ICCS has developed a manoeuvre classification algorithm for AVs equipped with RGB cameras mounted as sensor on top of vehicles driving in highways and boulevards. The system was trained with ML algorithms to process a time-series of frames leading to a manoeuvre and predict a probability set for each valid lane change action. We draw inspiration from the SOTA in video classification with CNNs (Convolutional Neural Networks) and LSTMs (Long Short-Term Memory) algorithms and train our models on the PREVENTION available in public, which is the only database with real driving videos of real-world highway scenarios.

Recent research has explored the impact of combining CNN architectures with time series modelling networks, e.g. LSTMs for ML tasks (classification, regression) on video data. Feichtenhofer et al. [] pioneered the development of ResNetlike models branded ”SlowFast” for action recognition and video classification, since they use two branches based on Faster R-CNN for detection of multi-scale spatiotemporal features, by processing the input features in two different frame rates (fps), per branch. Driven from research in biology, the parallel processing of fast-changing temporal information and spatially detailed information yields SOTA performance in the Kinetics and AVA baselines.

In addition, literature has researched the application of CNN and LSTM layers in sequence, as opposed to Slow Fast architectures, to embed pixel visual information in lower dimensional vector representations and therefore enable the application of time-series ML methods. The CNN-LSTM backbone with shared weights across time (CNN, LSTM resp.) extracts spatiotemporal features from the video inputs, and a classification head (MLP) processes the features to produce logit outputs. This dynamic approach has better performance than its opposed temporally static approach when LSTMs are ignored.

Previous approaches in automated driving manoeuvre classification have adopted similar approaches. Izquierdo et. al (2019) [43] have experimented with manoeuvre classification (Left Lane Change, Right Lane Change) on RGB camera frames on the PREVENTION dataset. They have developed a GoogleNetLSTM model – drawing inspiration from the CNN-LSTM – which encodes (GoogleNet) the cropped RGB frames of vehicles in the vicinity of the ego vehicle for 60 frames (2s) and decodes (LSTM) the predictions for the lane change manoeuvres in a future horizon of 3 seconds. They have also experimented with novel techniques of encoding contextual information in different colour channels and processing the resulting representation with CNN and classification head for manoeuvre prediction.

System Architecture

We follow the architecture inspired from [12], with minor modifications in the input fps value per branch and convolution kernels sizes which are shown below. The overall software layers for manoeuvre prediction in highways with SlowFast Resnet CNN, using the PREVENTION dataset are shown in Figure 18. The frames in the data segment are presented for some frames before the manoeuvre occurs. The frame on bottom right is closest to the manoeuvre and the frame on top left is furthest. The prediction classes stand for : LLC="Left Lane Change" , RLC = "Right Lane Change" , LK= "Lane Keep". The overall architecture for the Resnet SlowFast CNN algorithm is shown in Figure 19.

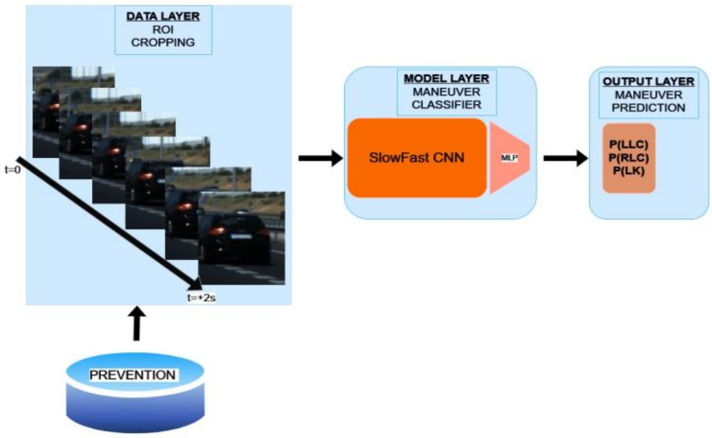

Figure 18: Software pipeline for manoeuvre prediction in highways with SlowFast Resnet CNN, using the PREVENTION dataset

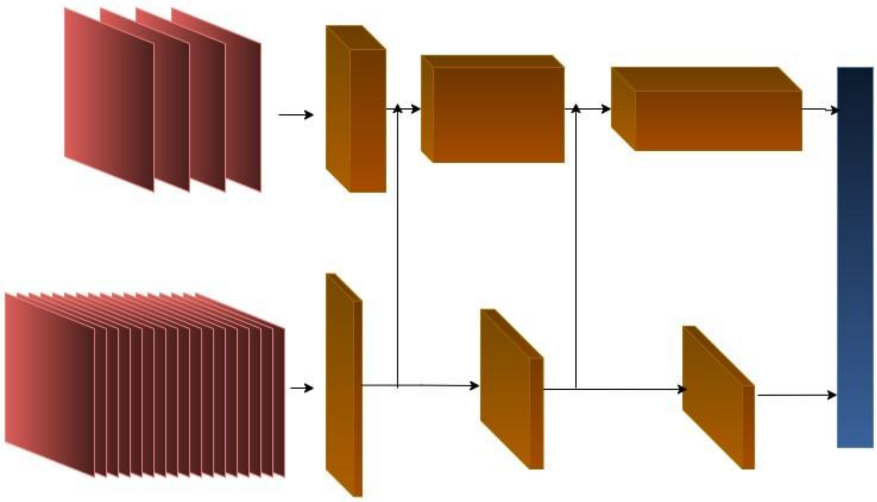

Figure 19: Manoeuvre Classification Model: A SlowFast architecture and activation volumes

We visualize the neural activations beginning from inputs to each branch (red, left) and leading to manoeuvre prediction probabilities (blue, right):

- Scene Encoding. We adopt the SlowFast modules shipped by PyTorch and we set τ = 20 for the slow pathway, i.e. read 1 out 20 frames, whereas α = 1 for the fast pathway [56]. These parameters were tuned in the validation set of the database. The backbone of the model is based on Resnet3D and is given below: we include lateral connections between the slow and fast branches after the layers mentioned in [56] (conv1, res1, res2, res3, res4).

- Manoeuvre Decoding. The neural activation volumes of the last layer of the slow and fast branch backbone are flattened and concatenated to produce a one-dimensional feature vector that is processed by the classification MLP.

The inputs are read with fps, for slow and fast branches (top, bottom), respectively head which produces unimodal manoeuvre predictions for one vehicle for the prediction horizon. The prediction is achieved by the linear layer (MLP) of 518 input neurons and 3x30 output neurons, for a prediction of 1 second in the future.

PREVENTION consists of 5 recordings with 3 clips each, of approximately 40 minutes driving logs and labels (object detections, lane annotations (polylines), trajectories). Lane change manoeuvres are split into lane change left, lane change right, cut-in and cut-outs. For preprocessing, we split the raw video clips on the manoeuvre sequences, and we select clips of lane keeping while keeping the dataset balanced in classes. We utilize object detections, lane change annotations and the videos for the learning task. Video is read at 30 fps, and we crop the video sequences using a region of interest cantered on the non-ego vehicle detected by the annotators' object detection framework, with a small added margin of 50 pixels per side and 100 pixels in the bottom detection side of the ROI. This ensures incorporation of lane information and therefore implicitly encoding vehicle lateral positioning for spatiotemporal lane change modelling. Lane change events with missing object or lane annotations for < 10 frames are linearly interpolated (obj. bounding box coordinates) and if > 10 frames, are filtered out during preprocessing.

	Slow Path.	Fast Path.	output dimensions
data reading layer	s = 20, 1^2	s = $1,1^2$	Slow: 2 x 224^2 *Fast* : $16x224^2$
conv1	$1x7^2$,64 s = $1,2^2$	$5x7^2$,8 s = $1,2^2$	Slow: 2 x 224^2 *Fast* : $16x224^2$
max$_p$ool1	$1x3^2$ s = $1,2^2$	$1x3^2$ s = $1,2^2$	Slow: 4x 56^2 *Fast* : $32x56^2$
res2 (x3)	$1x1^2$,64 $1x3^2$,64, $1x1^2$,256	$3x1^2$,8 $1x3^2$,8 $1x1^2$,128	Slow: $3x56^2$ *Fast* : $32x56^2$

res3 (x4)	$1x1^2$,128 $1x3^2$,128 $1x1^2$,512	$3x1^2$,16 $1x3^2$,16 $1x1^2$,64	Slow: $4x56^2$ *Fast* : $32x28^2$
res4 (x6)	$3x1^2$,256 $1x3^2$,256 $1x1^2$,1024	$3x1^2$,32 $1x3^2$,32 $1x1^2$,128	Slow: $4x14^2$ *Fast* : $32x14^2$
res5 (x3)	$3x1^2$,512 $1x3^2$,512 $1x1^2$,2048	$3x1^2$,64 $1x3^2$,64 $1x1^2$,256	Slow: $4x17^2$ *Fast* : $32x7^2$

Table 5: SlowFast network architecture for manoeuvre classification

Finally, we remove video clips in turns of highways, due to bad quality annotations. The dataset is shuffled and prepared for offline supervised learning. Table 5 explains the splits used per dataset video, where the strides are represented in format: s = temp$_s$trides,spatial$_s$trides.

Algorithms and Implementation

We trained with SGD optimizer for 100 epochs, using initial learning rate 0.001 with a multi-step (multiplicative) decay of 10^{-1} at *epochs* = [20,40,60,80]. We trained on a single RTX 3080 GPU with a mini batch size of 32. We use no weight regularization.

Evaluation is conducted on the manoeuvre classification task by considering the classification performance metrics: Accuracy which measures the correct manoeuvres out of all predictions, Precision, Recall and Area Under Receiver Operating Characteristic (AUROC) describing the overall capability of the developed classifier.

Recording	Clip	Split	LC	LK
1	1	train	8	10
2	1	train	9	10
2	2	val	6	10
3	1	train	11	10
3	2	val	12	10
4	1	train	8	10
4	2	val	7	10
4	3	val	7	10
5	All	test	9	10
Total train: LC: 36, LK:40 Total val: LC: 32, LK:40				

Table 6: PREVENTION Dataset class statistics after preprocessing

Metric	Expected Value to surpass SoTA	Citation to SoTA

Precision	76.43%	[44]
Recall	73.2%	[44]

Table 7: Metrics reported previously on maneuver prediction on highways using ego view videos

Preliminary results and baselines are shown in Table 6 and Table 7 (LK: Lane keeping, LC: Lane change). Full evaluation with quantitative results will be performed later as per the evaluation plan described in D6.1 of the EVENTS deliverables.

2.5.2 Trajectory Prediction

Trajectory prediction aims to predict trajectories in form of 3D waypoints (x, y, heading) for the non-ego vehicle as part of the planning pipeline for the ego vehicle. Trajectory Prediction is the step succeeding manoeuvre prediction and is conditioned on the predictions of the former module. It serves as a feature for contextual traffic information gathering prior to making decisions on the control action of the ego vehicle. We performed experiments on trajectory prediction in highways using the Ego view on images from a public database to complete the manoeuvre prediction module.

	Parameters conv=K, F pool=K	Dimensions out
data layer	Na	(255, 3)
conv1	(20,5)	(235,5)
pool1	(20)	(225,5)
conv2	(20,5)	(210,5)
pool2	(20)	(200,5)
conv3	(10,5)	(180,5)
pool3	(10)	(160,5)
conv4	(5,5)	(140,5)
pool4	(5)	(120,5)

Table 8: GNN architecture for trajectory prediction encoder

Operating on graphs instead of raster maps or vectorized maps [43], graphs offer a more versatile representation of map features and therefore more flexible modelling, combining features from multiple agents across different lanes of highways, and aggregating information with graph convolution layers [45].

System Architecture

The goal of the trajectory prediction module is to predict the time series $\{\hat{y}^m_{t+1}, \hat{y}^m_{t+2}, ...\hat{y}^m_{t+T}\}\forall m$ where m corresponds to a non-ego vehicle and T is the prediction horizon.

Two main parts of the architecture for this module. The first is the *Graph-based Scene Encoder*, in which an algorithm to featurize the ego view scene is developed, by constructing a graph from the lane annotation information on the current timestamp t = t′. Specifically: Given lane annotations as polylines of the form $y = a*x^2 + b*x+c$, with y,x the longitudinal and lateral coordinates. We construct discretized lane polylines every 1m step from the ego vehicle longitudinally and in between successive lane polylines laterally to make a dense grid with 2D coordinates. We construct an undirected graph with vertices on those coordinates: $G(V = (x,y), E = \{e \in Ns(V) \cup Nn(V)\})$ and edges connecting neighboring (Nn) and successive (Ns) lane nodes. We limit the neighboring node edges to k = 3 the left and right neighbors. Next, we search the database for objects (vehicles) located at any of the graph vertex locations and append a binary feature to represent if a vertex is preoccupied by an obstacle. We construct a 4-layer GNN (Graph Neural Network) that operates on the constructed graph and encodes the contextual scene information captured per graph node feature.

The second is the *Trajectory Decoder To decode*, similarly to the manoeuvre prediction module, we flatten the activations of the final GNN layer and apply an MLP for trajectory regression.

Algorithms and Implementation

ICCS conducted experiments on the PREVENTION dataset, utilizing the waypoint trajectory annotations as features. ICCS used an RTX 3080 GPU, with the Adam Optimizer, learning rate 0.001 and exponential decay of rate 0.1, weight regularization 0.001 and dropout 0.1 in the 3rd and 4th layer of the GNN.

Some common metrics for prediction performance evaluation were considered, i.e. Average Displacement Error (ADE), Final Displacement Error (FDE).

The metrics quantitative results will be reported on D6.1.

2.5.3 Joint Prediction and Planning

Joint Prediction and Planning is a new part of ICCS work that was taken over in the context of WP4 based on the second project amendement (approval is pending at the time of writing) and which is still under development. What is reported here is the progress that has been made until this deliverable's submission date.

Joint Prediction and Planning aims at merging the two problems with a unified framework, where prediction informs planning and vice-versa. This is achieved either as a modular pipeline by pre-training sub-components individually prior to evaluation or by constructing an end-to-end differentiable model, with one training and evaluation cycle. To reflect on the real-life behaviour of vehicles during their interaction in traffic situations, a reactive closed-loop simulator is suitable, which enables multiple models of non-ego agent driving intentions, i.e. aggressive, passive or safe. ICCS aims to unify prediction and planning with a public simulator, which is modified with added prediction capacities. As planning, we define the models producing a sequence of ego actions which aim to imitate an expert trajectory time series. In this deliverable, we report the details of the algorithmic approach followed on this task.

It was previously established that learned ML modules for planning in open-loop are hard to generalize in closed-loop as they do not provide safety guarantees, because of their dependence on data [46], introducing covariate shift. We, therefore, conduct experiments on the world's first differentiable simulator, which allows closed-loop reactive driver control during simulation with any custom ML model for the agents in the simulation.

As [45] has explained, integrating the Prediction and Planning modules in ADAS is critical for safe, long-term driving simulations. Since no works have previously been reported on adapting the nuPlan simulator for Joint approaches, we modify the open-source software to support end-to-end differentiable, joint prediction methodology. Similar to previous work [42], we adopt a BEV maps representation and past vehicle tracks as features from the nuPlan database and train on imitation loss, but augmented with hand-crafted cost terms which ensure ego vehicle safety, given trajectory forecasts of nonego agents, thereby improving contextual reasoning by the AI model controlling Ego.

System Architecture

The objective of the joint prediction and planning paradigm is to create ego plans informed by non-ego trajectory predictions.

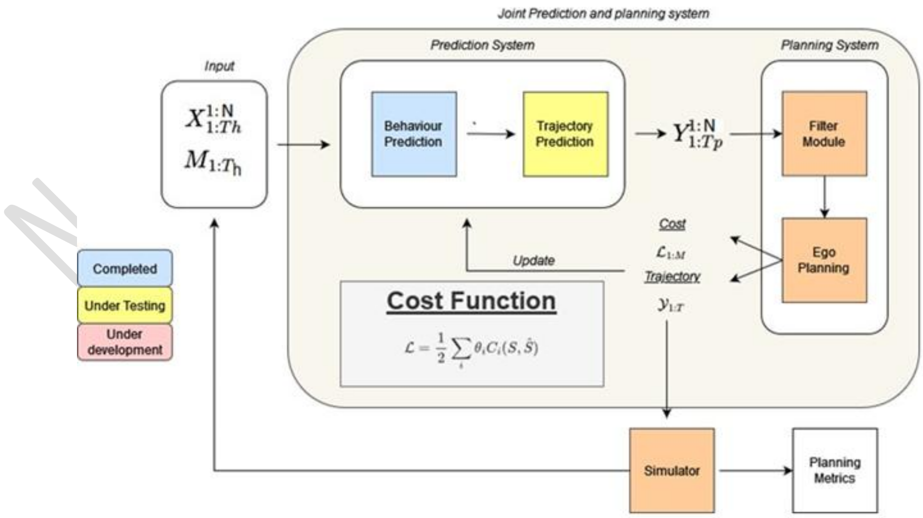

Figure 20: Virtual training and evaluation pipeline in a modular approach

The Cost function $L = \frac{1}{2}\sum_j \theta_j C_j(S_j, \hat{S}_j)$ is a planner-aware prediction hand crafted term and allows cost-based optimization of the planner trajectory with Gauss-Newton algorithm.

Figure 20 (Abbreviations are: X: Image Input, M: Vector Map, N: Number Agents, Th: Prediction input window, S: Predicted Trajectory of non-ego, Tp: Future horizon) shows the virtual pipeline for joint prediction and planning with *nuPlan* with a modular pipeline.

Algorithms and Implementation

ICCS trained the joint approach on the *nuPlan* dataset in Las Vegas, Boston, Pittsburgh and Singapore. All training and testing scenarios are selected to correspond to highways and urban boulevards, so as to satisfy EXP7 UCs. For validation and testing, the model outputs multi-agent trajectories for the future horizon.

	Parameter	Dimensions out
data layer	NA	146x30
conv1	K=2,F=20	74x20
pool1	K=2	35x20
conv2	K=2,F=30	15x30
pool2	K=2	7x30
conv3	K=2,F=40	3x40
pool3	K=2,	2x40

Table 9: Prediction Architecture which processes vectorized maps and agent tracks as features via a CNN model

For prediction, ICCS used the Resnet-50 and Resnet101 as baselines for a raster model. The predictor architecture layers are tabulated in Table 9, where the normalization and activation layers are omitted for simplicity. The model processes vector maps and historical non-ego features to predict their future trajectories in a +2s horizon.

As baselines for the evaluation, ICCS considered the aforementioned network, but modified in the last layer to conduct single-agent predictions, and the model without vectorized map semantics as input features.

Some metrics are considered in Table 10 and Table 11.

Metric	Description	Research Question	Units	Expected Outcome (SoTA)

Static Average Displacment Error(ADE) within bound	Average Displacement Error: L2 Error from ground truth of predictions for standalone Predictor training (Open-Loop). The scores are bounded within [0,1].	Is Open Loop (static) ADE performance worse than Closed loop (dynamic) performance?	m	< 1m []
Static Final Displacement Error (FDE) within bound	Final Displacement Error: L2 error at the end of the current training route. The score is bounded within [0,1].	Is Open Loop (static) ADE performance worse than Closed loop (dynamic) performance?	m	< 1m []
Dynamic Average displacement error (ADE).	Average Displacement Error in Dynamic Evaluation: L2 Trajectory Prediction error in Reactive Closed Loop Simulation.	Is Closed Loop Reactive (dynamic) ADE less in reactive evaluations and does it correlate better with Ego planning performance?	m	< 0.5m
Average Heading Error(AHE) within bound in Static Evaluation	Average heading Error in Static Evaluation: L2 error of predicted steering angles (heading) in Open-Loop evaluation.	What is the AHE in open-loop evaluation.	m	< 0.5m
Dynamic Average Heading Error(AHE) in Dynamic Evaluation	Averag Heading Error in Dynamic Evaluation: L2 Vehicle Heading error in Reactive Closed Loop simulation.	Is Closed Loop Reactive (dynamic) AHE less in reactive evaluations and does it correlate better with Ego planning performance?	rad	¡5rad

Table 10: Prediction metrics for Joint Prediction and Planning model evaluation in Open Loop evaluation protocol

Metric	Description	Research Question	Units	Expected Outcome
Expert Imitation	Displacement error of control outputs with expert demonstrations, when training in open-loop.	What is the Expert Imitation Error in Open-Loop training of ego agent?	m	< 0.3*m*
Navigation	Percentage of full track completed by the Ego con-troller. Percentage full track completed by the planner	What is the avarage percentage of tracks completed by the vehicle?	%	> 90
Ego Dynamics and Safety	Test if Longitudinal and lateral jerk of the ego vehicle within a predefined threshold.	What is the horizontal and lateral accelerations experienced by the vehicle and the passenger?	m_2	< 1.2
Time to Collision	Time To collision computed by project in ego and non-ego tracks and taking the minimum of the distances, if a collision occurs.	What is the TTC in open-loop vs closed-loop evaluation with Joint prediction and planning?	s	> 5*s*

Table 11: Closed-Loop evaluation metrics for Joint Prediction and Planning module

The metrics final values will be reported in EVENTS D6.2 deliverable.

2.6 EXP8

EXP8's title is "*Emergency evasion manoeuvre on the slippery roads under rain conditions*". The objective is to perform collision avoidance (e.g. pedestrian, cyclist or vehicle) in poor weather conditions on slippery roads. Experiment 8 (EXP8) is under the responsibility of the partners TUD and PERCIV.AI.

2.6.1 Architecture

The proposed behavioural decision-making aims to solve the scenario represented in Figure 21, which consists of a vehicle going straight on the road with an obstacle appearing in front of it. The decision-making task is to decide the safest course trajectory to avoid the collision. There are two options: braking and turning to perform an evasive manoeuvre or straight-line braking to stop the vehicle before the obstacle. It is essential to highlight that the described scenario represents a vehicle which needs to perform an emergency manoeuvre on a slippery surface, so it aims to evaluate the behaviour of the decision-making algorithm at the limit of handling. It does not consider decisions that are aligned with human expectations.

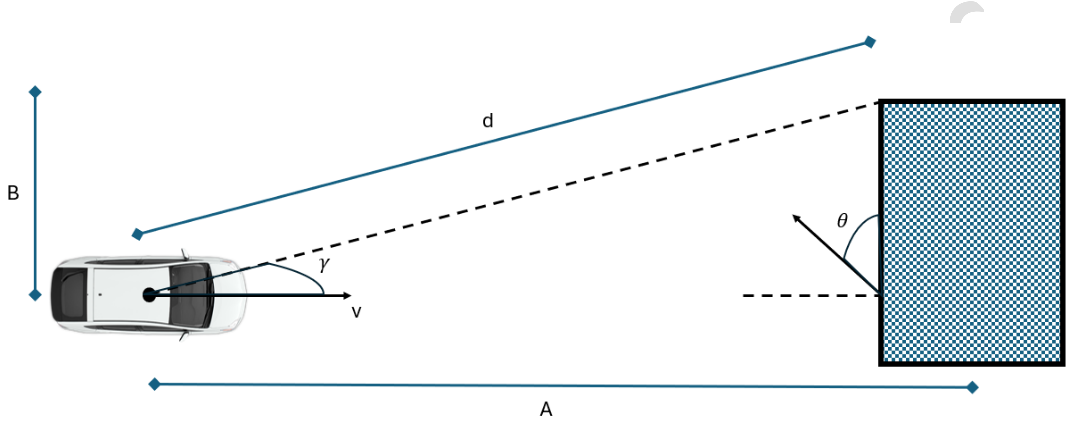

Figure 21: Scenario for behavioural decision making

The parameters are: *A* is the longitudinal vehicle-to-obstacle distance, *B* is the lateral vehicle-to-obstacle distance, γ is the angle between the vehicle velocity direction and the corner of the obstacle, *d* is the distance between the vehicle and the obstacle's corner, and ϑ is the direction of the vehicle acceleration.

The proposed decision-making consists of computing an analytical solution based on a point mass model constrained by the acceleration circle. The latter, commonly called the g-g diagram, is represented in Figure 22 and is employed to assess the performance of road vehicles [48]. The model assumes that during an emergency manoeuvre, the vehicle motion is mainly determined by the constraints on the acceleration due to friction. Thus, it is assumed that the vehicle's maximum acceleration is only limited by the friction coefficient μ and the gravitational acceleration g as follows:

$$a_x^2 + a_y^2 \leq (\mu g)^2 \tag{1}$$

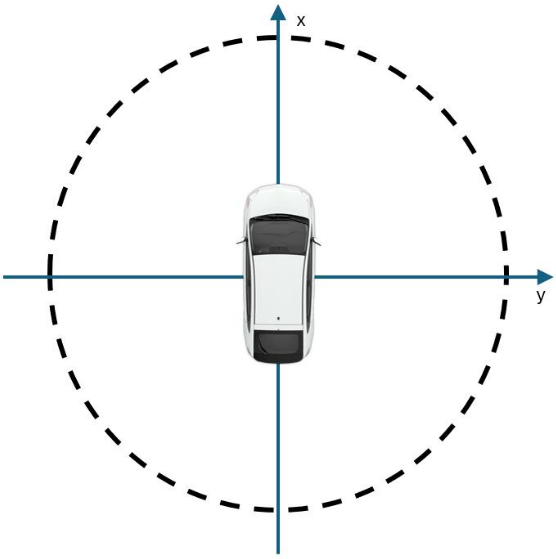

Figure 22 G-G diagram of a road vehicle

In the EVENTS project, the objective is to perform collision avoidance in poor weather conditions on slippery roads. For this reason, the decision-making algorithm is optimised for the minimum possible friction [50, 51].

2.6.2 Algorithmic Approach

Straight-line braking. The most straightforward decision is to stop before encountering the obstacle [49]. The minimum coefficient of friction necessary to halt within the braking distance *A* when travelling at velocity *v* is given by:

$$\mu = \frac{v^2}{2\,g\,A} \tag{16}$$

However, to compare its performance relative to other manoeuvres, a dimensionless, invariant quantity associated with straight-line braking is used, and it is computed as follows:

$$2\mu g A = v^2 \tag{17}$$

Minimum-Time Lane Change. Another straightforward approach involves travelling the distance B (see Figure 21) in the minimum time possible [52], [53]. This means that the vehicle does not need to brake straight, but it will avoid obstacles passing next to it. Given the point mass dynamics, the analytical solution can be computed from the vehicle's initial and final position. The initial conditions at time t are reported on the left side, and the final conditions at time t_f on the right:

$$\begin{array}{ll} X(t) = v\,t & X(t_f) = A = d\cos(\gamma) \\ Y(t) = \frac{1}{2}\mu\,g\,t^2 & Y(t_f) = B = d\sin(\gamma) \end{array} \tag{18}$$

The analytical solution formulated to simplify the comparison with the straight-line braking is represented as follows:

$$\frac{\mu\, g\, A}{v^2} = 4\tan\left(\frac{A}{B}\right) = 4\tan\gamma \tag{19}$$

Thus, it is possible to conclude that whenever $4\tan\gamma < 1$ or $|\gamma| < 14°$, the Minimum-Time Lane Change is a better strategy than straight-line braking to avoid a potential collision. However, it is important to mention that the vehicle will avoid the obstacle only when $\frac{\mu\, g\, A}{v^2} \leq 4\tan\gamma$.

Constant Curvature Turn. Another decision to the problem consists in avoiding the obstacle performing a constant radius corner. The analytical solution can be computed considering that the minimum curvature radius R is computed as $R = \frac{v^2}{\mu g}$, and the curvature radius to avoid the obstacle is $R^2 = A^2 + (B - R)^2$. Thus, the analytical solution to collision avoidance trajectory becomes:

$$\frac{\mu\, g\, A}{v^2} = \frac{2\mathrm{A}}{\mathrm{R}} = 2\sin(2\,\gamma) \tag{20}$$

The decision of constant curvature turn performs better than straight-line braking whenever $\gamma \leq 15°$.

Optimal Manoeuvre. The optimal solution for vehicle collision avoidance decisions can be expressed by computing the globally fixed acceleration vector with maximum magnitude [54],[55]. Thus, the optimal manoeuvre corresponds to computing the constant acceleration θ, see **Error! Reference source not found.**. Similarly to the M inimum-Time Lane Change, the solution can be computed by imposing initial and final conditions on the optimum problem described by the point mass model. The initial and final conditions are defined as follows:

$$\begin{aligned} X(t) &= v\, t - \sin\theta\, \frac{\mu g t^2}{2} \qquad & X(t_f) &= d\cos\gamma \\ Y(t) &= \cos\theta\, \frac{\mu g t^2}{2} \qquad & Y(t_f) &= \mathrm{d}\sin\gamma \end{aligned} \tag{21}$$

The solution can be summarised as:

$$\frac{2\mu g A}{v^2} = \frac{4\sin\gamma\cos\gamma\, cos\theta}{\cos^2(\theta - \gamma)} \tag{22}$$

The optimal constant acceleration θ can be computed, taking the first and second derivative of the Eq. 22. The optimal solution for all $\gamma \leq \arcsin {}^1/_3$ is:

$$\theta = \frac{\gamma + \arcsin(3 \sin \gamma)}{2} \quad (23)$$

However, it is also relevant to compute the trajectories with a constant acceleration which do not bring the vehicle to a collision when the minimum θ, in Eq. 23, is not defined. This is possible imposing the final longitudinal velocity of the vehicle higher or equal to 0 as follows:

$$v - \mu g \sin(\theta)\, t_f \geq 0 \quad (24)$$

By solving the system of equations in Eq. 7 for t_f and substituting it into Eq. 24, the following condition can be computed:

$$\frac{2\mu g A}{v^2} \leq \frac{\cos\gamma \cos\theta}{\sin\gamma \sin^2\theta} \quad (25)$$

The left side of Eq. 25 can then be substituted with Eq. 24, yielding the following analytical condition:

$$\theta \leq \arctan\left(\frac{1}{\tan\gamma}\right) \quad (26)$$

Figure 23 shows in red the boundary of the feasibility condition for the existing of an optimal manoeuvre with a constant acceleration θ. The minimum optimal constant acceleration θ is plotted in blue and in black are reported all the possible solutions reported in Eq. 22. Whenever the trajectories collide with the obstacles, they are reported as black dashed lines.

Analysing Eq. 20 and Eq. 15, it is possible to evaluate that the optimal manoeuvre can be performed at less friction than straight-line braking, when $\gamma < 16.7°$.

The black solid curves represent the constant acceleration trajectories that avoid the obstacle at different times. The black dashed curves are the constant acceleration trajectories that do not avoid the obstacle at different times. The solid blue line is the minimum optimal constant acceleration trajectory. The red line shows the boundary of the feasibility condition for the existence of an optimal manoeuvre with a constant acceleration.

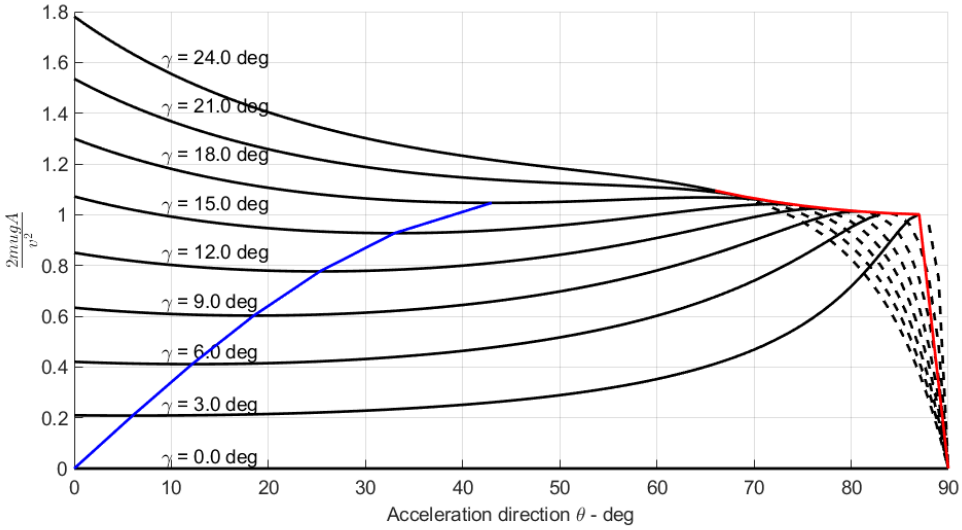

Figure 23: Performance comparison of the different strategies

The developed behavioral decision-making algorithm can provide a coarse trajectory for a vehicle that needs to avoid an obstacle on a slippery surface. The optimal coarse trajectory is optimized to work with the minimum friction coefficient to deal with poor weather conditions.

2.6.3 Future work

Future work includes quantitative evaluation of the proposed behavioural decision-making via simulations and real-world tests. Also, several limitations will be addressed: i) The vehicle dynamics are described by a point mass model (despite being highly computationally efficient, it is less accurate than the more complex model considering the tire slip); ii) The perception uncertainties are only considered when defining the obstacle size the vehicle needs to avoid (they could be further integrated into the optimization problem to reduce the conservativeness of the approach).

3. Conclusions

This deliverable highlights the behavioural decision-making strategies, including the identification of the optimal manoeuvre in a given situation, as implemented in T4.2. In particular, this task considers the inputs provided by WP3 and feeds T4.1 with high-level decisions (such as the start of lane-change and following a leading vehicle) or with low-level information (such as road boundaries and distance to stop).

The algorithms used in Task 4.2 are based on SOTA machine learning, probabilistic and optimal control methods (e.g., MDP or POMDP, GNN, MPC, Bezier Curves, etc.). In addition, T4.2 aimed also at providing high-level decision-making strategies, to identify the optimal manoeuvre in a given situation (including the cases in which an ODD limitation is reached).

This document also considers another important aspect: behaviour and trajectory prediction can be applied to describe the future behaviour of non-ego vehicles in the proximity of the ego vehicle, which can affect its decision-making and planning. So, this deliverable also presents how inference is conducted for all surrounding non-ego vehicles detected by the on-board perception system in the frontal view of the ego and targets a prediction horizon of ≥3s from the real timestamp of the prediction. This is in line with the interactive nature of prediction and planning in real-life driving.

To sum up, deliverable D4.3 presents an overview of the main algorithms used by EVENTS partners for the decision-making module, with many details of the related implementation. A complete test report will be instead provided within WP6, where the full evaluation of the decision-making modules will be performed for each experiment.

References

[1] H. Guo et al. "Simultaneous trajectory planning and tracking using an MPC method for cyber-physical systems: A case study of obstacle avoidance for an intelligent vehicle". *IEEE Transactions on Industrial Informatics*, vol. 14(9), pp. 4273-4283, 2018.

[2] Z. Zuo et al. "MPC-Based Cooperative Control Strategy of Path Planning and Trajectory Tracking for Intelligent Vehicles". *IEEE Transactions on Intelligent Vehicles*, vol. 6(3), pp. 513-522, 2020.

[3] C. Liu et al. "Path planning for autonomous vehicles using model predictive control". Proc. of IEEE Symposium on Intelligent Vehicle, 2017.

[4] M. Al-Sharman et al. "Autonomous Driving at Unsignalized Intersections: A Review of Decision-Making Challenges and Reinforcement Learning-Based Solutions". arXiv:2409.13144, 2024.

[5] L. García Cuenca et al. "Autonomous Driving in Roundabout Maneuvers Using Reinforcement Learning with Q-Learning". *MDPI Electronics*, vol. 8(12), 2019.

[6] W. Wang et al. "Imitation learning based decision-making for autonomous vehicle control at traffic roundabouts". *Multimedia tools and applications*, vol. 81, pp. 39873-39889, 2022.

[7] M. Al-Sharman et al. "Self-Learned Autonomous Driving at Unsignalized Intersections: A Hierarchical Reinforced Learning Approach for Feasible Decision-Making". *IEEE Transactions on Intelligent Transportation Systems*, vol. 24(11), pp. 12345-12356, 2023.

[8] L. Meng et al. "A reinforcement learning-based vehicle platoon control strategy for reducing energy consumption in traffic oscillations". *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, pp. 5309–5322, 2021.

[9] G Yuqing et al. "A survey of trajectory planning methods for autonomous driving—Part I: Unstructured scenarios". *IEEE Transactions on Intelligent Vehicles*, Early Access, pp. 1–29, 2023.

[10] Z. Yuxiang et al. "Adaptive decision-making for automated vehicles under roundabout scenarios using optimization embedded reinforcement learning". *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, pp. 5526–5538, 2020.

[11] X: Lu et al. "An optimized trajectory planner and motion controller framework for autonomous driving in unstructured environments". *MDPI Sensors*, vol. 21, pp. 4409, 2021.

[12] C. Feichtenhofer, H. Fan, J. Malik, K. He. "SlowFast Networks for Video Recognition". arXiv:1812.03982.

[13] L. García Cuenca et al. "Autonomous driving in roundabout maneuvers using reinforcement learning with Q-learning". *MDPI Electronics*, vol. 8, 2019.

[14] T. Liu et al. "Autonomous platoon control with integrated deep reinforcement learning and dynamic programming". *IEEE Internet of Things Journal*, vol. 10, pp. 5476–5489, 2022.

[15] W. Wang et al. “Imitation learning based decision-making for autonomous vehicle control at traffic roundabouts”. Springer Multimedia Tools and Applications, vol. 81, pp. 39873–39889, 2022.

[16] Z. Farkas et al. “MPC control strategy for autonomous vehicles driving in roundabouts”. Proc. of 30th Mediterranean Conference on Control and Automation, pp. 939–944, 2022.

[17] Z. Zuo et al. “MPC-based cooperative control strategy of path planning and trajectory tracking for intelligent vehicles”. *IEEE Transactions on Intelligent Vehicles*, vol. 6, pp. 513–522, 2020.

[18] C. Liu et al. “Path planning for autonomous vehicles using model predictive control”. Proc. of IEEE Intelligent Vehicles Symposium. pp. 174–179, 2017.

[19] EVENTS Deliverable D2.1: User and system requirements for selected use cases, 2023.

[20] EVENTS Deliverable D2.2: Full Stack Architecture & Interfaces, 2023.

[21] A.G. Cunningham et al. "MPDM: Multipolicy decision-making in dynamic, uncertain environments for autonomous driving." Proc. of IEEE International Conference on Robotics and Automation, 2015.

[22] C. Hubmann et al. "A pomdp manoeuvre planner for occlusions in urban scenarios." Proc. of IEEE Intelligent Vehicles Symposium, 2019.

[23] M. Treiber, A. Hennecke, and D. Helbing. "Congested traffic states in empirical observations and microscopic simulations." *Physical Review E*, Vol. 62(2), pp. 1805, 2000.

[24] P.G. Gipps. "A behavioural car-following model for computer simulation." *Transportation research part B: methodological*, vol. 15(2), pp. 105-111, 1981.

[25] A. Kesting, M. Treiber, and D. Helbing. "General lane-changing model MOBIL for car-following models." *Transportation Research Record*, pp. 86-94, 2007.

[26] L.P. Kaelbling, M.L. Littman, and A.R. Cassandra. "Planning and acting in partially observable stochastic domains." *Artificial intelligence*, vol. 101(1-2), pp. 99-134, 1998.

[27] D.S. González et al. "Human-like decision-making for automated driving in highways." Proc. of IEEE Intelligent Transportation Systems Conference, 2019.

[28] D. Silver, and J. Veness. "Monte-Carlo planning in large POMDPs." Proc. of Advances in neural information processing systems, vol. 23, 2010.

[29] D. Lenz, T. Kessler, and A. Knoll. "Tactical cooperative planning for autonomous highway driving using Monte-Carlo Tree Search." Proc. of IEEE Intelligent Vehicles Symposium, 2016.

[30] D-K. Kye, S.W. Kim, and S-W. Seo. "Decision making for automated driving at unsignalized intersection." Proc. of 15th International Conference on Control, Automation and Systems, 2015.

[31] P. Dagum, A. Galper, and E. Horvitz. "Dynamic network models for forecasting." Uncertainty in artificial intelligence. Morgan Kaufmann, 1992.

[32] L. Zhang et al. "Efficient uncertainty-aware decision-making for automated driving using guided branching." Proc. of IEEE International Conference on Robotics and Automation, 2020.

[33] Z. Zhang, and J.F. Fisac. "Safe occlusion-aware autonomous driving via game-theoretic active perception." arXiv preprint arXiv:2105.08169, 2021.

[34] S. Kammel et al. "Team AnnieWAY's autonomous system for the 2007 DARPA Urban Challenge." *Journal of Field Robotics*, vol. 25(9), pp. 615-639, 2008.

[35] S. Noh, and K. An. "Decision-making framework for automated driving in highway environments." *IEEE Transactions on Intelligent Transportation Systems*, vol. 19(1), pp. 58-71, 2017.

[36] J. Ziegler et al. "Making bertha drive—an autonomous journey on a historic route." *IEEE Intelligent Transportation Systems Magazine*, vol. 6(2), pp. 8-20, 2014.

[37] C. Hubmann, M. Aeberhard, and C. Stiller. "A generic driving strategy for urban environments." Proc. of 19th International Conference on Intelligent Transportation Systems, 2016.

[38] P. Hart, N. Nilsson, and B. Raphael. “A Formal Basis for the Heuristic Determination of Minimum Cost Paths”. *IEEE Transactions on Systems Science and Cybernetics*, vol. 4(2), pp. 100–107, 1968.

[39] M. Graf, O. Speidel, and K. Dietmayer. "A model based motion planning framework for automated vehicles in structured environments." Proc. of IEEE Intelligent Vehicles Symposium, 2019.

[39] X. Ma et al. "Legal Decision-making for Highway Automated Driving." *IEEE Transactions on Intelligent Vehicles*, vol. 9(8), pp. 5284–5298, 2023.

[40] C.E. Garcia, D. M. Prett, and M. Morari. "Model predictive control: Theory and practice—A survey." *Automatica*, vol. 25(3), pp. 335-348, 1989.

[41] J. Liang and R. Urtasun, “End-to-End Deep Structured Models for Drawing Crosswalks”. Proc. of European Conference on Computer Vision, 2018.

[42] D. Attard et al. “Reinforcement Learning for Autonomous Control of Articulated Vehicles in Roundabout Intersections”. Proc. of 10th International Conference on Control, Decision and Information Technologies, pp. 450–454, 2024.

[43] J. Gao et al. “VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation”. 10.48550/arXiv, 2020.

[44] R Izquierdo et al. “Experimental validation of lane-change intention prediction methodologies based on CNN and LSTM”. Proc. of IEEE Intelligent Transportation Systems Conference, 2019.

[45] S. Hagedorn et al. “Rethinking the Integration of Prediction and Planning in Deep LearningBased Automated Driving Systems: A Review”. arXiv:2308.05731v2, 2024.

[46] C. Gariboldi, M. Corno and B. Jin. “Hybrid Imitation-Learning Motion Planner for Urban Driving”. arXiv:2409.02871, 2024.

[47] M. Bansal, A. Krizhevsky, and A. Ogale. “ChauffeurNet: Learning to drive by imitating the best and synthesizing the worst”. arXiv:1812.03079, 2018.

[48] M. Massaro, S. Lovato, and M. Veneri. “An optimal control approach to the computation of g-g diagrams”. *International Journal of Vehicle Mechanics and Mobility*, vol. 62(2), pp. 448-462, 2024.

[49] S. C. Peters. “Optimal planning and control for hazard avoidance of front-wheel steered ground vehicles”. Massachusetts Institute of Technology, Boston, 2012.

[50] V. Fors, B. Olofsson, and L. Nielsen. “Autonomous Wary Collision Avoidance”. *IEEE Transactions on Intelligent Vehicles*, vol. 6(2), pp. 353-365, 2021.

[51] Y. Hattori, E. Ono, and S. Hosoe. “Optimum vehicle trajectory control for obstacle avoidance problem”. *IEEE/ASME Transactions on Mechatronics*, vol. 11(5), pp. 507-512, 2006.

[52] V. Fors et al. “Real-time minimum-time lane change using the modified Hamiltonian algorithm”. Proc. of Advances in Dynamics of Vehicles on Roads and Tracks, Gothenburg, Sweden, 2019.

[53] Y. Gao, M. Lidberg, and T. Gordon. “Modified Hamiltonian algorithm for optimal lane change with application to collision avoidance”. *MM Science Journal*, pp. 576-584, 2015.

[54] Y. Hattori, E. Ono, and S. Hosoe. “An optimum vehicle trajectory control for obstacle avoidance with the shortest longitudinal traveling distance”. Proc. of IEEE International Conference on Mechatronics and Automation, 2008.

[55] V. Fors, B. Olofsson, and L. Nielsen. “Attainable force volumes of optimal autonomous at-the-limit vehicle manoeuvres”. *International Journal of Vehicle Mechanics and Mobility*, vol. 58(7), pp. 1101-1120, 2020.