Designing cooperative interaction of automated vehicles with other road users in mixed traffic environments

D.3.1 Cooperation and Communication Planning Unit Concept

Work package	WP3: Cooperation and Communication Planning Unit
Task(s)	Task 3.1: Situation matching module Task 3.2: Software module for human-vehicle interaction planning, Task 3.3: Software module for execution of human-vehicle interaction Task 3.4: Safety layer for the CCP Unit
Authors	**Drakoulis, Richardos - ICCS**, Drainakis, Giorgos - ICCS, Portouli, Evangelia - ICCS, Althoff, Matthias - TUM, Magdici, Silvia - TUM, Tango, Fabio - CRF, Markowski, Robert - DLR
Dissemination level	Public (PU)
Status	Final
Due date	30/04/2018
Document date	23/05/2018
Version number	1.1
	This work is part of the interACT project. interACT has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395. Content reflects only the authors' view. The Innovation and

This is a draft version of deliverable D3.1 which has not been approved by the EC, yet.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Table of contents

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Index of figures

This report is part of interACT project that has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement no 723395.

Index of tables

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Glossary of terms

Term	Description
Automated vehicle (AV)	Vehicle that provides automation of longitudinal and lateral vehicle control and can free the driver from the driving task
Cooperation and Communication Planning Unit	interACT central software unit that plans AV behaviour and explicit HMI control in an integrated, timely, and synchronised manner
Fail-safe trajectory	A trajectory, which ensures that the AV is brought to a safe state in any case of a failure
Interaction	Within interACT interaction is understood as the complex process where multiple traffic participants perceive one another and react towards the continuously changing conditions of the situation resulting from actions of the other TP, to achieve a cooperative solution. These actions and reactions involve various means of communication
Lanelet	An atomic drivable road segment, which is defined by its left and right bound and may carry additional data to describe the static environment
Non-motorised TP	Pedestrians or cyclists
On-board user	Human on-board of the AV who acts as a driver in all cases the AV cannot handle (SAE level 3) or is a passenger for all SAE 4 and 5 applications
Other road users / other traffic participants	All possible road users from the perspective of the ego vehicle (the AV) i.e. pedestrians, bicyclists, motorcyclists, vehicles, automated vehicles

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

List of abbreviations and acronyms

Abbreviation	Meaning
A*	A-star algorithm
AI	Artificial Intelligence
AV	Automated Vehicle
CCPU	Cooperation and Communication Planning Unit
D*	Dynamic A*
DRL	The Drools system rule language
GNSS	Global Navigation Satellite System
HMI	Human Machine Interface
LCM	Lightweight Communications and Marshalling
LHS	Left Hand Side
LiDAR	Light detection and ranging sensor. Used for the detection of distances and velocities of objects
MDP	Markov Decision Process
ML	Machine Learning
MPC	Model Predictive Control
NGSIM	Federal Highway Administration’s Next Generation Simulation
OWL	Web Ontology Language
POMDP	Partially-observable Markov Decision Process
PRM	Probabilistic Roadmap Method
RHS	Right Hand Side
ROS	Robot Operating System
RRT	Rapidly-exploring Random Tree

This report is part of interACT project that has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement no 723395.

RRT*	An asymptotically optimal adaptation of the RRT algorithm
SPOT	Set-Based Prediction Of Traffic Participants
SWRL	Semantic Web Rule Language
TP	Traffic participant
TTR	Time-To-React
UDP	User Datagram Protocol
W3C	World Wide Web Consortium
WP	Work Package

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Executive Summary

As Automated Vehicles (AVs) will be deployed in mixed traffic, they need to interact safely and efficiently with other traffic participants (TPs). The interACT project is working towards the safe integration of AVs into mixed traffic environments.

In its Work Package (WP) 3, the interACT project aims to develop a novel Cooperation and Communication Planning Unit (CCPU) to enable the integrated planning and control of AV's behaviour, and the provision of time-synchronised Human Machine Interfaces (HMI) for both the on-board user and the other TPs.

This document is the first deliverable of WP3 and presents the concept of the CCPU. Based on the interACT system architecture, each CCPU component, namely the *Situation Matching*, *Interaction Planning*, *Trajectory Planning* and *Safety Layer*, is described. Each description is followed by a detailed analysis of the current status and implementation plan.

Apart from CCPU components, a detailed presentation of the *Scenarios* and *Interactions Strategies* digital catalogues (components of the Enablers functional block) is given, since they are not only part of WP3, but also essential for the realisation of the CCPU.

Finally, the document focuses on the technical collaboration within WP3, to enable components development and integration. The interfaces from the interACT system architecture are further defined and documented and Robot Operating System (ROS) has been selected as the common software framework.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

1. Introduction

1.1 Purpose and scope

The interACT project will develop solutions on how to safely integrate Automated Vehicles (AVs) into mixed traffic environments. The CCPU is the central core to the interACT AV, that enables the AV to interact with its on-board user and the other traffic participants (TPs) in an expectation-conforming and intuitive way.

The CCPU's objective is to plan how the encountered traffic scenario should evolve in the future from the AV perspective. The planning considers the motion of both the AV and all the other involved TPs. Taking into account the gestures, the anticipated intentions and the predicted behaviour of the other TPs, the CCPU will develop an expectation-conforming, safe plan for the future motion of the AV and its interactions with the other TPs and its on-board user. This plan will be used as an input for the actual, time-synchronised planning of the AV motion and the sequential control of the explicit HMI elements.

Towards this purpose, the CCPU has been split into four components [1], namely *Situation Matching*, *Interaction Planning*, *Trajectory Planning* and *Safety Layer*. Their functionality is supported by two data enablers: the *Scenarios* and the *Interactions Strategies* digital catalogues. Each component concept, current status and implementation plan is presented in detail in a separate section in the document at hand, followed by the description of the work on the technical collaboration within the WP.

1.2 Intended readership

This documents focuses on four different readerships. At first, it addresses mainly the interACT WP3 partners, since it presents the current status of the CCPU component and it serves as a basis for its further development, integration and finalisation. Secondly, the content of this document can also influence the technical work within the other interACT WPs, therefore it also addresses all other interACT partners who are involved in development or integration tasks. Thirdly, it gives the European Commission Project Officer of the interACT project an overview of the work conducted in the WP. Fourthly, since this is a public document, it is expected to serve as a useful reference to all interested researchers in academia and automotive industry.

1.3 Relationship with other interACT deliverables

This deliverable is part of the work in WP3: "Cooperation and Communication Planning Unit". It is directly related to the deliverables of WP1: D1.1 "Definition of interACT use cases and scenarios" (the scenarios digital catalogue is based on D1.1 use cases) and D1.2 "Requirements, system architecture and interfaces for software modules" (the CCPU concept is presented though its four main components defined in D1.2).

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

As the first deliverable of WP3 and also the first with a technical content (it proposes detailed interfaces and software framework), it will directly influence the development in all subsequent technical tasks within the project.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

2. Methodology

2.1 Introduction

The development of the CCPU in WP3 is (by definition) an elaborate task, as it handles the central intelligence of the system. It is the key module, through which the interactions of all the actors (AV, on-board user and other traffic participants) are planned and executed in an integrated and time-synchronized manner. In general, the CCPU interrelates with other interACT WPs, using as input the defined scenarios (WP1), the outcomes of the psychological modelling and observations (WP2), the human-vehicle interaction elements (WP4) and is the basis for the integration (WP5).

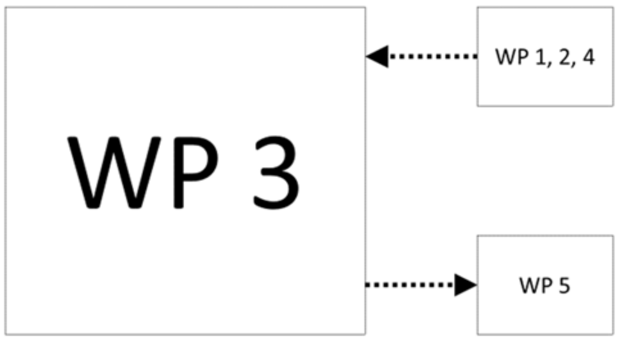


Figure 1: Orchestration with other project WPs.

To achieve this and at the same time ensure safety in a critical environment, where traffic interactions occur, the following tasks are defined, as necessary components for the implementation of the CCPU.

- **Situation matching task**, which is responsible for matching the actual traffic situation that occurs in real-time with one of the predefined scenarios. The scenarios are modelled into a digital catalogue and are selected during runtime, according to the environment information (perception). The interacting actors' behaviour will be continuously monitored and the situation matching will be re-evaluated according to the feedback.

- **Human-vehicle interaction planning task**, which will plan the strategy to be followed selecting from a predefined list and at the same time will be ready to adapt it, depending on the continuously changing traffic environment. Again, a digital catalogue will contain all possible action plans. Then, in accordance to the Situation Matching outcome, an action plan will be strategized, which will include the planning of the explicit communication via HMI elements. Similarly, to the Situation Matching task, this process will occur dynamically, always in relation to the evolution of the environment.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

- **Execution of human-vehicle interaction task**, which will execute the planned actions of the vehicle and also manage control of the HMI elements. Also, it will be in charge of monitoring the status of HMI elements, along with the status of the running action plan.

- **Safety layer task**, which will ensure safety, by providing a set of actions in case of emergency. It will verify that the outcome of the action plan does not conflict with the rest of the TPs and will predefine a fail-safe plan to be executed, if a conflict occurs.

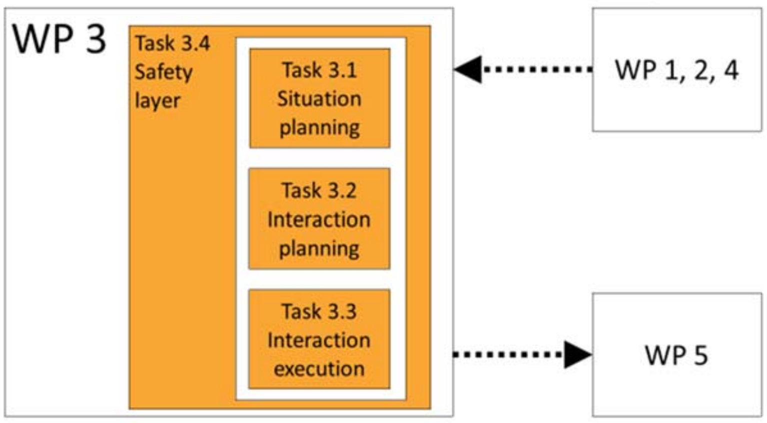

WP 3
Task 3.4 Safety layer
Task 3.1 Situation planning
Task 3.2 Interaction planning
Task 3.3 Interaction execution
WP 1, 2, 4
WP 5

Figure 2: CCPU (WP3) tasks definition.

The overall architecture of the CCPU, as defined in D1.2 [1], including the four central CCPU modules and the two enablers is depicted in Figure 3. It is noted that the generic term "Catalogue" will be used for the two enablers, instead of the more restrictive term "Ontology".

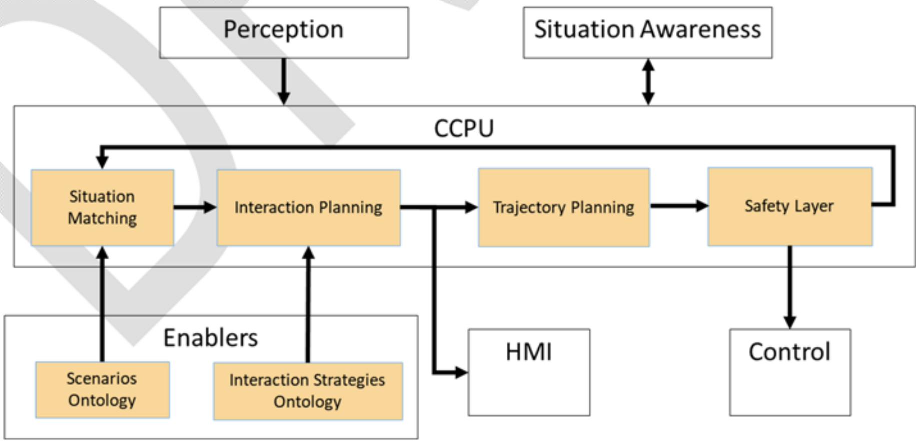


Figure 3: CCPU high-level architecture.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

2.2 Task 3.1 - Situation Matching

The implementation of this task's procedures is performed mainly by the Situation Matching module in collaboration with the Scenarios digital Catalogue. The latter is a static enabler, which formally and conceptually describes the interaction scenarios that can be encountered by the AV.

The Situation Matching module receives information regarding the external environment from the Perception component and information regarding other TPs' behaviour from the Situation Awareness component. It then matches the current traffic situation with one or more scenarios from the catalogue and marks the TPs as interacting actors, as required for the Perception module monitoring.

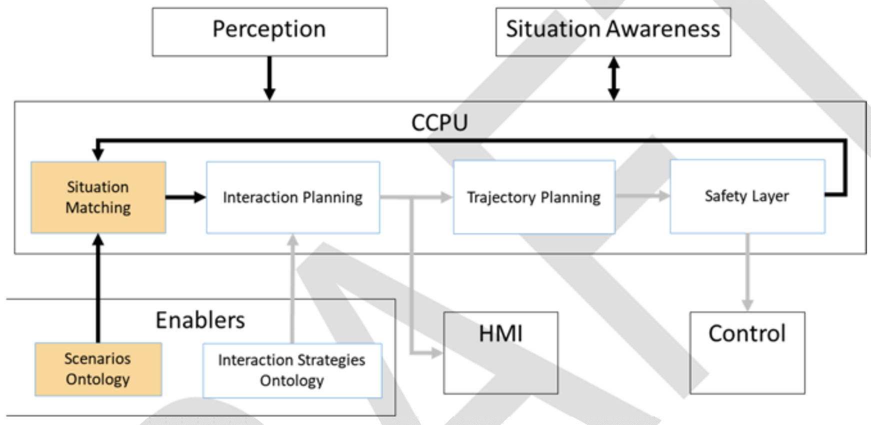


Figure 4: Situation Matching (Task 3.1) basic architecture.

2.3 Task 3.2 - Human-vehicle interaction planning

The functionalities envisaged in this task are covered by the Interaction Planning module, together with the Interaction Strategies digital Catalogue and by the Trajectory Planning module.

The Interaction Strategies Catalogue is a digital catalogue, similar to the Scenarios Catalogue in Task 3.1, which provides interaction plans, which are appropriate for the matched situation. A plan is a sequence of AV actions and HMI tasks (communication), connected by time restrictions. The Interaction Planning module works on top of the catalogue, as it chooses a plan, in case there are multiple, and also specifies it, according to sensor data. In the end, the Trajectory Planning module is responsible to receive the plan and extract the vehicle-related details, as to how it will be actualized.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

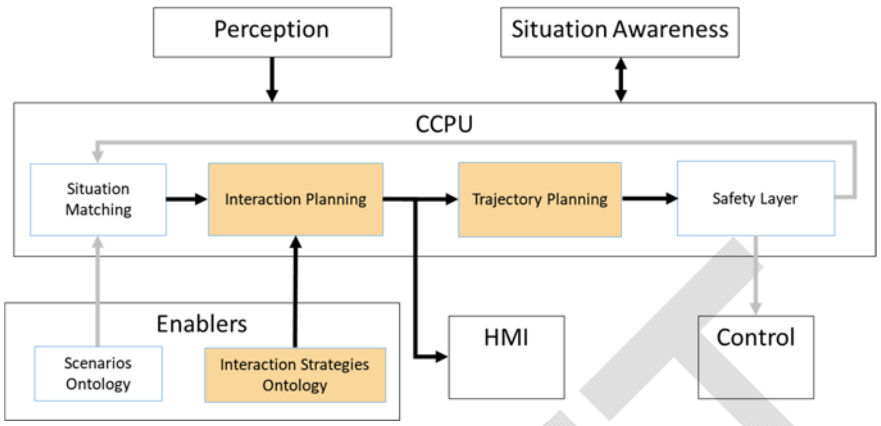


Figure 5: Human-vehicle interaction planning (Task 3.2) basic architecture.

2.4 Task 3.3 - Execution of human-vehicle interaction

The functionalities envisaged in this task are spread among the following components: Situation Matching module, Interaction Planning module, Safety Layer module, HMI controllers and Vehicle controllers.

The execution of the chosen interaction plan is performed by the Safety Layer module, in terms of planned trajectory initiation and by the Interaction Planning module, in terms of HMI interaction initiation. The monitoring is done in both a software and a hardware level.

In a software level:

- The CCPU continuously perceives the environment via the Perception and Situation Awareness blocks, which interface with the Situation Matching module.
- The Situation Matching module continuously checks if the current scenario is still valid.
- The Interaction Planning module continuously validates the outcome, by keeping the state of the performed actions.

In a hardware level:

- The HMI controllers forward HMI commands to the corresponding actuators.
- Vehicle controllers forward motion commands to the corresponding actuators.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

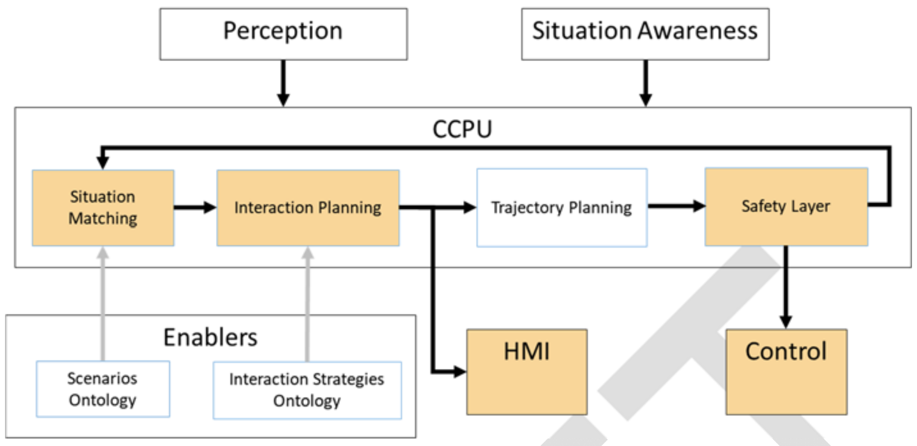


Figure 6: Execution of human-vehicle interaction (Task 3.3) basic architecture.

2.5 Task 3.4 – Safety Layer

This task is performed solely by the Safety Layer module. In case the calculated actions are above a danger threshold, a fail-safe manoeuvre is activated by the module to ensure safety of all TPs.

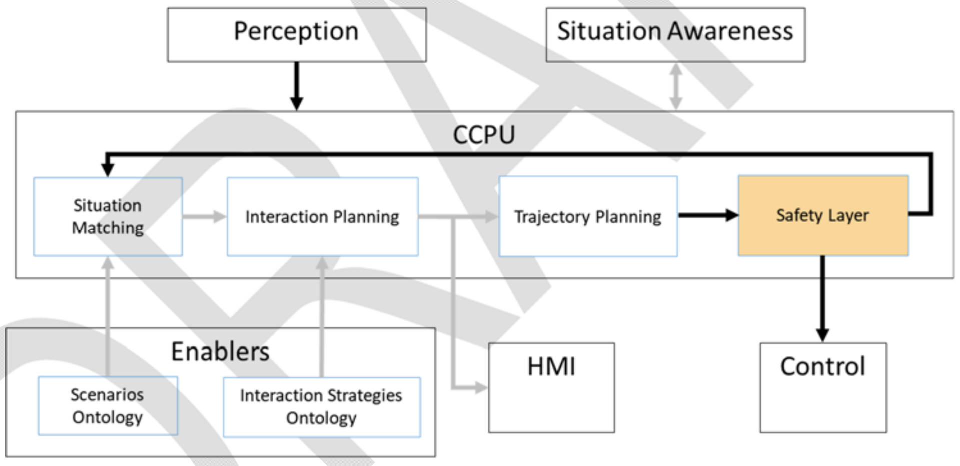


Figure 7: Safety layer (Task 3.4) basic architecture.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

3. Scenarios and Strategies Catalogues

3.1 Scenarios Catalogue

3.1.1 Description

In interACT D1.1 [2], a number of relevant use cases and scenarios, that the AV could encounter and are needed to be studied in terms of interaction among the AV, its on-board user and other TPs, were identified. The AV should be able to recognize them automatically at any time and act accordingly. For this purpose, the scenarios need to be encoded in a way that allows them to be automatically accessed and understood by the system. The behaviour of the triggering point of CCPU, i.e. the Situation Matching component (presented in section 4), depends on the scenarios digital representation, which is the main responsibility of the Scenarios Catalogue.

Each scenario should be described so that it is fully recognizable by the AV. The main features that define a scenario are: (i) the planned trajectory of the AV, (ii) the road characteristics and topology, (iii) the type and number of other TPs, (iv) the intentions of other TPs. For each different combination of the above mentioned attributes - which are not the only ones that have to be represented - a different interaction scenario arises. It is obvious that a very large amount of data is needed to represent all possible cases, even if one limits to the use cases defined in D1.1. In order to solve this problem and to generalize the solution, it has been chosen to digitalise the scenarios as a knowledge representation that should consist of:

- a vocabulary that defines the domain of knowledge
- a set of rules that will use the vocabulary concepts to extend the knowledge, using a reasoning mechanism

Reasoning provides an exponential or more compression in the knowledge one needs to store. Without it, the amount of information one would have to store would be infeasible.

The scenarios vocabulary should define the concepts and relationships used to describe and represent our domain of knowledge. Although, in general, a vocabulary can be a simple data model, in our case this representation is not sufficient. A data model is specifically related to data only and does not provide machine-interpretable definitions in a specific domain.

On the contrary, what is necessary for the scenarios description, in order to be able to reason and produce new knowledge, is a formal definition of terms in a hierarchical taxonomy with attributes and relations. In particular, the vocabulary that should be created comprises descriptions for the following kinds of concepts:

- Classes (the general things)

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

- Individuals (objects instances)
- Properties that things may have
- Relations between classes and individuals

Therefore, according to the World Wide Web Consortium (W3C) definition [3], we are implementing an "Ontology".

The ontology can be extended with rules, which fulfils our second requirement for our knowledge representation. The existence of rules is necessary not only for extending the domain with new knowledge (e.g. by defining the class hierarchy), but also for the scenario matching, as described in the following subsection (from the design perspective) as well as in section 4 (from the implementation perspective).

3.1.2 Design and Implementation plan

The Scenarios Catalogue is responsible to provide the design of the scenarios ontology, which, as already mentioned, consists of:

- Hierarchical taxonomy of classes
- Properties and relations
- Rules

On the other hand, the development of the ontology is part of the Situation Matching component, therefore the languages and tools that will be used for its realisation will be discussed in the corresponding section (section 4). Consequently, the ontology and rule definitions that will be provided to the Situation Matching module should be independent of the tools and languages used.

The domain of knowledge that should be represented, mainly consists of road and traffic related elements, TPs and their relations. Preliminary taxonomies of classes and properties are illustrated in Figure 8 and Table 1 respectively. Additionally, Figure 9 depicts a subset of classes and relations in an intersection for clarification of the presented concepts. These will work as a basis for the development of the final ontology and will be refined, as soon as the observational studies results become available.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

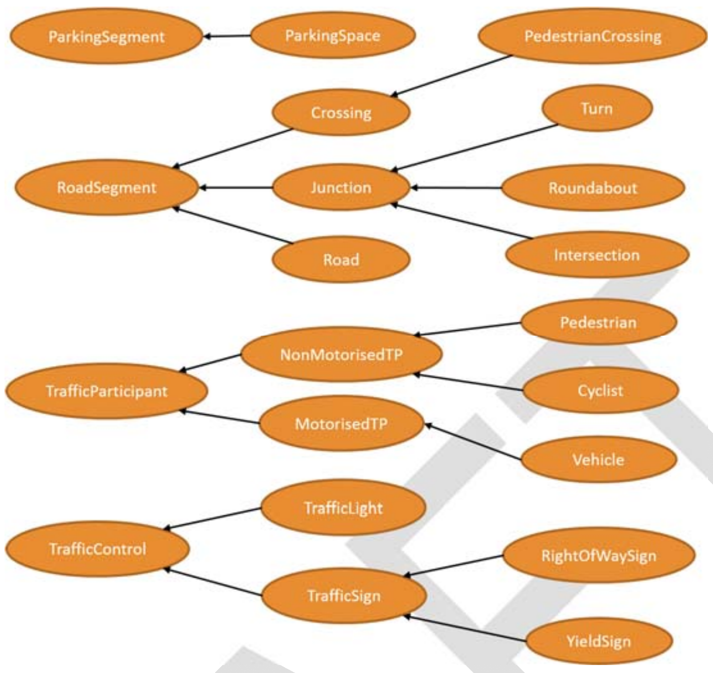


Figure 8: Preliminary ontology taxonomy.

Table 1: Preliminary ontology properties

Property	**Description**
approaches	Relates a TrafficParticipant with a RoadSegment (TrafficParticipant is approaching RoadSegment).
entering	Relates a TrafficParticipant with a ParkingSpace (TrafficParticipant is entering ParkingSpace).
leaving	Relates a TrafficParticipant with a ParkingSpace (TrafficParticipant is leaving ParkingSpace).
wantsToCross	Relates a TrafficParticipant with a RoadSegment (TrafficParticipant intending to cross the RoadSegment).
detected	Property of a TrafficParticipant indicating that is detected by the AV.
willTurn	Property of a TrafficParticipant indicating that is wanting to turn.
crossPaths	Relates two TrafficParticipant objects indicating that their future paths will cross.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

isOn	Relates a TrafficParticipant with a RoadSegment (TrafficParticipant is on RoadSegment).
hasOn	Inverse property of isOn.
hasSign	Relates a RoadSegment with a TrafficControl (RoadSegment has the specific TrafficControl).
isSignOf	Inverse property of hasSign.
isConnectedTo	Relates two RoadSegment objects (indicating that are directly connected).
hasPredecessor	Relates two RoadSegment objects. Subproperty of isConnectedTo.
hasSuccessor	Inverse property of hasPredecessor.
signalised	Property of a RoadSegment indicating that has at least one TrafficControl.
unsignalised	Property of a RoadSegment indicating that has no TrafficControl.
noTrafficLights	Property of a RoadSegment indicating that has no TrafficLight.

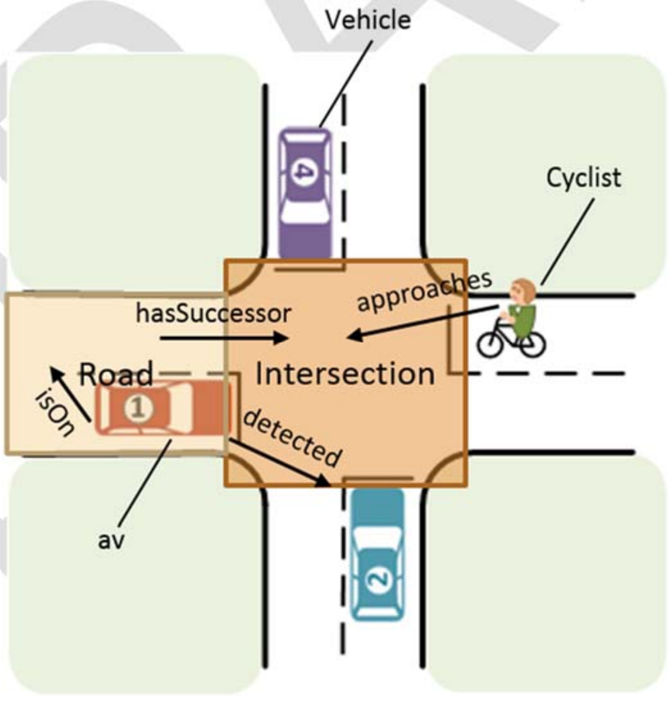


Figure 9: Subset of classes and relations in an intersection.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

The process of scenario matching is modelled using logical formulas, as a mean of a formal system to describe relations over quantified variables. Using the use-case description, as defined in [2], the scenario entities along with their relations are formulated into a rule, based on first-order logic syntax [4], as shown in Table 2. The rules, though, should be written in a less expressive representation in order to support for more efficient reasoning. It was decided to represent the rules as Horn clauses [5], since every Horn rule can be integrated in the majority of rule engines.

Table 2: First-order logic syntax

Terms & Formulas	Symbol description	Example
Variable	underscore, followed by variable name in uppercase	_PERSON
Constant	constant name in lowercase	gate
n-ary Predicate	a function of arity n	enters(_PERSON, gate)
Conjunction	∧	enters(_PERSON, gate) ∧ isMale(_PERSON)
Disjunction	∨	enters(_PERSON, gate) ∨ exits(_PERSON, gate)
Implication	→	enters(_PERSON, gate) → color(_LIGHT, green)

From the Horn rule, one can extract the literals, which in turn are broken down into specific data, acquired from the various interfaces of the Situation Matching module. As a characteristic example, the must-have use case 6.1.1 "React to crossing non-motorised TP at crossings without traffic lights", as described in [2], is presented below.

Table 3: Use case 6.1.1 description

Description
The AV approaches a non-motorised TP who wants to cross the road at a crossing without traffic lights. Main goal is to handle the situation safely by using a clear and understandable communication of the AV's intention. After that the AV should continue driving.

Then the rule is modelled as:

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

$$\left.\begin{array}{l}\text{approaches(av,\_RS)} \wedge \\ \text{crossing(\_RS)} \wedge \\ \text{noTrafficLights(\_RS)} \wedge \\ \text{detected(\_TP)} \wedge \\ \text{nonMotorised(\_TP)} \wedge \\ \text{wantsToCross(\_TP, \_RS)}\end{array}\right\} \Longrightarrow \text{encountered (scenario1)}$$

The modelling process of all must-have and optional use cases of the interACT project [2] is presented in Table 4 below. Scenario 3 rule, unlike the other scenario rules, seems to violate the requirement of a Horn clause representation. In practice, it can be split into two equivalent Horn clauses, but it was decided to present it in the table in a single clause for brevity and consistency.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Table 4: Basic Scenarios modelled as Horn rules

Scenarios				Rule
	Idx	Name	Short Description	
Must-have	1	React to crossing non-motorised TP at crossings without traffic lights	The AV approaches a non-motorised TP who wants to cross the road at a crossing without traffic lights.	approaches(av, _RS) ∧ crossing(_RS) ∧ noTrafficLights(_RS) ∧ detected(_TP) ∧ nonMotorised(_TP) ∧ wantsToCross(_TP, _RS) → encountered(scenario1)
	2	React to an ambiguous situation at an unsignalised intersection	The AV approaches an unsignalised intersection which requires interaction with another/ multiple other vehicles.	approaches(av, _RS) ∧ intersection(_RS) ∧ unsignalised(_RS) ∧ detected(_TP) ∧ vehicle(_TP) ∧ crossPaths(av, _TP) → encountered(scenario2)
	3	React to non-motorised TP at a parking space	The AV is driving into or out of a parking space and has to react to a nonmotorised TP in their path.	(entering(av, _PS) ∨ leaving(av, _PS)) ∧ parkingSpace(_PS) ∧ detected(_TP) ∧ nonMotorised(_TP) ∧ crossPaths(av, _TP) → encountered(scenario3)
	4	React to vehicles at a parking space	The AV approaches a parking space which another vehicle is leaving.	entering(av, _PS) ∧ detected(_TP) ∧ leaving(_TP, _PS) ∧ parkingSpace(_PS) ∧ vehicle(_TP) → encountered(scenario4)
Optional	5	React to vehicles on the road in turning situation (AV wants to turn)	The AV approaches an intersection and wants to turn. While turning, the AV has to obey the traffic rules and to react to other vehicles from the opposite or even the same direction (e.g. bicycle).	approaches(av, _RS) ∧ intersection(_RS) ∧ willTurn(av) ∧ detected(_TP) → encountered(scenario5)
	6	React to crossing non-motorised TP at a signalised crossing	The AV approaches a signalised crossing e.g. a pelican crossing while a nonmotorised TP intends to cross the road.	approaches(av, _RS) ∧ crossing(_RS) ∧ signalised(_RS) ∧ detected(_TP) ∧ nonMotorised(_TP) ∧ wantsToCross(_TP, _RS) → encountered(scenario6)

3.2 Interaction Strategies Catalogue

3.2.1 Description

The Interaction Strategies Catalogue represents the knowledge base for interactions of and with the AV. The corresponding knowledge will be provided by WP2 and WP4. While the results of these work packages are not yet available, a concept for this catalogue was already developed.

The Interaction Strategies Catalogue describes all the interactions that can occur between the AV and other TPs. Crucial for the interaction are the intentions of the TPs. These intentions need to be

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

communicated between the AV and the TP and negotiated if they compete. If this is not done, competing intentions may lead to dangerous situations and discomfort. Thus, intentions and resulting actions need to be modelled.

The first part of the model describes the goal of the AV. While each situation might have a clear initial goal for the AV, e.g. a goal that is dictated by traffic rules, the goal of the AV can change based on the interaction with other TPs. For example, pedestrians at a zebra crossing generally have the right of way, but they can renounce their right of way and let a motorized vehicle pass. So, while the initial goal of the motorized vehicle might be to give way, the goal changes within the interaction and the motorized vehicle does not have to wait until the pedestrian crosses the road.

The second part of the model represents the intentions of other TPs. These intentions have a big influence on the goal of the AV, because the AV, like every traffic participant, has to act in a safe manner. To stick with the example of pedestrians waiting at a zebra crossing, they might have the intention of crossing. In that case, the AV should let them cross. If they wait to let the AV pass, the AV should continue without stopping. It is the task of the AV to correctly interpret the intentions of the TPs around it.

Additionally, the AV can try to explicitly communicate its intentions and influence the behaviour of the other TPs through implicit cues and explicit communication via additional HMI elements. It is not safe to assume that the other TPs react in an expected or intended manner though. The only way for the AV to close the control loop is to observe the other TPs and correctly interpret their intentions. The goals of the AV then again completely depend on the actions of the other TPs. In the end, the AV has to act in a safe manner and cannot unnecessarily put the other TPs at risk. A more detailed look into the models is given in the design and implementation plan.

3.2.2 Design and Implementation plan

The Interaction Strategies Catalogue consists of two parts: The AV Goals and the TP Intentions. The AV Goals describe how the AV wants to reach a given goal with a set of allowed actions as well as constraints to ensure safety and comfort. The TP Intentions describe how the AV is allowed to react to a given situation. The TP Intentions have to be observed from reality and then classified to fit this model.

An AV Goal describes the general goal in a given situation. The three components of manoeuvres, the HMI output and constraints should work together to draw a coherent and consistent picture of the AV Goals for the other TPs in the near environment of the AV.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

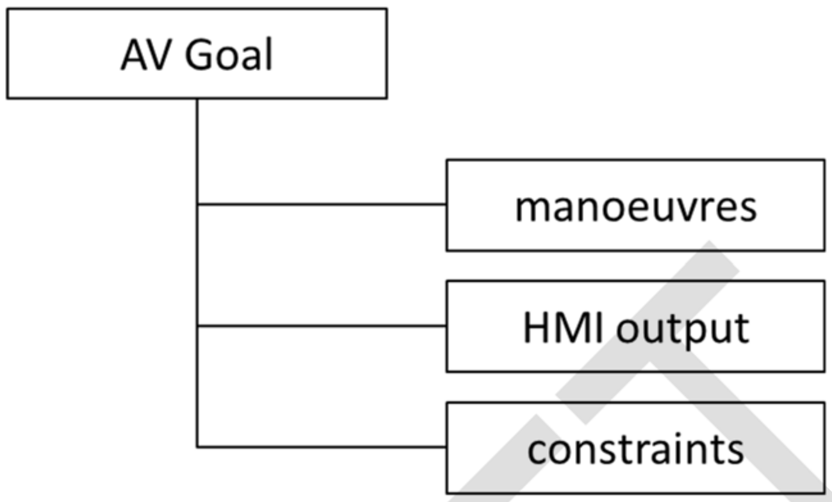


Figure 10: General structure of an AV Goal.

The manoeuvres define a set of allowed manoeuvres to achieve the given AV Goal. It is important to note that manoeuvres also represent a mean of implicit communication for the AV. Thus, the AV may not perform any manoeuvre that might see fit in similar situations, but it has to be ensured that the manoeuvre implicitly communicates the same AV Goal like the HMI output and the followed constraints.

The HMI output represents the means of explicit communication for the AV. Most vehicles differ though, thus the direct control of the low level HMI components is not an appropriate approach. Instead, an HMI controller translates semantic commands into low level control commands for the HMI, deciding based on the available means which one is appropriate in the current situation. More specific commands can be given to components that exist in all vehicles, like headlights or the horn.

Last but not least, certain constraints have to be fulfilled depending on the AV Goal. They not only represent a first safety net, but should also ensure a certain level of comfort for other TPs. Among others, possible measures include:

- A maximum approach speed when giving way to other TPs: A high approach speed signalizes the intention to pass another TP without giving way.
- Minimum distances to other TPs: Even when driving slowly, a very close vehicle can seem very threatening and may lead to discomfort.
- Minimum gap time: The time gap describes the time between one TP leaving an area and another TP entering that area. Gap times can also be used as a safety measure where small gap times indicate safety-critical situations.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

* Minimum time to collision: The time to collision describes the time before two TPs would collide if they keep their current speeds and directions. Low times to collision also indicate safety-critical situations.

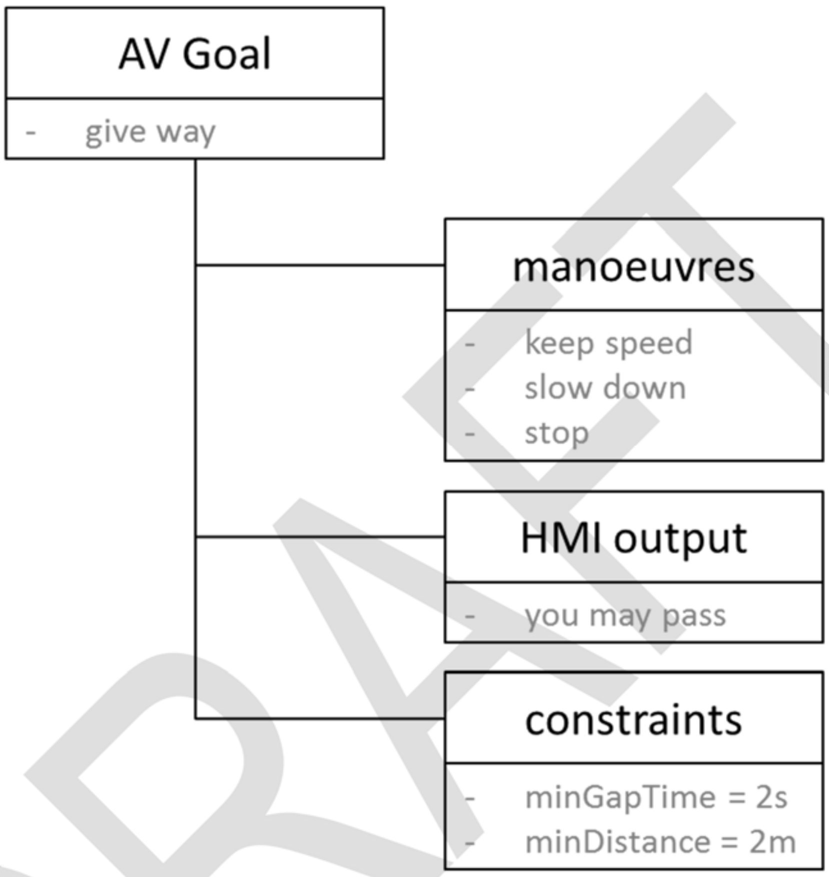


Figure 11: Specification of AV Goal 'give way'.

An example for a specification of an AV Goal is given in Figure 11. Here goal of the AV is to 'give way', e.g. to a pedestrian at a zebra crossing. The allowed manoeuvres are 'keep speed' (driving on with the current speed), 'slow down' (breaking to a new, lower maximum speed) and 'stop' (stopping, e.g. in front of the zebra crossing). The semantic HMI output 'you may pass' should signal to the pedestrians that the vehicle will yield. To ensure safety and comfort, a minimum gap time of 2 seconds and a minimum distance to the zebra crossing of 2 meters (while pedestrians are still on the zebra crossing) should be respected.

Notably missing from the manoeuvres in this example is 'accelerate', as this would communicate that the AV would want to cross before the pedestrians, causing insecurity and discomfort should the pedestrians decide to cross the road. It would also contradict the explicit HMI output. The exact allowed combinations of manoeuvres, HMI output and constraints will be delivered by WP2 and WP4.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

When there are no other TPs around, there is also an AV Goal 'default'. This simply implies that there are currently no restrictions from the AV Goal and planning can be done freely in that regard. Whether the HMI shows that the AV is in 'default' mode is to be determined. This default goal also comes into effect after all recent interactions have finished and no new interactions are necessary. Thus, after all pedestrians crossed the zebra crossing, the AV changes into default mode and can continue its course, also being allowed to accelerate again.

Furthermore, the general AV Goal 'emergency' is defined. This goal comes into effect whenever any effort of producing a reasonable plan fails, e.g. because constraints cannot be met. This results in the AV performing a minimum risk manoeuvre. Additional AV Goals will be defined based on the results of WP2 and WP4.

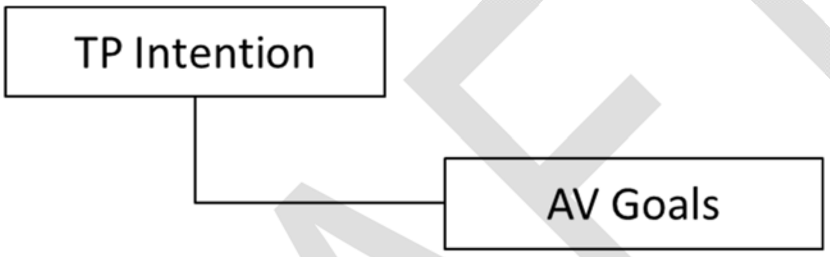


Figure 12: General structure of a TP Intention.

The TP Intentions define an allowed set of AV Goals as shown in Figure 12. The AV then chooses the appropriate goal based on the circumstances. Additionally, an initial goal is provided by the module Situation Matching, also taking into account traffic rules and possibly other factors.

TP Intention and AV Goals can even contradict each other to a certain extent. An example is given in Figure 13: the intention of the TP is to cross the road, the AV is allowed to choose 'use right of way' as a goal. If both the TP and the AV would insist on their behaviour, a critical situation could occur. This is prevented in two ways: First, the AV may only choose the goal 'use right of way' if it actually has right of way. A similar situation would be a pedestrian trying to cross a road without a zebra crossing while the AV approaches. Furthermore, the AV Goal 'use right of way' also includes constraints that prevent situations from becoming dangerous. One of those constraints could be a minimum time to collision of several seconds. Once the time to collision falls below this threshold, 'use right of way' is no longer applicable and the AV has to choose a different goal, which may be 'give way' or 'emergency'. Alternatively, the TP Intention might have changed from 'cross the road' to 'give way', enabling the AV to safely pass without conflict. In either case, the situation is solved safely. Additional TP Intentions will be defined based on the results of WP2 and WP4.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

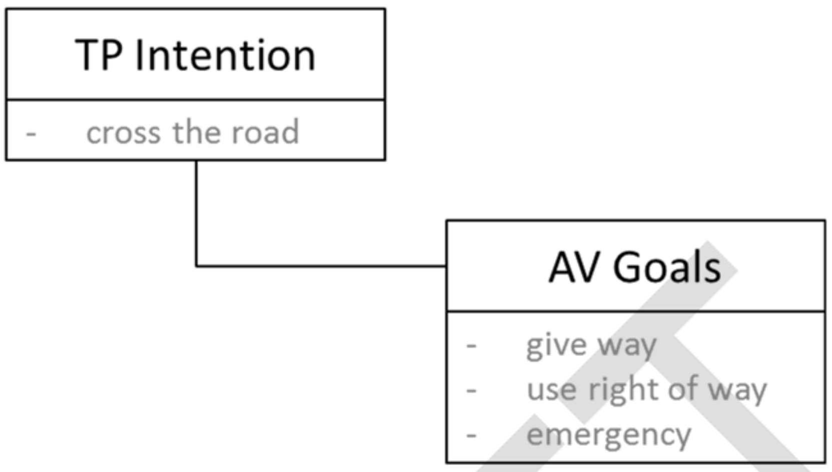


Figure 13: Specification of TP Intention 'cross the road'.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

4. Situation Matching module

4.1 Description

The matching of the actual current traffic situation with one or more digital scenarios is the main responsibility of the Situation Matching module. This is essential for the Interaction Planning module, in order to identify the set of actions to be performed by the AV. To achieve this matching, the Situation Matching module needs both data from the environment, which is provided by the Perception module, the Situation Awareness module and the sensors and the list of digital scenarios, provided by the Scenarios Catalogue.

After that, its role is twofold. Firstly, it produces the matching with one (or more) of the provided scenarios, which is fed to the Interaction Planning module, and also annotates the detected TPs (needed both by the Interaction Planning module and the Traffic Participants Behaviour Prediction Module).

Finally, it takes into account the previous actions (former calculated trajectory), which are given by the Safety Layer module, in terms of a feedback loop. The feedback enables the module to run dynamically at all times, by checking the AV's current trajectory and either ensuring that the current matched scenario is still valid or repeating the matching process, in case of an unexpected change. The overall functionality and communication of the Situation Matching module is presented in Figure 14.

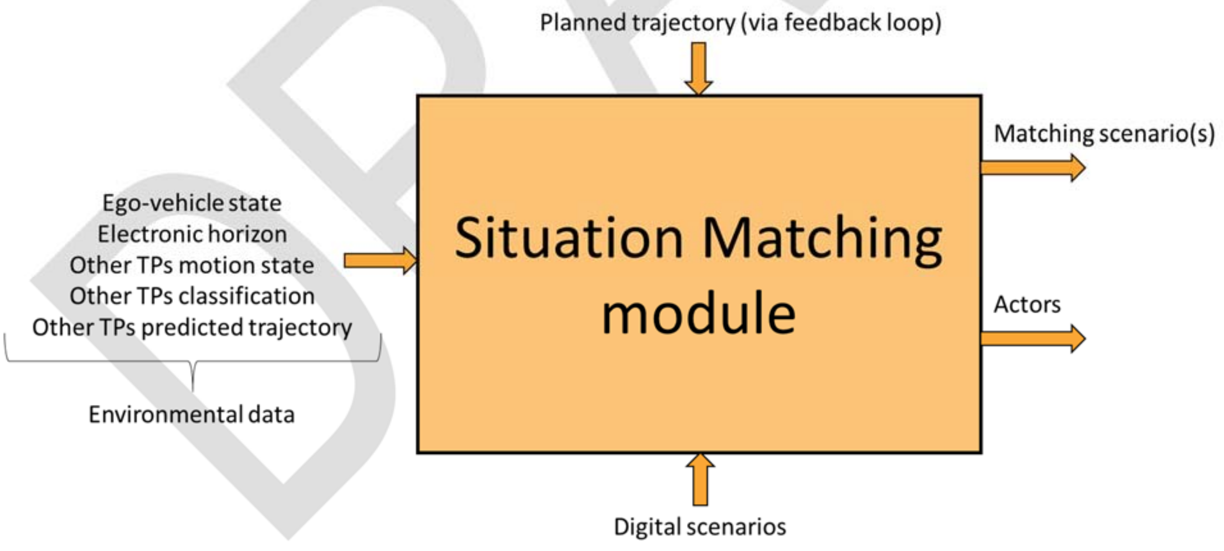


Figure 14: Situation Matching concept.

When all the literals are well-defined in the Scenarios digital catalogue, each one can be broken down into specific data to be provided from the system. For example, the following data are needed for the "approaches(av,_RS)" literal:

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

- AV Planned Trajectory (from trajectory planner/safety layer)
- AV position on the map (from localisation)
- Road Topology (from the map)

Through that process, the real-life scenarios are modelled into digital rules. From that point, it is the responsibility of the other interconnecting components to provide the Situation Matching module with accurate data, regarding the current state of the traffic scene and the responsibility of the implemented rule engine to process the provided data and calculate the most relevant scenario(s). All required data are presented in Table 5.

Table 5: Situation matching data requirements

Required input		
Literal	Data	Provider component
approaches(av, _RS)	AV Planned Trajectory, AV position on map, Road Topology	Trajectory planner / Safety layer, Localisation, Map
entering(av, _PS) or leaving(av, _PS)		
approaches(av, _RS) and willTurn(av)		
crossing(_RS)	type of road segment	Map
intersection(_RS)		
noTrafficLights(_RS)	traffic lights on road segment	Map
detected(_TP)	existence of other TPs	Dynamic object detection and classification
nonMotorised(_TP)	TP classification	Dynamic object detection and classification
vehicle(_TP)		
wantsToCross(_TP, _RS)	non-motorised TP intention	Pedestrian Intention feature recognition and/or Traffic participants behaviour prediction
unsignalised(_RS)	traffic signs and lights on road segment	Map
signalised(_RS)		
crossPaths(av, _TP)	AV Planned Trajectory, TP trajectories	Trajectory planner/Safety layer, Traffic participants behaviour prediction
parkingSpace(_PS)	type of environment segment	Map
leaving(_TP, _PS)	motorised TP intention	Motorised TP's intention feature recognition and/or Traffic participants behaviour prediction

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Lastly, another thing to consider is that since the basis of all incoming information is the result of what the sensors perceive from the environment, it is natural to assume that in some cases the requested data is not provided as a definite value, but will include a probability of the corresponding measurement. Hence, the Situation Matching module should also account for that case and be prepared to accept information of this nature. Moreover, as the system scales up and the number of all possible scenarios grows, there can easily occur some scenario mismatch; the Situation Matcher can either detect multiple scenarios for one traffic scene or generate a whole array of possible scenarios, each with different probability.

Assume that *N* different observation inputs are needed to match the situation with scenario *s*. The observations (e.g. classification, localization, behaviour prediction etc.) are noted $o_i\,, i\ \in [1, N]$. The (marginal) probability of *s* is noted as *P(s)* and is calculated as follows (using the law of total probability) [6]:

$$P(s) = \sum_{i=1}^{N} P(s\ \cap o_i) = \sum_{i=1}^{N} P(s\mid o_i)\,P(o_i)$$

If every measurement contributes equally to the recognition of the scenario, we can define:

$$P(s\mid o_i) = \frac{1}{N}, \forall\, i\ \in [1, N]$$

Therefore, we have:

$$P(s) = \frac{1}{N}\sum_{i=1}^{N} P(o_i)$$

Where $P(o_i)$, i.e. the probability for each measurement, is taken directly from the corresponding data source.

4.2 Design and Implementation plan

After the real-life scenarios and use-cases have been digitalized, some kind of a rule engine should exist to handle all incoming information and then decide the current scenario that takes place in the traffic environment. Moreover, given the feedback loop of the CCPU architecture, this process should occur dynamically and continuously at all times, not only to detect new scenarios and actors, but also to validate the previously chosen scenario or to re-evaluate, in case of an unexpected event.

By definition, the process described above requires calculations and decisions to be made in an extremely fast pace for two reasons. Firstly, the rate of scene changes in a traffic environment depends on the velocity of the acting vehicles and TPs, which in turn is relatively large, producing a wide range of fast changing incoming data. Secondly, the outcome of the CCPU must be produced almost in real-

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

time, as a potential delay may jeopardize safety of the rest acting TPs (other vehicles, pedestrians, bicyclists, etc.). This requirement was considered during the investigation of the possible languages and tools for the realisation of the component.

The first proposal was to develop the ontology using the Web Ontology Language (OWL) [7] in combination with the Semantic Web Rule Language (SWRL) [8]. OWL offers many advantages. Since it is a Web standard, it would enable integration of existing ontologies to our system. Moreover, it would facilitate the implementation, due to the existence of many tools for ontology creation, which are supported by a large community. Finally, it comes with different expressivities to ensure decidability and efficiency (e.g. OWL Lite and OWL DL).

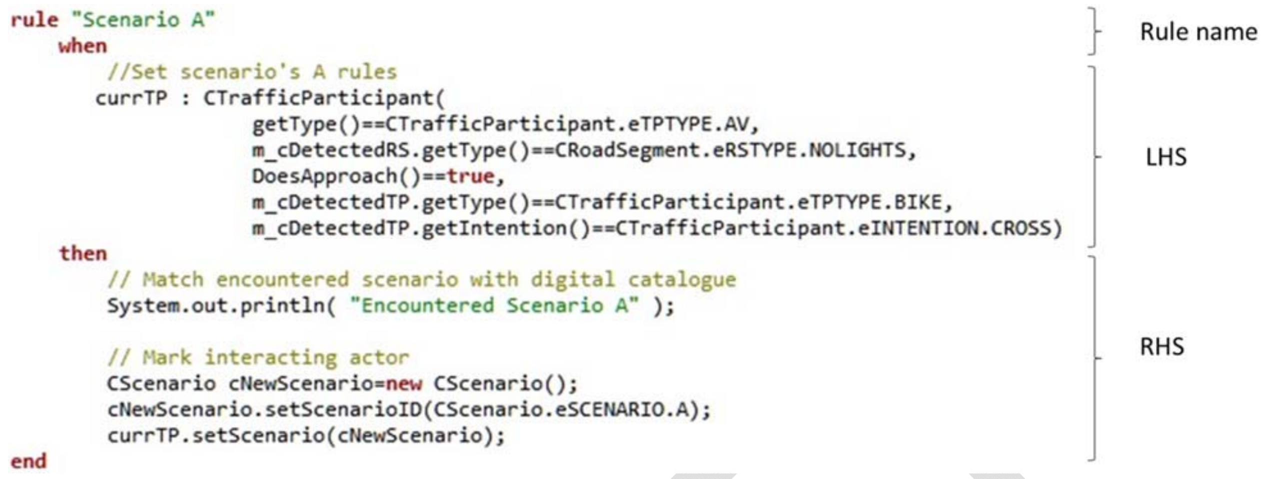
Nevertheless, it seems that this solution could not fulfil the execution time restriction [9, 10]. Consequently, different approaches were proposed and shall be examined in parallel:

a. Drools Engine, a Java-based inference engine with its own rule language (DRL) [11], since it seems to overpower OWL with SWRL and similar rule engines in terms of inference execution time [9, 12].
b. SWI-Prolog, an implementation of the programming language Prolog by the University of Amsterdam, focused on semantic web applications [13], which promises great scalability w.r.t the number of rules [10].
c. One-to-one database directives approach, instead of a semantic approach, as a back-up solution.

The Drools Engine approach is currently prioritized for the purposes of interACT's investigation. The goal is to produce a first working prototype system that operates with a small number of rules to examine the time-complexity, after an optimization has been performed, specifically for our module. In that sense, we can evaluate the performance of the Drools Engine -and of any other tool- for the scope of our project and work towards expanding in a larger scale of rules afterwards.

Drools is a Business Rules Management System solution, with its rule engine implemented in Java. It utilizes a production rule system, a knowledge-based system that performs reasoning, according to the knowledge representation input of the knowledge base. The production rule system works in a declarative manner to express first-order logic, which matches the logic required for the Scenario rules of the Situation Matching module, as described in Table 4. The processing is handled by an inference engine, which in turn is responsible for the comparison of data against predefined rules and draw out conclusions. An example of a Drools rule can be seen in Figure 15.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.



```
rule "Scenario A"
    when
        //Set scenario's A rules
        currTP : CTrafficParticipant(
                    getType()==CTrafficParticipant.eTPTYPE.AV,
                    m_cDetectedRS.getType()==CRoadSegment.eRSTYPE.NOLIGHTS,
                    DoesApproach()==true,
                    m_cDetectedTP.getType()==CTrafficParticipant.eTPTYPE.BIKE,
                    m_cDetectedTP.getIntention()==CTrafficParticipant.eINTENTION.CROSS)
    then
        // Match encountered scenario with digital catalogue
        System.out.println( "Encountered Scenario A" );

        // Mark interacting actor
        CScenario cNewScenario=new CScenario();
        cNewScenario.setScenarioID(CScenario.eSCENARIO.A);
        currTP.setScenario(cNewScenario);
end
```

Figure 15: Drools rule example.

Drools' framework is ideal for the purposes of the Scenario Matching module, as the scenario rules (shown in Table 4) can be directly translated into conditions and be placed in the Left Hand Side (LHS), while the outcome along with the communication with the rest of the modules can be implemented in the Right Hand Side (RHS), as a Java process.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

5. Interaction Planning and execution module

5.1 Description

In interACT, the AV has to take actions autonomously, recognize the intentions of the other TPs and adapt its behaviour accordingly. The knowledge base for the interactions between the AV and other TPs is described in chapter 3.2, but it still has to be applied. In order to do so, the following concept for the Interaction Planning module is presented.

The task of the CCPU is to make the AV behave in a consistent manner regarding its HMI output and the vehicle motion to communicate its goals with other TPs, enabling the AV and other TPs to interact with each other. The task of the Interaction Planning module is to plan that interaction ahead of time to make sure HMI output and vehicle motion are consistent over the complete course of the interaction.

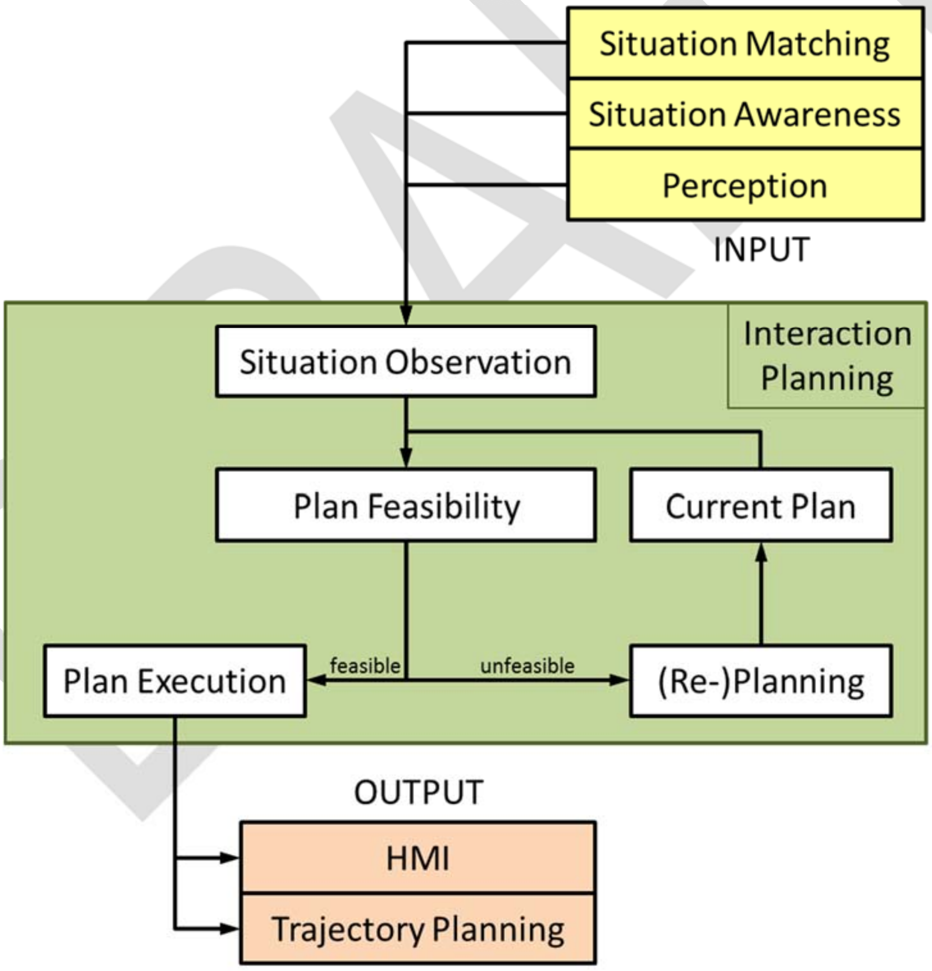


Figure 16: General structure of Interaction Planning.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Figure 16 shows the high-level structure of the module. It receives data from the functional blocks Situation Matching, Situation Awareness and Perception and processes the data. The generated output is then used by Trajectory Planning and the HMI.

The inputs from the other components are collected and evaluated during a Situation Observation process. Based on the new information, the Plan Feasibility process evaluates whether the current plan is still in the detected situation. If the plan becomes unfeasible, a new planning process is started to generate a new plan. If the plan is still feasible, it is executed and the relevant outputs are generated. This process is repeated several times per second as the situation continuously evolves. A more detailed description is given in the design and implementation plan.

5.2 Design and Implementation plan

Each process in the module Interaction Planning has its dedicated task. The first process is the Situation Observation as shown in Figure 17. Here all the inputs from other modules are processed. The currently detected situation from Situation Matching is stored along the recommended initial AV Goal. Furthermore, the intentions of the other TPs need to be classified based on the data from Situation Awareness and Perception to be able to derive according AV Goals from there as well. Last but not least, the internal environment representation needs to be updated to always fit the real world situation, e.g. positions of other TPs, but also the ego vehicle state.

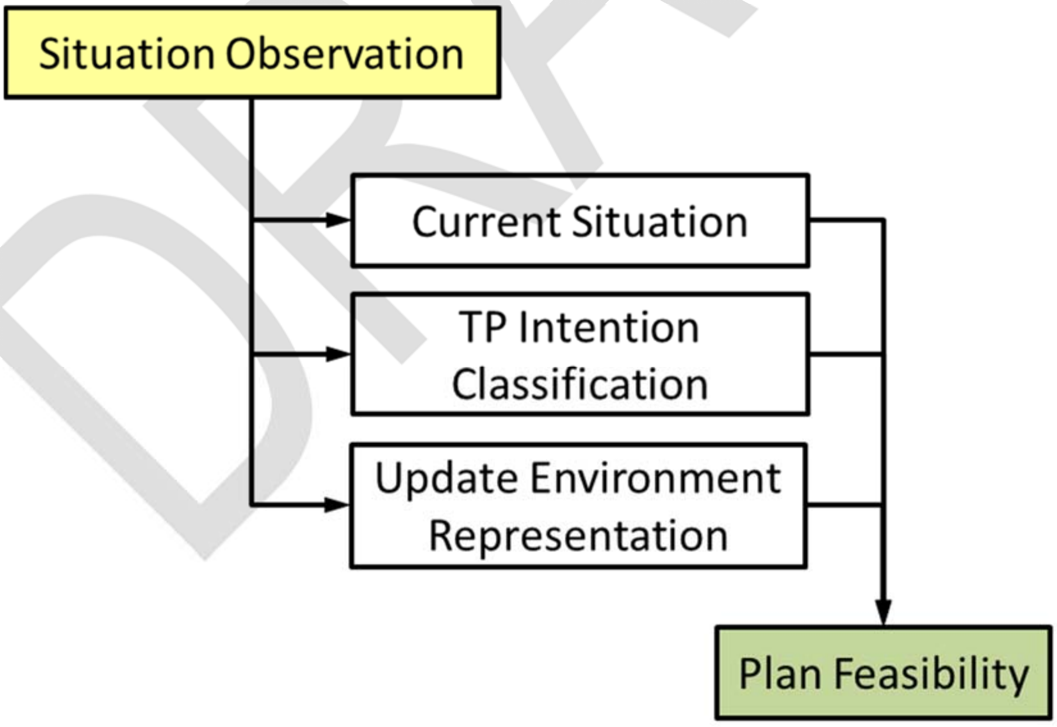

Situation Observation
Current Situation
TP Intention Classification
Update Environment Representation
Plan Feasibility

Figure 17: Structure of Situation Observation.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

A plan may only be executed when it is feasible, thus the feasibility has to be determined. A plan is not feasible anymore if the current situation has other AV Goals than the current plan was made for. A new situation does not necessarily need a new plan though if the AV Goal is still applicable in the new situation. In this way, the AV becomes more consistent and predictable as a constant change of behaviour is prevented. Furthermore, all constraints concerning the current AV Goal must be met. If they are, the plan can be executed, otherwise a new plan has to be generated.

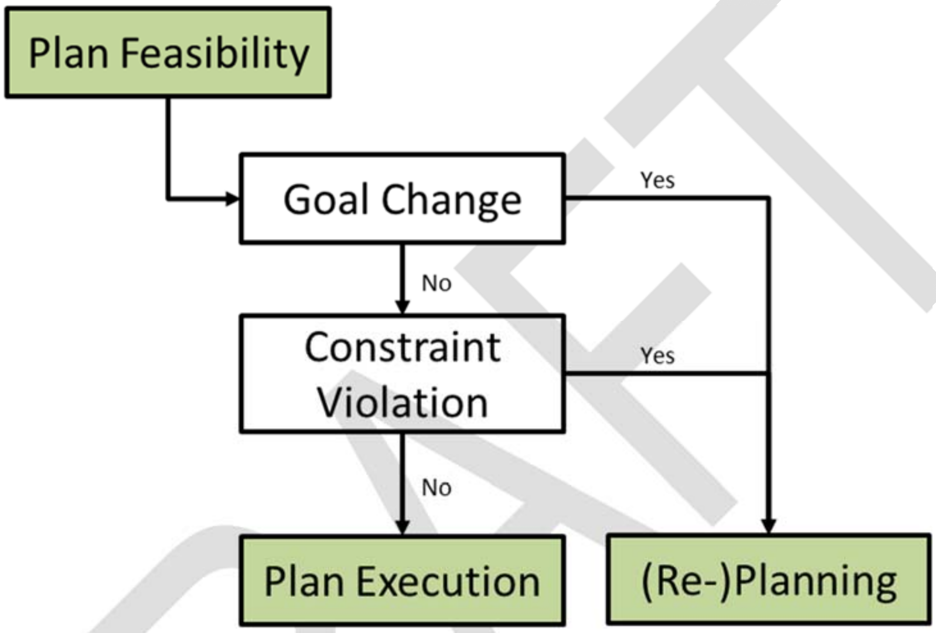


Figure 18: Structure of Plan Feasibility.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

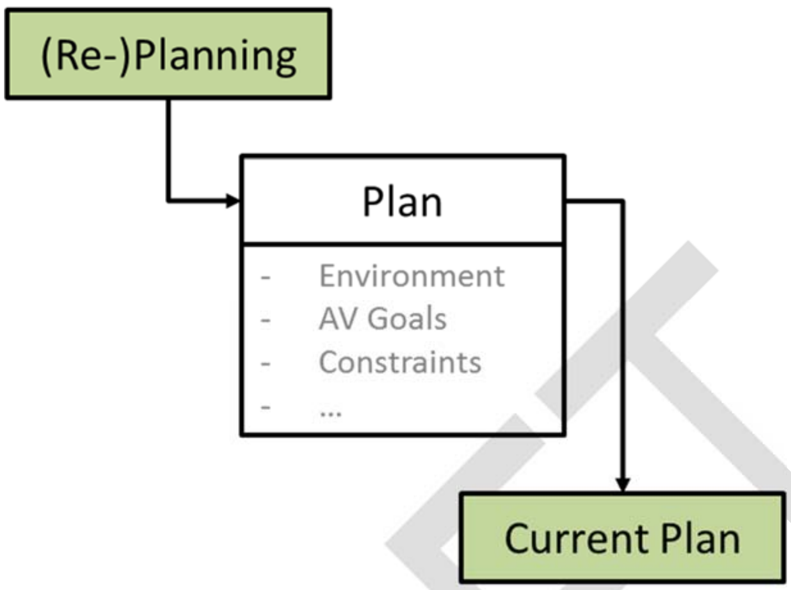


Figure 19: Structure of Planning.

When a new plan is requested, the Planning process takes into account the current environment, the allowed AV Goals and corresponding constraints to generate a new plan fitting for the new situation as shown in Figure 19. How the exact outputs will look like is still to be determined by WP2 and WP4. For the approach behaviour of the AV to other TPs, a set of maximum speeds for certain road sections could be generated as well as virtual stop lines, if stops are necessary. The planned output for the HMI could be based on a set of triggers and HMI tasks. This is necessary to make sure that HMI and vehicle movement function in a synchronized manner. As the HMI only gets information from Interaction Planning, it has no information about the environment and thus cannot ensure the synchronized behaviour itself.

Once a feasible plan is found, it is executed by the Plan Execution process. Whenever a new plan is generated, Plan Execution generates the appropriate commands for Trajectory Planning (e.g. constraints for the planner) as well as the initial HMI task. If the plan does not change, Plan Execution makes sure that whenever a trigger is reached, the HMI also gets an update and thus the AV acts in a coherent and synchronized manner. Depending on the functionalities of Trajectory Planning, in particular the planning horizon and the duration of the interaction, it might also be necessary to send more regular updates here as well.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

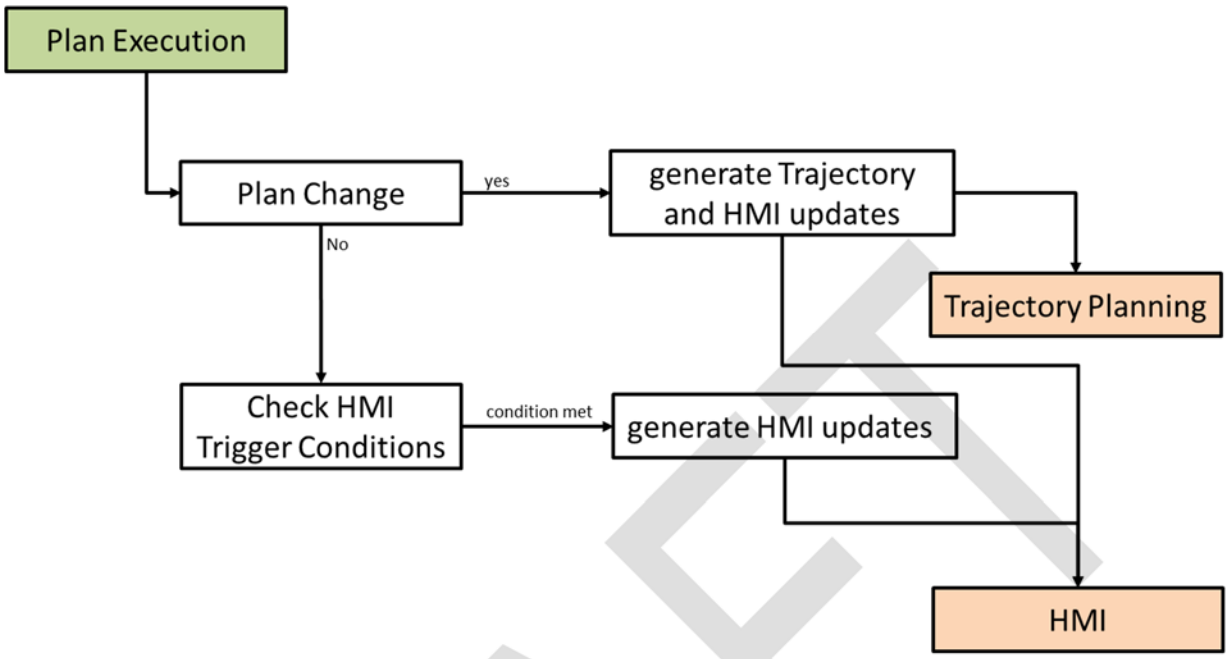


Figure 20: Structure of Plan Execution.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

6. Trajectory Planning and execution module

With reference to deliverable D1.2 “Requirements and system architecture and interfaces for software modules”, the following figure presents the system architecture for the *interACT* demonstrators:

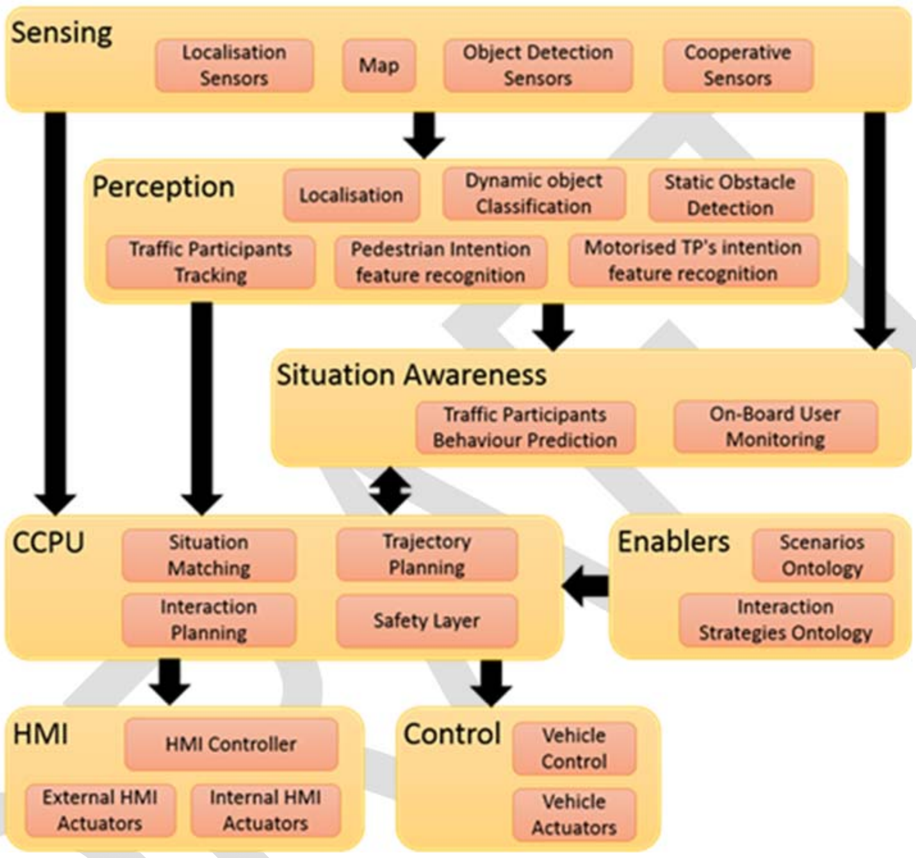


Figure 21: Sub-system and component in the functional blocks diagram.

As illustrated in the figure above, the Trajectory Planning is a module inside the CCPU. The goal of this section is to provide an overview of the possible methods and algorithms, among which, one will be selected for the implementation in the next phase of the project.

6.1 Description

Path/Trajectory planning has been a subject of study for the last decades, especially in mobile robotics. Most of the authors divide the problem into global and local planning. A review of the different approaches and concept definitions (as global, local or reactive motion planning) can be found in [14–16]. In fact, a great amount of navigation techniques have been taken by this domain and then modified

This report is part of interACT project that has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement no 723395.

to face the challenges of road networks and driving rules [17–19]. Some relevant algorithms are described in the next section.

6.2 Design and Implementation plan

To limit the scope of this survey about the most relevant path planning algorithms implemented in motion planning for automated driving, we focus on aspects of decision making, motion planning, and control for self-driving cars, in particular, for systems falling into the SAE automation level of 3 and above. For the same reason, the broad field of perception for autonomous driving is omitted (for more details on that, please see [20–23]). The decision making in contemporary autonomous driving systems is typically hierarchically structured into route planning, behavioural decision making, local motion planning and feedback control (see also the European project ROBUST-SENSE, https://www.robustsense.eu/). The partitioning of these levels are, however, rather blurred with different variations of this scheme occurring in the literature or in different projects.

6.2.1 Graph Search-based Planners

At the highest level, a vehicle decision-making system must select a route through the road network from its current position to the requested destination (so, the basic idea is to traverse a state space to get from point A to point B). By representing the road network as a directed graph with edge weights corresponding to the cost of traversing a road segment, such a route can be formulated as the problem of finding a minimum-cost path on a road network graph. This means that the state space is often represented as an occupancy grid or lattice that depicts where objects are in the environment. From the planning point of view, a path can be set implementing graph searching algorithms that visit the different states in the grid, giving a solution (that not necessarily is the optimal one) or not (there is no possible solution) to the path planning problem. Thus, the Graph Search methods discretize the configuration space of the vehicle as a graph, where the vertices represent a finite collection of vehicle configurations and the edges represent transitions between vertices. The desired path is found by performing a search for a minimum-cost path in such a graph. Graph search methods are not prone to getting stuck in local minima, however, they are limited to optimize only over a finite set of paths, namely those that can be constructed from the atomic motion primitives in the graph. In addition, although useful in many contexts, the applicability of variational methods is limited by their convergence to only local minima.

Dijkstra algorithm. It is a graph searching algorithm that finds single-source shortest path in the graph. The configuration space is approximated as a discrete cell-grid space, lattices, among others (e.g. [18, 24]). Description of the concept and implementation of the algorithm can be found in [25–28].

A-star algorithm (A*). It is a graph searching algorithm that enables a fast node search due to the implementation of heuristics (it is an extension of Dijkstra's graph search algorithm). Its most important

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

design aspect is the determination of the cost function, which defines the weights of the nodes. It is suitable for searching spaces mostly known a priori by the vehicle [29], but costly in terms of memory and speed for vast areas. Several applications in mobile robotics have used as basis for improvement, such as the dynamic A* (D*) [30], and Anytime D* [31], among others (as well as [32–34]).

However, since the graphs representing road networks can contain millions of edges, the use of these classical shortest path algorithms (such as Dijkstra [35] or A* [36]) may be impractical or very difficult.

State Lattice algorithm. The algorithm uses a discrete representation of the planning area with a grid of states (often a hyper-dimensional one). This grid is referred as state lattice over of which the motion planning search is applied [37]. The path search in this algorithm is based in local queries from a set of lattices or primitives containing all feasible features, allowing vehicles to travel from an initial state to several others. A cost function decides the best path between the precomputed lattices. Examples of the applications of these algorithms can be found in [38] for A* and in [39] for D*).

6.2.2 Sampling-based Planner

These planners try to solve timing constrains (i.e. planning in high dimensional spaces) that deterministic methods cannot satisfy. The approach consists on randomly sampling the configuration space or state space, looking for connectivity inside it [17]. The downside is the fact that the solution is suboptimal. The most commonly used methods in robotics are the **Probabilistic Roadmap Method (PRM)** [40] and the **Rapidly-exploring Random Tree (RRT)** [41, 42]. It belongs to the sampling-based algorithms applicable to on-line path planning [43] and it has been proposed by La Valle as an efficient method for finding feasible trajectories for high-dimensional non-holonomic systems. It allows a fast planning in semi structured spaces by executing a random search through the navigation area. It also has the ability to consider non-holonomic constraints (such as maximum turning radius and momentum of the vehicle). In more details, the rapid exploration is achieved by taking a random sample from the free configuration space and extending the tree in the direction of the random sample. In RRT, the vertex selection function returns the nearest neighbour to the random sample, according to the given distance metric between the two configurations. The extension function then generates a path in the configuration space by applying a control for a fixed time step that minimizes the distance to random sample (originally selected). Under certain simplifying assumptions (random steering is used for extension), the RRT algorithm has been shown to be probabilistic complete. Moreover, Karaman and Frazzoli [43, 44] demonstrated that the RRT converges to a suboptimal solution with probability one and designed an asymptotically optimal adaptation of the RRT algorithm, called RRT*. The RRT* at every iteration considers a set of vertices that lie in the neighbourhood of newly added vertex and: a) connects this new vertex to the vertex in the neighbourhood that minimizes the cost of path; b) rewires any vertex in the neighbourhood to the new vertex if that results in a lower cost path (from the initial one).

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

6.2.3 Interpolating Curve Planners

Interpolation is defined as the process of constructing and inserting a new set of data within the range of a previously known set (reference points). This means that these algorithms take a previously set of knots (e.g. a given set of way-points describing a global road map), generating a new set of data (a smoother path) in benefit of the trajectory continuity, vehicle constraints and the dynamic environment the vehicle navigates [45]. In the presence of obstacles, it suffices to generate a new path to overcome the event and then re-entry the previously planned path.

Lines and circles. Different segment road network can be represented by interpolating known waypoints with straight and circular shapes. It is a simple mathematical method to approach the planning problem in car-like vehicles [46, 47].

Clothoid Curves. In the interest of having continuous curvature paths, clothoid segments are also sometimes used [48]. The motion primitives can be also obtained by recording the motion of a vehicle driven by an expert driver [49]. In particular, this type of curve is defined in terms of Fresnel integrals [50]. Using clothoid curves is possible to define trajectories with linear changes in curvature since their curvature is equal to their arc-length; making smooth transitions between straight segments to curved ones and vice versa. Usually, the clothoids are implemented for the design of highways and railways, as well as they are suitable for autonomous cars [51].

Polynomial Curves. These curves are commonly implemented to meet the constraints needed in the points they interpolate, i.e. they are useful in terms of fitting position, angle and curvature constraints, among others. The desired values or constraints in the beginning and ending segment will determine the coefficients of the curve (see [52–54]).

Bézier Curves. These are parametric curves that rely on control points to define their shape. The core of Bézier curves are the Bernstein polynomials. These curves have been extensively used in many domains, such as aeronautical and automotive design; moreover, they are also used to approximate Clothoid curves [55]. The advantage of this kind of curves is their low computational cost, since the curve behaviour its defined by control points. The constraints at the beginning and the end of the curvature can be met by correctly placing these control points according to different properties described in [56–58].

Spline Curves. A spline is a piecewise polynomial parametric curve divided in sub-intervals that can be defined as polynomial curves [52, 59], b-splines [60, 61]. The junction between each sub-segment is called knot and they commonly possess a high degree of smoothness constraint at the joint between the pieces of the spline.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

6.2.4 AI-based Planner

Real-world driving is however characterized by uncertainty over the intentions of other traffic participants. The problem of intention prediction and estimation of future trajectories of other vehicles, bikes and pedestrians has also been studied. Machine Learning (ML) based techniques can represent a good solution, when many data are available. Examples of that are the Gaussian mixture models [62], Gaussian process regression [63], the learning techniques reportedly used in Google's self-driving system for intention prediction [64], and model-based approaches for directly estimating intentions from sensor measurements (see [65, 66]). This uncertainty in the behaviour of other TPs is commonly considered in the behavioural layer for decision making using probabilistic planning formalisms, such as Markov Decision Processes (MDPs) and their generalizations (see an example in [67]). Several works [68–71] model unobserved driving scenarios and pedestrian intentions explicitly using a partially-observable Markov Decision Process (POMDP) framework, proposing specific approximate solution strategies.

A more exhaustive survey can be found in [19, 72], for a complete overview on these topics.

6.3 Next steps

At the current state of the project, we are still evaluating which algorithm(s) can be used in more efficient way, based on the experience of the involved partners as well as on the needs of the use-cases and scenarios selected in WP1. Among the possible approaches illustrated so far, the more likely are the Model Predictive Control (MPC) and, in particular, A* and/or RRT/RRT* (Rapidly-exploring Random Tree), since they are particularly suitable for planning problems where obstacles and differential constraints (nonholonomic or kinodynamic) are present. Moreover, RRTs can be regarded as a technique to create open-loop trajectories for non-linear systems with state constraints. In this sense, RRTs has the also the objective to explore the environment and, intuitively, can be considered as a Monte-Carlo method to search the solution(s) in large Voronoi space (as aforementioned in the related section).

In particular, A* and Voronoi (or RRT*) algorithms are used as "path-finder" (that is, to find a kind of safety corridor) in a static environment. Of course, for the specific use-case of automated driving in parking space, we need to consider also moving objects. Given that, at the moment, MPC is used, inside this corridor, to find the "real" trajectory to follow, taking into account also the dynamic obstacles. At the moment, this evaluation is carrying out in simulation, to check the timing constraints and select the best solutions. The environment reconstruction is provided by the perception platform of BOSCH partner.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

7. Safety Layer

7.1 Description

Safety is ensured in interACT by predicting possible future occupancies of other TPs and checking whether there exists a time when the future occupancy of the ego vehicle intersects with the one of another TPs as shown in Figure 22. For this reason, we compute occupancies for consecutive time intervals so that no time is missed where a collision might be possible. We also always hold available a fail-safe trajectory in case of unexpected events, which is described in more detail subsequently. The use of formal methods in our safety layer enables a clear path towards certifiability of our developed system.

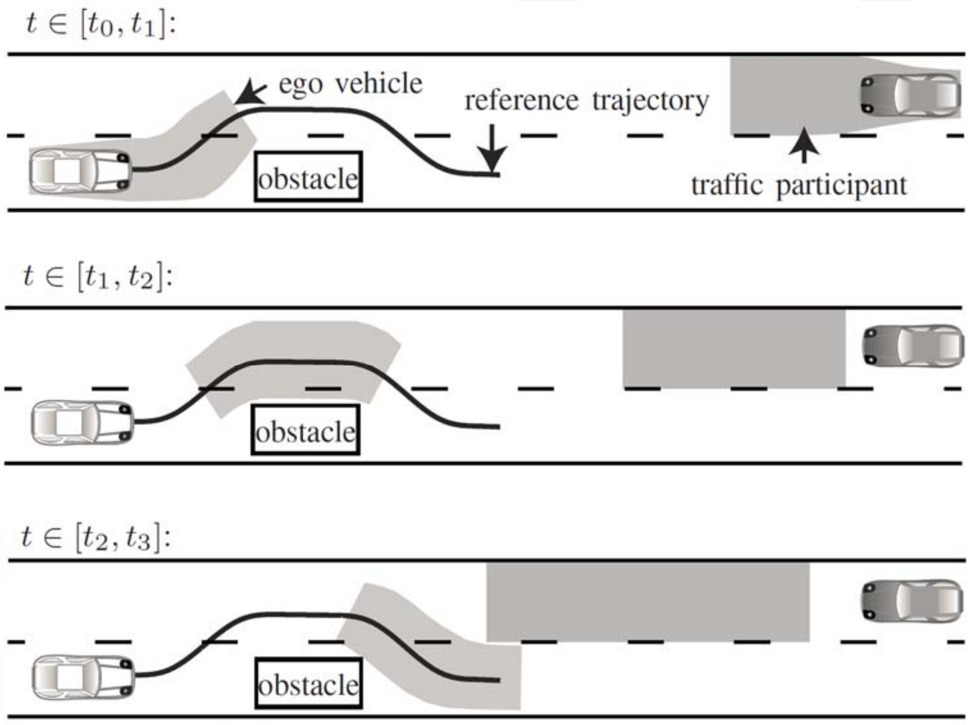


Figure 22: Safety verification using occupancy prediction.

7.1.1 Set-based prediction

Behaviour prediction is essential for planning cooperative manoeuvres since future positions of other traffic participants are forbidden regions of the ego vehicle to avoid collisions [73–75]. We provide a unified interface for TPs, as well as for manually driven vehicles and automated vehicles. For non-communicating vehicles (manually-driven and automated), exact information about future behaviour is not available, i.e. sets of possible behaviours have to be considered, resulting in occupancy regions that

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

grow over time. In order to limit the occupancy, possible behaviours of other traffic participants are restricted by traffic rules [76]. This principle is discarded, however, if the violation of a traffic rule is detected (e.g. another vehicle is over-speeding).

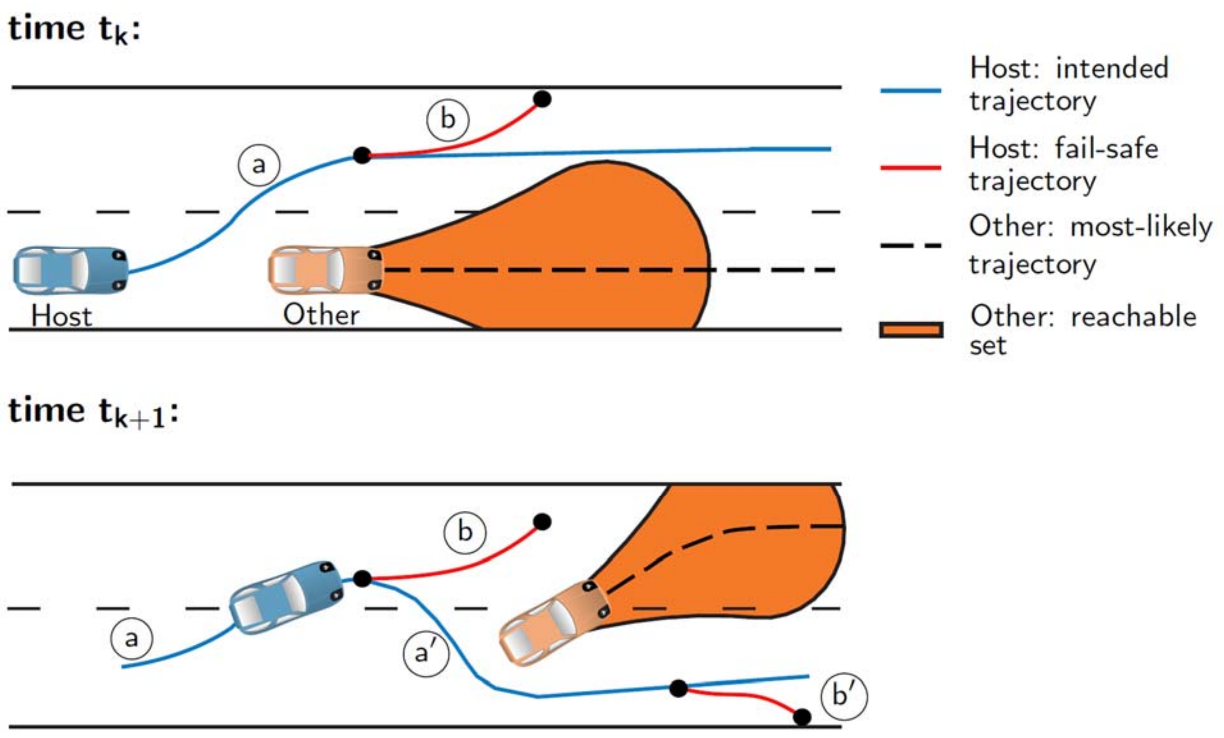

time t_k:
Host
Other
a
b
Host: intended trajectory
Host: fail-safe trajectory
Other: most-likely trajectory
Other: reachable set
time t_{k+1}:
a
b
a′
b′

Figure 23: Intrinsic legal safety using reachable sets and fail-safe trajectories.

The set-based occupancy prediction results in increasingly larger regions when the prediction horizon is increased [77]. Larger occupancies of other vehicles block space for the trajectory planning of the ego vehicle. Thus, the trajectory planning should be performed twofold as shown in Figure 23: an intended trajectory should be generated based on non-formal occupancy prediction techniques, such as single behaviours or probabilistic methods. A single behaviour for an intended, long-term manoeuvre is shown in blue in Figure 23. This long-term trajectory is repeatedly executed for a short time horizon similarly to model predictive control [78]. To verify the safety of the trajectory for the short time horizon, we attach a fail-safe trajectory, which brings the vehicle to a safe stop as illustrated by a red line in Figure 23 [79]. The time horizon of this combined trajectory (first part of intended trajectory plus fail-safe trajectory) is short, such that the proposed set-based techniques do not block overly large regions for the verification procedure. If the manoeuvre is safe, the next part of the long-term plan is executed, otherwise, the fail-safe trajectory is initiated. As a consequence, the proposed set-based occupancy prediction guarantees safe manoeuvres, while non-formal techniques provide long-term plans based on likely behaviour of other TPs.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Let us illustrate this principle using the example in Figure 23: At time t_{k+1} the other vehicle performs an unexpected lane change. As a reaction to this, the ego vehicle plans a new intended trajectory a' and a new fail-safe trajectory b'. If a' and b' can be verified on time, the ego vehicle will follow a', otherwise the fail-safe trajectory b is executed, which is known to be safe since at time t_k we have already considered all possible behaviours of the other vehicle. Since we only execute manoeuvres which are verified for all times (fail safe trajectory ends in an indefinitely safe state), by induction we are always safe. Clearly, there always exist situations where the ego vehicle cannot avoid an accident, e.g. when it is trapped in a traffic jam and is rear-ended by another vehicle. However, our approach guarantees that the ego vehicle will never cause a crash.

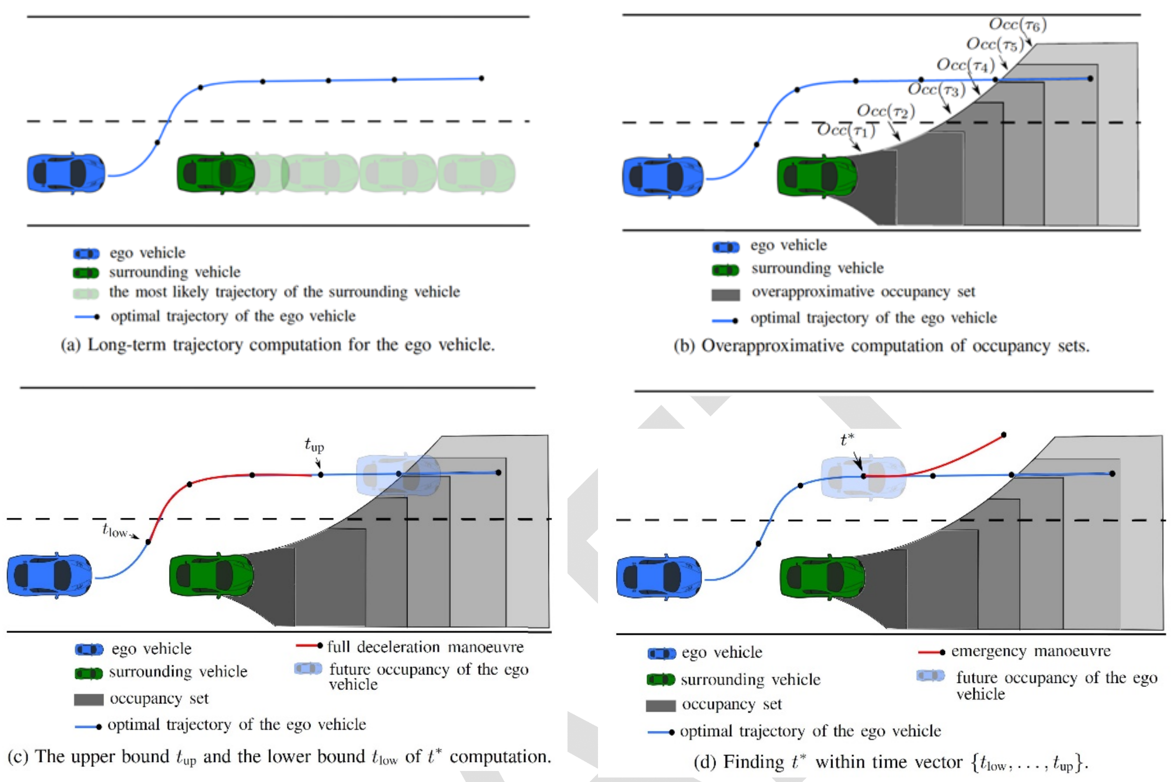
7.1.2 Fail-Safe Trajectory Generation

The other TP can perform infinitely many unpredictable manoeuvres, which are not taken into account when generating an optimal trajectory. Of course, not considering all possible manoeuvres of the leading TP might result in a collision. In order to avoid collisions, an emergency plan must be hold available, which accounts for all possible manoeuvres the leading TP can perform in a given time horizon, as shown in Figure 23.

A collision-free optimal trajectory is assumed to be given for the AV. The difference between the optimal trajectory generation and computing the emergency manoeuvre is that for the latter, the velocity must be reduced, and all possible trajectories of the lead TPs enclosed by the entire occupancy set must be avoided. There already exists research on emergency trajectory generation [80, 81]. However, most work computing possible emergency trajectories assumes that the lead TP is moving with constant acceleration or only considers static obstacles. To guarantee safety, the occupancy sets [77] are embedded in the constraint function [79]. Driving along a reference trajectory is no longer desired, but rather minimising the velocity v.

The question that arises now is when the emergency manoeuvre must be generated, such that safety is not jeopardised [82]. Generating an emergency manoeuvre for each time step is computationally expensive and typically not required. Therefore, we determine the maximum time horizon t* for which the AV can safely follow a given optimal trajectory. Moreover, a fail-safe manoeuvre starting at t* is guaranteed to exist.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.



ego vehicle
surrounding vehicle
the most likely trajectory of the surrounding vehicle
optimal trajectory of the ego vehicle
(a) Long-term trajectory computation for the ego vehicle.
$Occ(\tau_1)$
$Occ(\tau_2)$
$Occ(\tau_3)$
$Occ(\tau_4)$
$Occ(\tau_5)$
$Occ(\tau_6)$
ego vehicle
surrounding vehicle
overapproximative occupancy set
optimal trajectory of the ego vehicle
(b) Overapproximative computation of occupancy sets.
t_{up}
t_{low}
ego vehicle
surrounding vehicle
occupancy set
optimal trajectory of the ego vehicle
full deceleration manoeuvre
future occupancy of the ego vehicle
(c) The upper bound t_{up} and the lower bound t_{low} of t^* computation.
t^*
ego vehicle
surrounding vehicle
occupancy set
optimal trajectory of the ego vehicle
emergency manoeuvre
future occupancy of the ego vehicle
(d) Finding t^* within time vector $\{t_{low}, \dots, t_{up}\}$.

Figure 24: Fail-safe manoeuvre generation.

To generate a fail-safe manoeuvre, we propose an algorithm which can be summarized in three main steps, as illustrated in Figure 24. After new measurements are collected and an optimal trajectory is generated for the ego vehicle, an upper bound t_{up} of t* is determined, which represents the maximal time for which the ego vehicle can follow the long-term trajectory without intersecting with the corresponding occupancy set of other TPs. To further prune the search interval of t*, a lower bound t_{low} is computed, which represents the latest time at which full braking can be initiated so that standstill is reached before or at t_{up}. Both t_{up} and t_{low} are illustrated in Figure 24c. Finally, the maximum time horizon t* is calculated using binary search within the interval [t_{low}, t_{up}].

7.1.3 Criticality Metrics

To determine the optimal point in time at which collision avoidance should be initiated, time-based criticality metrics [83] such as the Time-To-React (TTR) [84, 85] are commonly used. The TTR describes the last point in time along the current trajectory at which an evasive trajectory exists. We present a novel approach to determine the point in time after which it is guaranteed that no evasive manoeuvre exists, i.e., by using reachable sets [86], we over-approximate the TTR. Our deterministic upper bound of the TTR can be used to find a feasible emergency manoeuvre, which avoids the collision.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

We propose an efficient method to over-approximate the TTR. Existing sampling-based methods under-approximate the TTR, since they determine the time at which they can still obtain a feasible evasive trajectory [84, 85]. In contrast, our novel set-based approach determines an over-approximation of the TTR, since by using reachable sets, we determine the time at which it is guaranteed that no evasive manoeuvre exists.

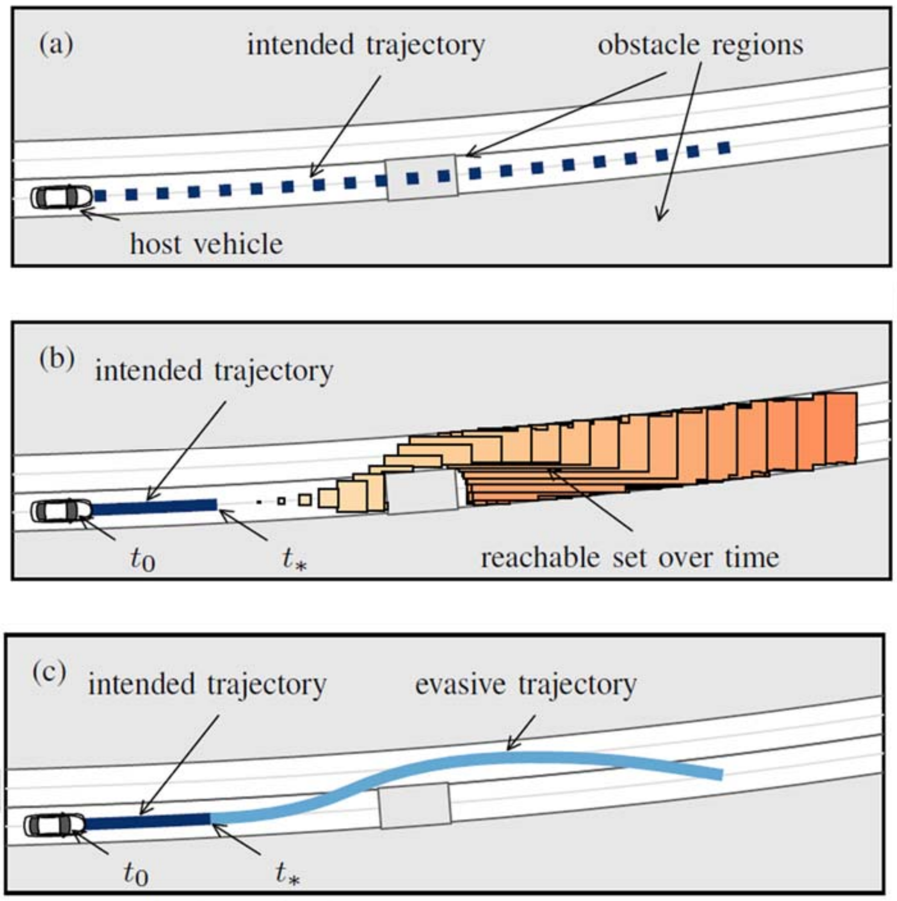


Figure 25: Determining the time to reaction of a situation in a) using reachable sets in b) or optimization techniques in c).

Given an assumed motion of the vehicle, our upper bound of the TTR makes it possible to know beforehand when, at the latest, to definitely intervene. Similarly, an AV can use the over-approximated TTR as the upper bound when searching for evasive trajectories, since it is guaranteed that no collision-free trajectory exists after that time.

Using our over-approximated TTR, one can now judge the accuracy of existing TTR computations. We demonstrate that our upper bound is a tight over-approximation by estimating the TTR using an optimization-based trajectory planner. Note that our method is deterministic, i.e., it always returns the

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

same TTR for the same configuration. Furthermore, our approach is independent of a particular prediction of other objects and can be used with any given set-based traffic prediction.

Let us explain the principle for computing a strict over-approximation of the TTR based on Figure 22. Given an intended trajectory in a), we determine the latest point in time for which it is possible to follow the intended plan and for which subsequently a reachable set exists that does not become the empty set for future point in time. This implies that we know that an evasive manoeuvre exists and that we can safely follow the intended plan. Instead of computing a reachable set, one can also solve an optimization problem and see whether a feasible solution exists that is still drivable by the vehicle. We demonstrate later that the computation with reachable sets is much more efficient than previous approaches using optimization techniques.

7.2 Design and Implementation plan

Our implementation strategy is to first realize a prototype tool in a prototyping language to see if all the components work together as intended. In a second step, the tool is realized in C++. Meanwhile, we add new features using our prototyping language and integrate them in C++ over time.

As a prototyping language, we chosen MATLAB in which we have implemented our set-based prediction to verify planned manoeuvres. Our tool is called Set-Based Prediction Of Traffic Participants (SPOT). Subsequently, we will present the structure and capabilities of SPOT [87].

7.2.1 Representation of the Surrounding Environment

In order to predict the future occupancy of other traffic participants, we have to represent the surrounding environment in a standardized way within SPOT. We have developed a standard traffic scene representation called *CommonRoad*. The overall structure of CommonRoad is presented in Figure 26. The structure is defined using XML and basically consists of a road network composed of lanelets, different types of obstacles (representing the other TPs), and the current planning problem of the AV.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

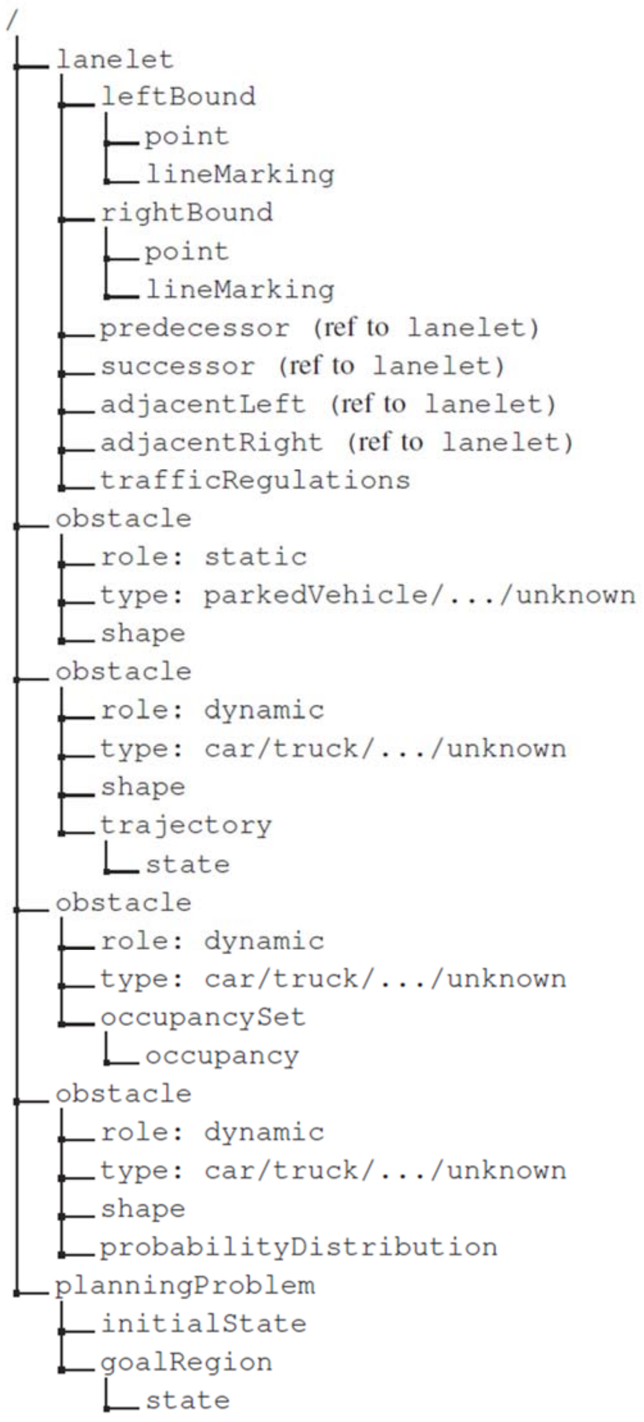


Figure 26: Data structure of CommonRoad.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

A lanelet [88] is an atomic drivable road segment which is defined by its left and right bound, where each bound is represented by an array of points (a polyline), as shown in Figure 27. We have chosen lanelets since they are as expressive as other formats, such as e.g. OpenDRIVE [89], yet have a lightweight and extensible representation. Using lanelets allows the road network to be modeled as a directed graph, where each node has four types of outgoing edges: *successor*, *predecessor*, *adjacentLeft*, and *adjacentRight*. Lanelets additionally contain traffic regulations, e.g. the speed limit.

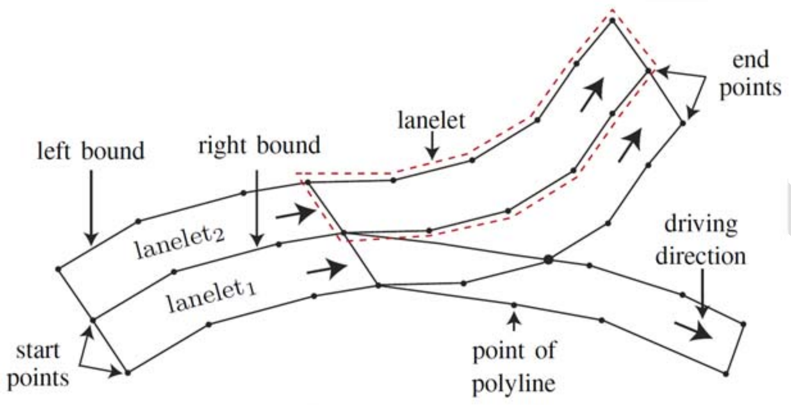


Figure 27: A road segment modeled by lanelets.

The element obstacle is used to represent different kinds of TPs within the scenario. An obstacle is either static or dynamic, which is specified by the element role. Each role allows different types of an obstacle as listed in Table 6. A static obstacle is specified by the elements role, type, and shape. In addition to static obstacles, traffic scenarios can contain dynamic obstacles. Please note that only elements of either of the following three behaviour models (with known behaviour, with unknown behaviour, or with unknown stochastic behaviour) may be present.

Table 6: Types of obstacles

Role	**Type**
Static	Parked vehicle, construction zone, unknown
Dynamic	car, truck, bus, bicycle, pedestrian, priority vehicle, unknown

7.2.2 Software Structure

The architecture of SPOT is presented in Figure 28 using the class diagram of UML3. We emulate the perception of the ego vehicle (class Perception) to consider sensor range limitations among others. Our model of the traffic scene is stored as a map (class Map) and contains lanes (class Lane), obstacles (class Obstacle), and the ego vehicle itself (class Vehicle). A hierarchical class structure behind the superclass Obstacle allows us to distinguish between static and dynamic obstacles (class *StaticObstacle* and

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

DynamicObstacle) and to represent different types of traffic participants, like passenger cars, trucks, and bicycles. To combine these different types, we use the terms obstacle and traffic participant interchangeably. Each obstacle has a property of the class Occupancy, which is computed as described in the next section.

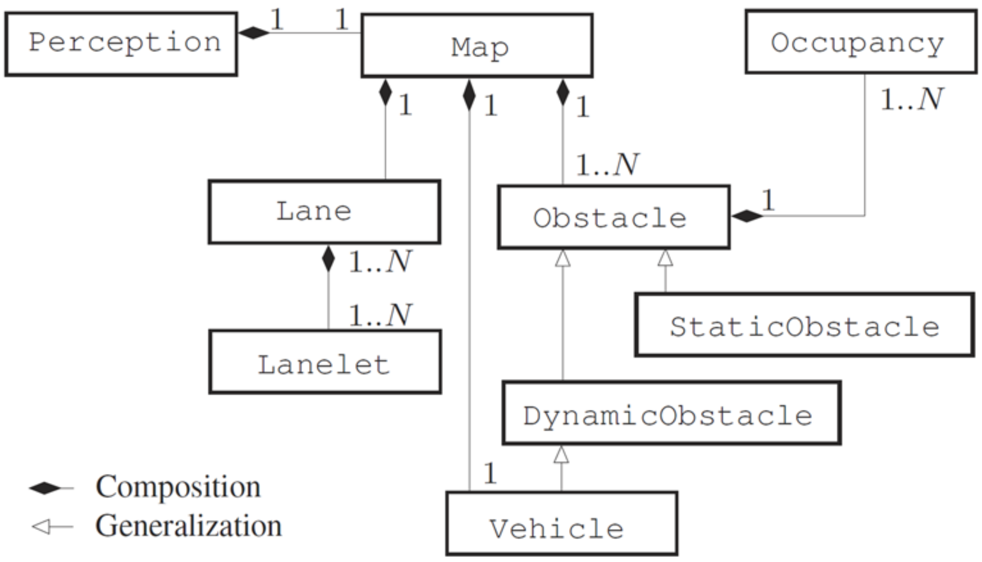


Figure 28: Class structure of SPOT.

7.2.3 Overall Algorithm

SPOT predicts the occupancy for all TPs up to a given prediction horizon and can run in parallel for each TP. First, the constraint management configures the parameters according to the set of last-measured states. Second, the reachable lanes are extracted from the road network as presented in the subsequent paragraph. Next, the occupancies of all abstractions are computed. Finally, the overall occupancy of an obstacle is returned as the intersection of all occupancies from various abstractions of the dynamics of traffic participants. Please note that we store a separate occupancy polygon for each traffic participant, time interval, and lane to make an efficient collision detection possible.

Managing Constraints: As mentioned before, we immediately adapt our model when constraints are violated. During the prediction, the method *manageConstraints* constantly checks for violations of the constraints based on the set of last-measured states, i.e. initial states X_0, and previously-measured states X_{-1} of every obstacle. As soon as we detect violations of traffic rules, we individually adjust the obstacle's parameters according to Table 7. We briefly motivate our actions for each constraint: If a traffic participant is driving faster than the speed limit by more than the speeding factor f_S, we increase the latter to ensure an over-approximative occupancy. When constraint C_{engine} is violated, our classification of the engine limits has been incorrect, and thus we remove this constraint. As soon as we detect that a traffic participant is driving backwards, its occupancy for negative speeds is included in

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

our prediction. Likewise, we anticipate an illegal lane change of an obstacle in the set of its reachable lanes (which are described next). Note that C_{amax} cannot be violated, since it is a physical constraint. Constraint C_{safe} must also not be checked for violations as explained later.

Table 7: Implemented traffic rules of other traffic participants

Constraint	**Description**
C_{amax}	Maximum absolute acceleration is limited
C_{vmax}	Positive longitudinal acceleration is stopped when a parameterized speed is reached
C_{engine}	The maximum power of an engine is considered so that maximum acceleration is not constant across all velocities.
C_{back}	Backwards driving is prohibited (except for parking manoeuvres or similar exceptions)
C_{lane}	Driving in opposite traffic if forbidden, except for allowed overtaking manoeuvres.
C_{safe}	A minimum distance to the ego vehicle must be kept to comply with the safe distance rule

Reachable Lanes: To consider all possible routes through the road network, the method *reach* searches for all reachable lanes according to constraint C_{lane}. As mentioned before, a lane is the union of longitudinally adjacent lanelets. Additionally, we define the current lanes of an obstacle as all lanes in which the obstacle is currently positioned. This simplifies the computation for the set-based prediction since other traffic participants only move along lanes within road networks.

7.2.4 Results

To illustrate the introduced methods for set-based prediction of traffic participants, we present the results of some scenarios.

Occupancy Prediction at an Intersection (Scenario I): Occupancy Prediction at road intersections is not only particularly important, but also challenging. Scenario I presents an intersection in Munich's inner city (CommonRoad ID: S=GER MUC 3a): The north-south street Leopoldstraße (5 lanes) is crossed by Hohenzollernstraße (2 lanes) and Nikolaistraße (2 lanes), which is modelled as an uncontrolled intersection. Figure 29 shows the initial configuration at t0 with Obstacles 1-3, which are all subject to the official speed limit of 50.0 km/h which corresponds to 13.89 m/s. While Obstacle 1 (blue) is driving

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

south and can manoeuvre to the two left adjacent lanes, Obstacle 2 (red) is heading north with the possibility of continuing to one of the two straight lanes or taking a right or left turn. Note the median strip on Leopoldstraße where driving is not allowed. Obstacle 3 (green) is located on Hohenzollernstraße and can take a right turn. While Figure 29 illustrates the occupancies O(t) for an intermediate time interval, Figure 30 shows the predicted occupancy for the entire prediction horizon.

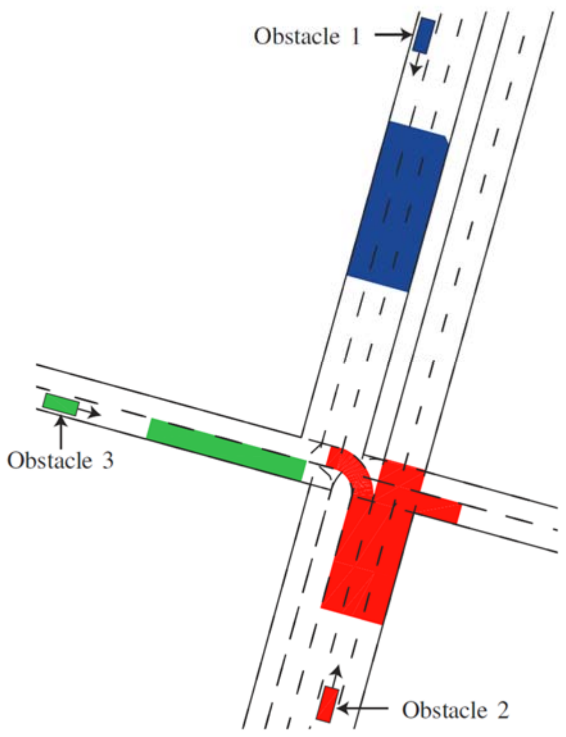


Figure 29: Initial situation and occupancy for the time interval [t3,t4].

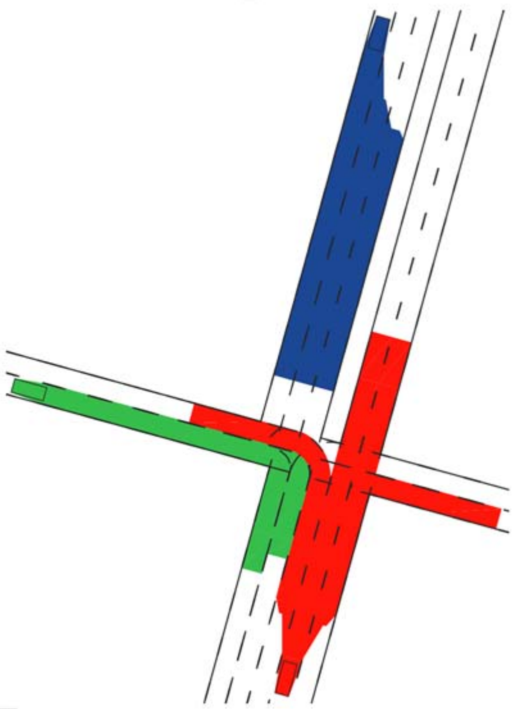


Figure 30: Occupancy of the entire prediction horizon.

Computing time to reaction at a T-intersection (Scenario II): Figure 31 illustrates the next urban traffic scenario, where the current trajectory of the ego vehicle continues straight with constant velocity, while three other traffic participants are detected at the T-intersection ahead. Since we are uncertain about the intended manoeuvre of the other vehicles, the occupancy prediction includes full acceleration and braking, and, for the vehicle approaching the intersection, turning left and right.

Figure 31 depicts the reachable set and optimized trajectory starting at different TTR candidates. It can be seen that the reachable set is very small, and thus, only few evasive manoeuvres exist. Note that, as shown in Figure 31, the optimized trajectory starting at t = 0.4 s leaves the reachable set and its maximum acceleration is larger than the maximum possible acceleration; thus, this trajectory is not dynamically feasible for our vehicle model.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

In contrast to the computation of the optimized trajectories, which take several seconds, the TTR using reachable sets can be computed in less than 30ms. We have used a machine with a 2.6 GHz Intel Core i7 processor with 20GB 1600MHz DDR3 memory.

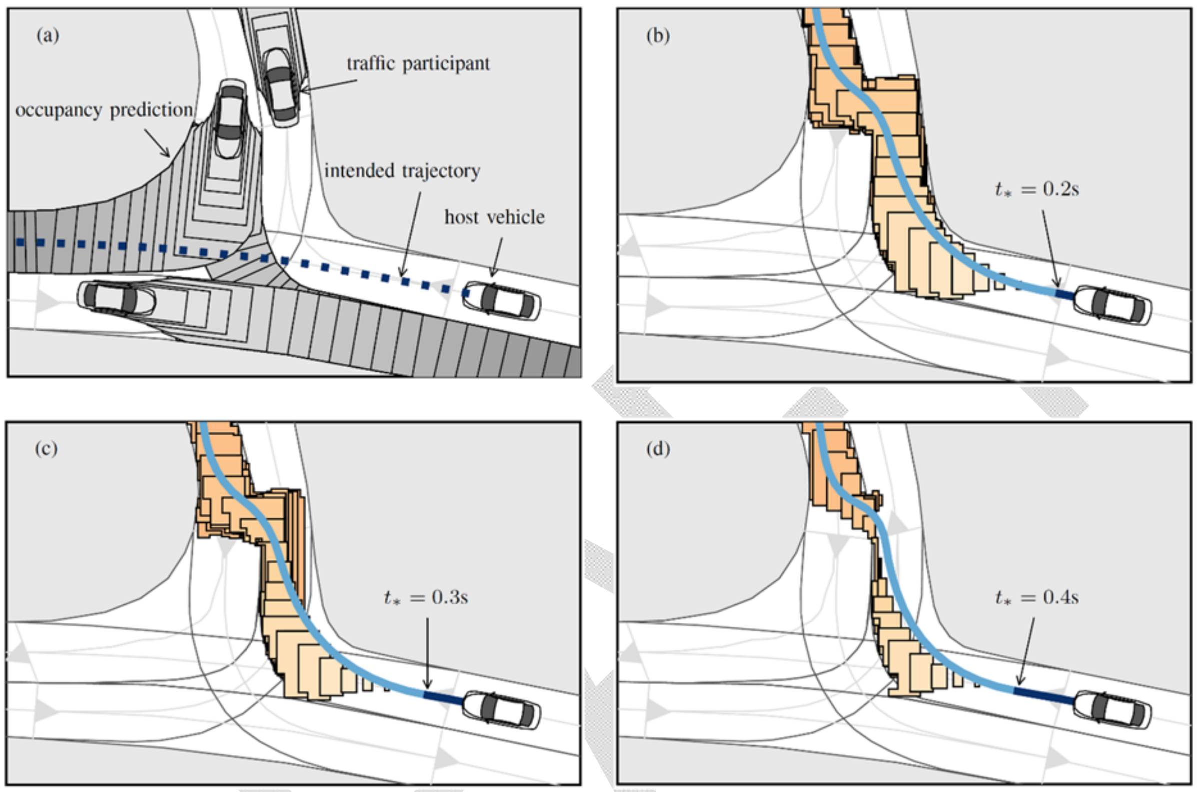


Figure 31: Obtaining the time to reaction at a T-intersection.

Fail-safe Motion Planning (Scenario III) [79]: Real traffic data is used to evaluate our proposed approach for fail-safe motion planning. The provided dataset is part of the Federal Highway Administration's Next Generation Simulation (NGSIM) project, and it contains detailed vehicle trajectories. These data were collected on June 15th, 2005, on a segment of U.S. 101 Highway (Hollywood Freeway) located in Los Angeles, using eight video cameras.

For each considered vehicle, the following information is available for every 0.1 seconds: position, velocity, and acceleration. In the simulation, the vehicles whose trajectories were recorded are considered to be leading vehicles. The host vehicle is positioned behind the leading vehicle(s); the initial velocity and acceleration are arbitrarily chosen within the given limits.

Here, a scenario with two surrounding vehicles is considered. The initial distances between the host vehicle and the other vehicles are 37 m and 49 m, and the initial velocity of the lead vehicles is 13.5m/s

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

and 13 m/s. Figure 32 shows the measured position of the surrounding vehicles at each time step, together with the generated path of the host vehicle.

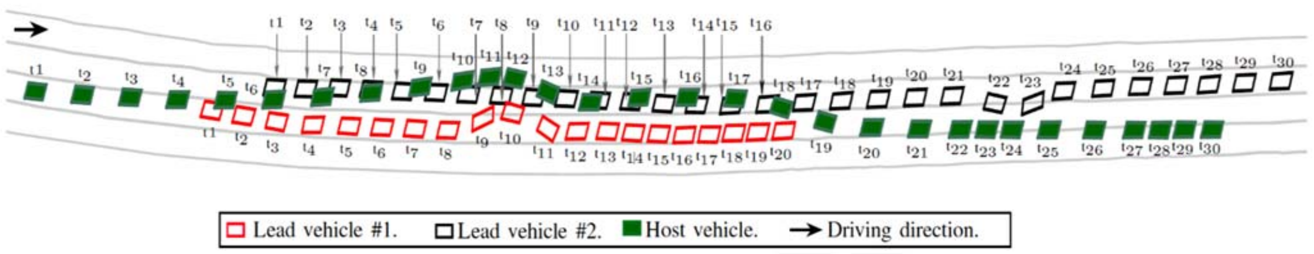


Figure 32: Fail-safe motion planning.

The lead vehicle #1 (red) performs an unexpected manoeuvre at time t_9 towards the left lane, where the host vehicle is driving. Since following the current trajectory is not safe anymore, the precomputed emergency manoeuvre, which considers all possible behaviour of the other traffic participants, is engaged. Thus, the host vehicle successfully avoids the collision by applying the available emergency manoeuvre. Next, at time t_{22} the lead vehicle #2 (black) starts a lane change manoeuvre towards the current lane, i.e. the lane on which the ego vehicle is driving. At the next time step, the lane change manoeuvre is aborted, and the host vehicle continues driving along the planned trajectory. As it can be seen, the host vehicle's path (green) is collision-free for the entire simulation.

By precomputing emergency manoeuvres which accounts for every possible behaviour of other traffic participants, the ego vehicle can indeed avoid collisions.

To conclude, by introducing the safety layer in the motion planning generation, we can guarantee that the AV does not cause any collision with the surrounding TPs. Therefore, the AV is able to safely react even to the unexpected manoeuvres of the other TPs, when the actual behaviour is different than the intended/communicated one. This property is important in all scenarios considered within interACT.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

8. Interfaces

The next step after the interfaces definition, which was presented in interACT deliverable D1.2 [1], is their detailed description and specification. Every interface each CCPU component exposes or depends on was further defined to enable components development and integration. For this purpose, there has also been a collaboration with partners outside WP3, who are responsible for interfaces on which CCPU components depend.

In parallel, proposals on tools and frameworks for technical collaboration were evaluated. The selected framework should enable easy collaboration and integration by (i) providing mechanisms for convenient message passing, (ii) supporting many programming languages (since each component may be implemented in a different language), (iii) offering tools for recording, replaying and visualizing exchanged messages and (iv) being - ideally - platform independent. Two alternatives were investigated, namely the Lightweight Communications and Marshalling (LCM) [90] and the Robot Operating System (ROS) [91].

8.1 Lightweight Communications and Marshalling

LCM is a set of libraries and tools for message passing and data marshalling, targeted at real-time systems [92]. It is based on UDP multicast communication for high-bandwidth and low-latency applications. LCM provides a publish/subscribe message passing model and automatic marshalling/unmarshalling code generation with bindings for applications in a variety of programming languages and platforms. The primary goal of LCM is to simplify the development of low-latency message passing systems, especially for real-time robotics research applications.

Its main features include:

- A publish-subscribe messaging system that uses User Datagram Protocol (UDP) multicast as its underlying transport layer.
- A type specification language that supports many platforms (GNU/Linux, OS X, Windows, Any POSIX-1.2001 system) and languages (C, C++, C#, Java, Lua, MATLAB, Python).
- Automatic marshalling/unmarshalling code generation for the abovementioned languages.
- Command line and graphical tools for logging, replaying, and inspecting traffic.

8.2 Robot Operating System

The ROS framework represents a suite of tools, libraries, and conventions whose goal is to simplify the task of creating complex and robust robot behaviour for different robotic platforms.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Visualisation tools are already available in ROS, where a suite of graphical tools that allow the easy recording and visualisation of data captured by the sensors, and represent the status of the vehicle in a comprehensive manner were created. Also, it provides a simple way to create additional visualisations required for particular needs. This is useful when developing the control software and trying to debug the code [93].

The ROS framework is language independent; it is easy to implement it in any modern programming language (Python, C++, and Lisp, Matlab, and there are experimental libraries in Java and Lua). ROS can be implemented also on several operating systems - ROS 2 is currently being CI tested and supported on Ubuntu Xenial, OS X El Capitan as well as Windows 10 [94].

8.3 Software framework within interACT

It was decided that ROS will be used in interACT project as a common software framework, because of its multiple assets. ROS is an open source framework, which is widely used in the robotics research community [95]. Due to this, there exists a huge selection of existing software packages. In particular, there is a lot of software for autonomous cars already created. Autonomous cars require the creation of algorithms that are able to localise the robot using LiDARs or GNSS, plan paths along maps, avoid obstacles, process pointclouds or cameras data to extract information, etc.) [93]. Moreover, ROS shows stability and reliability from a very large user base [96].

Another important asset is that there are quick tests and integration of already available algorithms and software packages. ROS is scalable for large runtime systems and for large development processes.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

9. Conclusions

The aim of this deliverable was to provide the conceptual design, along with a high-level analysis of each of the CCPU components, which were defined during the system architecture creation phase.

Initially, we presented a description of operation both for the core CCPU components (Situation Matching, Interaction Planning, Trajectory Planning & Safety Layer modules) and for the CCPU enablers (Scenarios & Interactions Strategies digital catalogues). The resulted work, although in an abstract level at the moment, gives a basic understanding of every component's expected functionality.

Secondly, a notion on how these elements will be designed was given. The focus mainly, was to establish an implementation plan, as to how the elements can be modelled into software modules that will include all necessary procedures, for the system to be operational. This is basically the first step towards the realisation of the CCPU.

Finally, we showed the process of evaluation and decision making, regarding the application of various technologies, as a means of communication (interfacing) between CCPU's components.

The presented work will be used as a basis, in the next stage, for the technical development of WP3's modules, so that the CCPU can be formulated, as an entity.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

10. References

1. interACT D1.2 Requirements, system architecture and interfaces for software modules. (2017).
2. interACT D1.1 Definition of interACT use cases and scenarios. (2017).
3. Ontologies. Retrieved on April 17, 2018 from https://www.w3.org/standards/semanticweb/ontology.
4. Ebbinghaus H-D, Flum J, Thomas W (1994). *Mathematical Logic* (Springer-Verlag New York).
5. Horn A (1951). On sentences which are true of direct unions of algebras. *J Symb Log* 16(1):14–21.
6. Thrun S, Burgard W, Fox D (2005). *Probabilistic Robotics* (The MIT Press) doi:10.1145/504729.504754.
7. Bechhofer S, et al. (2004). OWL Web Ontology Language Reference. *W3C Recomm* 10(February):2006–1.
8. Horrocks I, et al. (2004). SWRL : A Semantic Web Rule Language Combining OWL and RuleML. *W3C Memb Submiss 21* (May 2004):1–20.
9. Ordóñez A, Eraso L, Ordóñez H, Merchan L (2016). Comparing Drools and Ontology Reasoning Approaches for Automated Monitoring in Telecommunication Processes. *Procedia Computer Science*, pp 353–360.
10. Lampa S (2010). SWI-Prolog as a Semantic Web Tool for semantic querying in Bioclipse: Integration and performance benchmarking.
11. Drools - Business Rules Management System. Retrieved on April 17, 2018 from https://www.drools.org/.
12. Fobel A, Subramanian N (2016). Comparison of the performance of Drools and Jena rule-based systems for event processing on the semantic web. *2016 IEEE/ACIS 14th International Conference on Software Engineering Research, Management and Applications, SERA 2016*, pp 24–30.
13. SWI-Prolog. Retrieved on April 17, 2018 from http://www.swi-prolog.org/.
14. Han S, Choi BS, Lee JM (2008). A precise curved motion planning for a differential driving mobile robot. *Mechatronics* 18(9):486–494.
15. Kunchev V, Jain L, Ivancevic V, Finn A (2006). Path Planning and Obstacle Avoidance for Autonomous Mobile Robots: A Review. *Knowledge-Based Intelligent Information and Engineering Systems* (Springer), pp 537–544.
16. Hwang YK, Ahuja N (1992). Gross motion planning---a survey. *ACM Comput Surv* 24(3):219–291.
17. Elbanhawi M, Simic M (2014). Sampling-Based Robot Motion Planning: A Review. *IEEE Access* 2:56–77.
18. Marchese FM (2006). Multiple mobile robots path-planning with MCA. *2006 International*

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Conference on Autonomic and Autonomous Systems, ICAS'06 doi:10.1109/ICAS.2006.38.

19. Gonzalez D, Perez J, Milanes V, Nashashibi F (2015). A Review of Motion Planning Techniques for Automated Vehicles. *Intell Transp Syst IEEE Trans* PP(99):1–11.

20. Geronimo D, Lopez AM, Sappa AD, Graf T (2009). Survey of Pedestrian Detection for Advanced Driver Assistance Systems. *IEEE Trans Pattern Anal Mach Intell*:1–21.

21. Ros G, Sappa AD, Ponsa D, Lopez AM (2012). Visual SLAM for Driverless Cars : A Brief Survey. *IEEE Intell Veh Symp Work*:1–6.

22. Geiger A, Ziegler J, Stiller C (2011). StereoScan: Dense 3d reconstruction in real-time. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp 963–968.

23. Liu P, Kurt A, Redmill K, Ozguner U (2016). Classification of Highway Lane Change Behavior to Detect Dangerous Cut-in Maneuvers. *Transp Res Board, 95th Annu Meet*:1–14.

24. LaValle SM, Hutchinson SA (1998). Optimal motion planning for multiple robots having independent goals. *IEEE Trans Robot Autom* 14(6):912–925.

25. LaValle SM (2006). *Planning Algorithms* doi:10.1017/CBO9780511546877.

26. Hwang JY, Kim JS, Lim SS, Park KH (2003). A Fast Path Planning by Path Graph Optimization. *IEEE Trans Syst Man, Cybern Part ASystems Humans* 33(1):121–128.

27. Kala R, Warwick K (2013). Multi-level planning for semi-autonomous vehicles in traffic scenarios based on separation maximization. *J Intell Robot Syst Theory Appl* 72(3–4):559–590.

28. Stentz A, Stentz A (1994). Optimal and efficient path planning for partially-knownenvironments. *Robot Autom 1994 Proceedings, 1994 IEEE Int Conf* 3(1):3310–3317.

29. Likhachev M, Ferguson D, Gordon G, Stentz A, Thrun S (2008). Anytime search in dynamic graphs. *Artif Intell* 172(14):1613–1643.

30. Nilsson NJ (1969). *A Mobile Automaton: An Application of Artificial Intelligence Techniques*.

31. Bast H, et al. (2015). *Route Planning in Transportation Networks* doi:10.1007/978-3-319-49487-6.

32. Koenig S, Likhachev M (2005). Fast replanning for navigation in unknown terrain. *IEEE Trans Robot* 21(3):354–363.

33. Likhachev M, Ferguson D, Gordon G, Stentz A, Thrun S (2005). Anytime dynamic A*: an anytime, replanning algorithm. *ICAPS'05 Proceedings of the Fifteenth International Conference on International Conference on Automated Planning and Scheduling*, pp 262–271.

34. Likhachev M, Stentz A (2009). Path Clearance. *Robot Autom Mag IEEE* 16(2):62–72.

35. Dijkstra EW (1959). A note on two problems in connexion with graphs. *Numer Math* 1(1):269–271.

36. Likhachev M, Ferguson D (2009). Planning long dynamically feasible maneuvers for autonomous vehicles. *Int J Rob Res* 28(8):933–945.

37. Pivtoraiko M, Kelly A (2005). Efficient constrained path planning via search in state lattices. *Eur*

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Sp Agency, (Special Publ ESA SP (603):249–255.

38. Kushleyev A, Likhachev M (2009). Time-bounded Lattice for Efficient Planning in Dynamic Environments Time-bounded Lattice for Efficient Planning in Dynamic Environments Time-bounded Lattice for Efficient Planning in Dynamic Environments. *Robot Autom*:1662–1668.

39. Rufli M, Siegwart R (2009). On the Application of the D* Search Algorithm to Time-Based Planning on Lattice Graphs. *Proc. of The 4th European Conference on Mobile Robotics (ECMR)*, pp 105–110.

40. Kavraki LE, Švestka P, Latombe JC, Overmars MH (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans Robot Autom* 12(4):566–580.

41. LaValle SM, Kuffner JJ (2001). Randomized kinodynamic planning. *Int J Rob Res* 20(5):378–400.

42. Jo K, et al. (2013). Overall reviews of autonomous vehicle a1 - System architecture and algorithms. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, pp 114–119.

43. Karaman S, Frazzoli E (2011). Sampling-based algorithms for optimal motion planning. *Int J Robot …* 30(7):846–894.

44. Karaman S, Frazzoli E (2010). Optimal kinodynamic motion planning using incremental sampling-based methods. *Proceedings of the IEEE Conference on Decision and Control*, pp 7681–7687.

45. Labakhua L, Nunes U, Rodrigues R, Leite FS (2008). Smooth trajectory planning for fully automated passengers vehicles: Spline and clothoid based methods and its simulation. *Lecture Notes in Electrical Engineering*, pp 169–182.

46. Horst JA, Barbera AJ (2006). Trajectory Generation for an On-Road Autonomous Vehicle. *Defense and Security Symposium. International Society for Optics and Photonics*.

47. Reeds JA, Shepp LA (1990). Optimal Paths for a Car that goes both Forwards and Backwards. *Pacific J Math* 145(2).

48. Fleury S, Soueres P, Laumond JP, Chatila R (1995). Primitives for Smoothing Mobile Robot Trajectories. *IEEE Trans Robot Autom* 11(3):441–448.

49. Velenis E, Tsiotras P, Lu J (2007). Aggressive Maneuvers on Loose Surfaces: Data Analysis and Input Parametrization. *Mediterranean Conference on Control & Automation*, pp 1–6.

50. Brezak M, Petrovic I (2014). Real-time approximation of clothoids with bounded error for path planning applications. *IEEE Trans Robot* 30(2):507–515.

51. Walton DJ, Meek DS (2005). A controlled clothoid spline. *Comput Graph* 29(3):353–363.

52. Piazzi A, Lo Bianco CG, Bertozzi M, Fascioli A, Broggi A (2002). Quintic G2-Splines for the Iterative Steering of Vision-Based Autonomous Vehicles. *IEEE Trans Intell Transp Syst* 3(1):27–36.

53. Glaser S, Vanholme B, Mammar S, Gruyer D, Nouvelière L (2010). Maneuver-based trajectory planning for highly autonomous vehicles on real road with traffic and driver interaction. *IEEE Trans Intell Transp Syst* 11(3):589–606.

54. Simon A, Becker JC (2009). Vehicle guidance for an autonomous vehicle. *International Conference on Intelligent Transportation Systems*, pp 429–434.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

55. Wang LZ, Miura KT, Nakamae E, Yamamoto T, Wang TJ (2001). An approximation approach of the clothoid curve defined in the interval [0, π/2] and its offset by free-form curves. *CAD Comput Aided Des* 33(14):1049–1058.

56. Rastelli JP, Lattarulo R, Nashashibi F (2014). Dynamic trajectory generation using continuous-curvature algorithms for door to door assistance vehicles. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp 510–515.

57. Choi JW, Curry R, Elkaim G (2008). Path planning based on bezier curve for autonomous ground vehicles. *Proc - Adv Electr Electron Eng - IAENG Spec Ed World Congr Eng Comput Sci 2008, WCECS 2008* (2):158–166.

58. Walton DJ, Meek DS, Ali JM (2003). Planar G2 transition curves composed of cubic Bézier spiral segments. *J Comput Appl Math* 157(2):453–476.

59. Bacha A, Bauman C, Faruque R, Fleming M, Terwelp C (2008). Odin: Team VictorTango's Entry in the DARPA Urban Challenge. *J F Robot 25(8),* 25(8):467–492.

60. Shiller Z, Gwo Y-R (1991). Dynamic motion planning of autonomous vehicles. *IEEE Trans Robot Autom* 7(2):241–249.

61. Berglund T, Brodnik A, Jonsson H, Staffanson M, Söderkvist I (2010). Planning smooth and obstacle-avoiding B-spline paths for autonomous mining vehicles. *IEEE Trans Autom Sci Eng* 7(1):167–172.

62. Havlak F, Campbell M (2014). Discrete and continuous, probabilistic anticipation for autonomous robots in Urban environments. *IEEE Trans Robot* 30(2):461–474.

63. Tran Q, Firl J (2013). Modelling of traffic situations at urban intersections with probabilistic non-parametric regression. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp 334–339.

64. Madrigal AC (2014). The Trick That Makes Google's Self-Driving Cars Work. *Atl Media Co*:1–11.

65. Verma R, Del Vecchio D (2011). Semiautonomous multivehicle safety. *IEEE Robotics and Automation Magazine*, pp 44–54.

66. Yong SZ, Zhu M, Frazzoli E (2014). Generalized innovation and inference algorithms for hidden mode switched linear stochastic systems with unknown inputs. *Proceedings of the IEEE Conference on Decision and Control*, pp 3388–3394.

67. Brechtel S, Gindele T, Dillmann R (2011). Probabilistic MDP-behavior planning for cars. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp 1537–1542.

68. Ulbrich S, Maurer M (2013). Probabilistic online POMDP decision making for lane changes in fully automated driving. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp 2063–2070.

69. Brechtel S, Gindele T, Dillmann R (2014). Probabilistic decision-making under uncertainty for autonomous driving using continuous POMDPs. *2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014*, pp 392–399.

70. Galceran E, Cunningham AG, Eustice RM, Olson E (2017). Multipolicy decision-making for autonomous driving via changepoint-based behavior prediction: Theory and experiment. *Auton*

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

Robots 41(6):1367–1382.

71. Bandyopadhyay T, et al. (2013). Intention-aware motion planning. *Springer Tracts in Advanced Robotics*, pp 475–491.

72. Paden B, Čáp M, Yong SZ, Yershov D, Frazzoli E (2016). A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Trans Intell Veh* 1(1):33–55.

73. Lefèvre S, Vasquez D, Laugier C (2014). A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH J* 1(1):1.

74. Althoff M, Stursberg O, Buss M (2008). Stochastic reachable sets of interacting traffic participants. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp 1086–1092.

75. Greene D, et al. (2008). A computationally-efficient collision early warning system for vehicles, pedestrians, and bicyclists. *15th World Congr Intell Transp Syst ITS Am Annu Meet 2008* 1:35–46.

76. Rizaldi A, Althoff M (2015). Formalising Traffic Rules for Accountability of Autonomous Vehicles. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp 1658–1665.

77. Althoff M, Magdici S (2016). Set-Based Prediction of Traffic Participants on Arbitrary Road Networks. *IEEE Trans Intell Veh* 1(2):187–202.

78. Camacho EF, Bordons Alba C (2007). *Model Predictive Control* (Springer-Verlag London). 2nd Ed.

79. Magdici S, Althoff M (2016). Fail-Safe Motion Planning of Autonomous Vehicles. *IEEE Conf Intell Transp Syst*:452–458.

80. Werling M, Liccardo D (2012). Automatic collision avoidance using model-predictive online optimization. *Proceedings of the IEEE Conference on Decision and Control*, pp 6309–6314.

81. Hattori Y, Ono E, Hosoe S (2006). Optimum vehicle trajectory control for obstacle avoidance problem. *IEEE/ASME Trans Mechatronics* 11(5):507–512.

82. Magdici S, Ye Z, Althoff M (2017). Determining the maximum time horizon for vehicles to safely follow a trajectory. *Proc. of the IEEE Conference on Intelligent Transportation Systems*, pp 1893–1899.

83. Vogel K (2003). A comparison of headway and time to collision as safety indicators. *Accid Anal Prev* 35(3):427–433.

84. Hillenbrand J, Spieker AM, Kroschel K (2006). A Multilevel Collision Mitigation Approach&mdash;Its Situation Assessment, Decision Making, and Performance Tradeoffs. *IEEE Trans Intell Transp Syst* 7(4):528–540.

85. Tamke A, Dang T, Breuel G (2011). A flexible method for criticality assessment in driver assistance systems. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp 697–702.

86. Sontges S, Althoff M (2017). Computing possible driving corridors for automated vehicles. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp 160–166.

87. Koschi M, Althoff M (2017). SPOT: A tool for set-based prediction of traffic participants. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp 1686–1693.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

88. Bender P, Ziegler J, Stiller C (2014). Lanelets: Efficient map representation for autonomous driving. *IEEE Intelligent Vehicles Symposium, Proceedings*, pp 420–425.

89. OpenDRIVE. Retrieved on April 17, 2018 from http://www.opendrive.org/.

90. LCM: Lightweight Communications and Marshalling. Retrieved on April 17, 2018 from https://lcm-proj.github.io/.

91. ROS.org. Retrieved on April 17, 2018 from http://www.ros.org/.

92. Huang AS, Olson E, Moore DC (2010). LCM: Lightweight Communications and Marshalling. *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, pp 4057–4062.

93. How to Start With Self-Driving Cars Using ROS. (2017). Retrieved on April 17, 2018 from http://www.theconstructsim.com/start-self-driving-cars-using-ros/.

94. Thomas D Changes between ROS 1 and ROS 2. Retrieved on April 17, 2018 from http://design.ros2.org/articles/changes.html.

95. Aeberhard M, et al. (2015). Automated Driving with ROS at BMW. *ROSCon* Retrieved from https://roscon.ros.org/2015/presentations/ROSCon-Automated-Driving.pdf.

96. Foote T (2016). Michael Aeberhard (BMW): Automated Driving with ROS at BMW. Retrieved on April 17, 2018 from http://www.ros.org/news/2016/05/michael-aeberhard-bmw-automated-driving-with-ros-at-bmw.html.

This report is part of interACT project that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no 723395.

For more information:

interACT Project Coordinator

Anna Schieben

DEUTSCHES ZENTRUM FUER LUFT - UND RAUMFAHRT e.V. (DLR)

Lilienthalplatz 7

38108 Braunschweig, Germany

Anna.Schieben@dlr.de

interact-roadautomation.eu/

Designing cooperative interaction of automated vehicles with other road users in mixed traffic environments