UP-Drive

Automated Urban Parking and Driving

Research and Innovation Action - Horizon 2020
H2020-ICT-24-2015: Robotics
Grant Agreement Number 688652

Project start date: January 1, 2016, Duration: 48 months

Deliverable D6.1

Software specification and architecture for scene understanding

Coordinating partner: Volkswagen A.G., Group Research

Coordinating person: Wojciech Derendarz

Lead contractor for this deliverable: ČVUT

Deliverable editor: Václav Hlaváč

Due date of deliverable: June 30, 2016

Date of submission: October 27, 2016

Dissemination level: Public
Deliverable type: Report

Executive Summary

This Deliverable 6.1 contributes to the *UP-Drive* (automated Urban Parking and Driving) project's endeavor to create a car capable of self-driving in an unconstrained urban environment with speeds up to 30 km/h. It states the requirements for WP6, which should provide the relevant scene/scenario understanding functionality.

The main purpose of the scene/scenario understanding module is to extract contextual information about other traffic participants, that are relevant to the particular traffic situation, from the environment.

Scene/scenario understanding in the *UP-Drive* project consists of several functionalities, such as *object validation & stabilization*, *object intention estimation*, and *object motion prediction.* The percepts/information come from different sensors as lidar, radar, cameras, GPS receivers, as well as the RoadGraph (a graphical structure representing the traffic infrastructure). They will be provided by WP4 (perception) and WP5 (lifelong localization & mapping). WP6 will contribute by mitigating the uncertainty before the information is passed on to WP7 (decision-making and navigation).

This document analyzes in more detail the scene/scenario understanding functionalities and related requirements. Discussions with *UP-Drive* project partners clarified the partners' roles and their expected contributions. They also confirmed the initial *UP-Drive* software architecture and identified pending work tasks: Besides keeping learning from the work of others, the implementation work has to start. The first job will be to gain the ability to read sensory data and processed percepts from the *UP-Drive* test car. The second job is to create/use tools for making the data understandable by humans. The side effect of this analysis will be a list of several scenarios that need to be understood in order to handle particular traffic situations. Deliberating about the scenarios and practical experimenting with them will bring the needed insight into the scene/scenario understanding task. First implementations will be done on toy examples.

Contributing Authors/Partners

Partner	Author	Contribution
ČVUT	Václav Hlaváč	document editor
ČVUT	Júlia Kučerová	related work of others, possible approaches to scenario understanding
VW	Carsten Last	scenario understanding architecture, data types, and terminology

Revision Table

Revision	Date	Revision details	Contributors
1.0	October 27, 2016	Initial document submission.	V. Hlaváč, J. Kučerová, C. Last

Contents

1 Introduction

This Deliverable 6.1 contributes to the *UP-Drive* (automated Urban Parking and Driving) project's endeavor to create a car capable of self-driving in an unconstrained urban environment with speeds up to 30 km/h. The Work Package 6 (WP6) will deliver the required *scene understanding* functionality. We use the name *scenario understanding* interchangeably (see also Section 1.3).

Deliverable 6.1 elaborates the tasks given to WP6 in Deliverable 1.1 as the *UP-Drive* project's Description of Work specifies on page 42: *"In this report detailed input and outputs of scene understanding will be produced. Input will include the specific interfaces from other Work Packages. Outputs will define the format of the data produced to detect the current scene and predict the expected behavior of traffic occupants. Also the interfaces between the individual modules involved into scene understanding will be provided."*

The *UP-Drive* goals are specified in the Description of Work [12] and in the Deliverable 1.1 [9]. We will also shortly repeat them in Section 1.1. Afterwards, in Section 1.2 we will repeat the task formulation for the *scene understanding* work package, and in Section 1.3 we will provide a definition of the most relevant terms.

1.1 *UP-Drive* project context

The *UP-Drive* consortium is building up a demonstrator platform consisting of an automated car, a sensor-rich electric VW e-Golf, and a cloud environment to demonstrate automated transportation in urban environments. *UP-Drive* builds up on the know-how and the technology of the previous *V-Charge* project [34], in which the partners VW and ETHZ participated.

The *UP-Drive* scientific and technological scope is the automated parking and driving in urban environments with speeds up to 30 km/h. The functionality needed to deliver the required *UP-Drive* skills, which will enable to demonstrate the above use-cases, was decomposed into a top level system architecture as shown in Figure 1.

1.2 Task formulation for the scene understanding work package

Recall the interplay between the *UP-Drive* work packages depicted in Figure 1: WP6 (scene understanding) has to deliver to WP7 (navigation and decision making) the interpretation of the scene and predictions related to the scene. The data for deriving the semantic context will be provided by the Perception (WP4) as well as the lifelong localization & mapping (WP5) work packages. The goal of the scene understanding WP6 is to provide a sound understanding of the vehicle surrounding enabling the computation of profound vehicle navigation decisions. The requirements are (see [9], Section 4.5):

- Combine the results from the digital map with those from the perception subsystem.
- Detect and handle inconsistencies between the map data and the perception data.
- Estimate the intentions of other traffic participants (pedestrians, cyclists, and vehicles), e.g. whether a pedestrian intends to cross the street or whether a vehicle is likely to drive straight, make a left/right turn, make a u-turn, yield, park, etc.

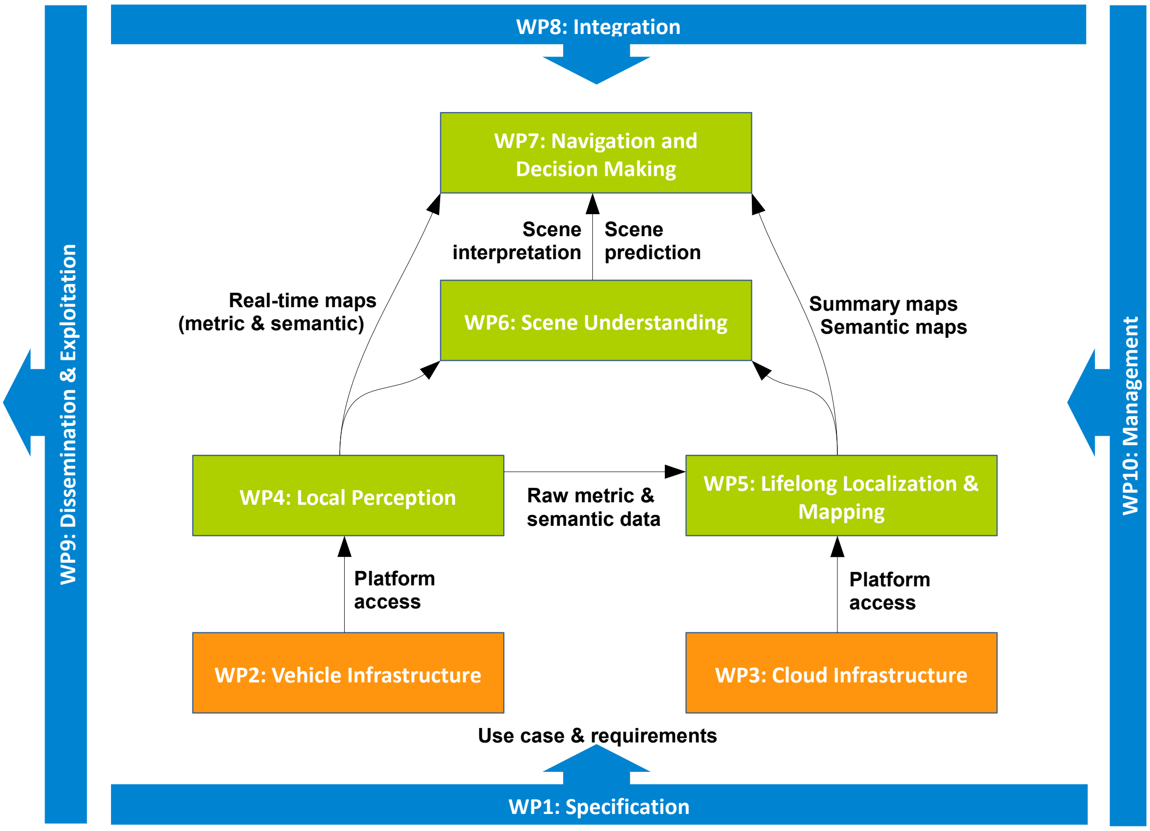
WP8: Integration
WP7: Navigation and Decision Making
Scene interpretation
Scene prediction
Real-time maps (metric & semantic)
WP6: Scene Understanding
Summary maps Semantic maps
WP9: Dissemination & Exploitation
WP10: Management
WP4: Local Perception
Raw metric & semantic data
WP5: Lifelong Localization & Mapping
Platform access
Platform access
WP2: Vehicle Infrastructure
WP3: Cloud Infrastructure
Use case & requirements
WP1: Specification

Figure 1: Work Package overview with key interfaces between the individual work packages.

- Provide stable motion predictions (future trajectories) for the other traffic participants (pedestrians, cyclists, and vehicles).
- Distinguish static from dynamic objects.
- Keep track of the past motion state of a now standing vehicle.
- Keep track of the past motion state of an occluded vehicle, even if it has not been perceived by the sensors for a certain amount of time.

The *scene understanding* (WP6) functionality serves as the intermediate layer between *perception* (WP4) and *navigation and decision making* (WP7). Scene understanding (WP6) has a slightly longer time slot at hand than perception (WP4). WP6 time slot is shorter than that of navigation and decision making (WP7). On the other hand, scene understanding (WP6) can utilize a little more semantic context than perception (WP4). Navigation and decision making (WP7) has the richest semantics in *UP-Drive*.

The perception (WP4) detects/recognizes objects, their identity and their other attributes. The identity constitutes a much simplified instance of the symbol grounding problem [18], [19]. In spite of this simplification, the detection/recognition process is error-prone. Hence, the scene understanding subsystem is required to validate and stabilize the detected objects as it has access to a broader context including partial semantic knowledge. This semantic knowledge is also needed to predict the intentions/actions of active agents in the scene, like other cars, pedestrians, and cyclists. The probabilistic reasoning is our tool of choice in *UP-Drive*. Such semantically-enhanced objects will contribute to improved navigation decisions and they will also likely help in the validation and stabilization of newly perceived objects.

These semantically-enhanced objects are the main foreseen outputs of the scene understanding subsystem. However, it has to be investigated throughout the project, to which extend additional situation-specific reaction skills will be needed.

1.3 Terminology

The correct definition of the key terms is crucial for the understanding and proper use of these terms in the *UP-Drive* project. We follow the terminology summarized in [7], [11], and [38]. A quick overview of relevant scene understanding terms is provided by Figure 2. From the literature follows that the name 'scenario understanding' probably covers the scope of WP6 better than the term 'scene understanding', which originates from the Description of Work [12]. Other terminologies can be found e.g. in [21]. A definition of the most relevant terms from the *UP-Drive* terminology is provided below.

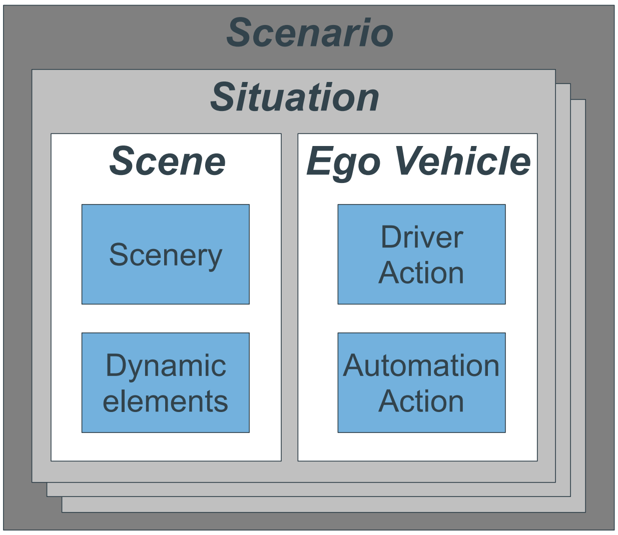

Figure 2: Definition of relevant scenario understanding concepts (according to [7].)

1.3.1 Scene understanding terms

Ego vehicle

The own (automated) vehicle is called ego vehicle.

Scenery

A scenery is a collection of static elements that form the surrounding for the scene. Items of a scenery are e.g. roads with varying lane types and numbers (such as motorways, rural roads, or crossroads), traffic signs, traffic lights, static obstacles (e.g. construction zones), building, and trees [7].

Scene

A scene is defined by a scenery and dynamic elements. Dynamic elements are e.g. other traffic participants, the state of traffic lights, or different light and weather conditions. The ego vehicle is not part of a scene. [7].

Situation

A situation consists of a scene together with an associated action of the ego vehicle. Depending on the action, the same scene can evolve into different situations where different objects become relevant [7]. The difference between the scene and the situation is demonstrated by the example in Figure 3 where the maneuver of the automated blue car can be different:

- pass the intersection straight - bike is not relevant for the situation,
- right turn – bike is relevant for the situation.

The scene representation needs to contain the bike at all times as it is independent of the automated vehicle's goals and actions.

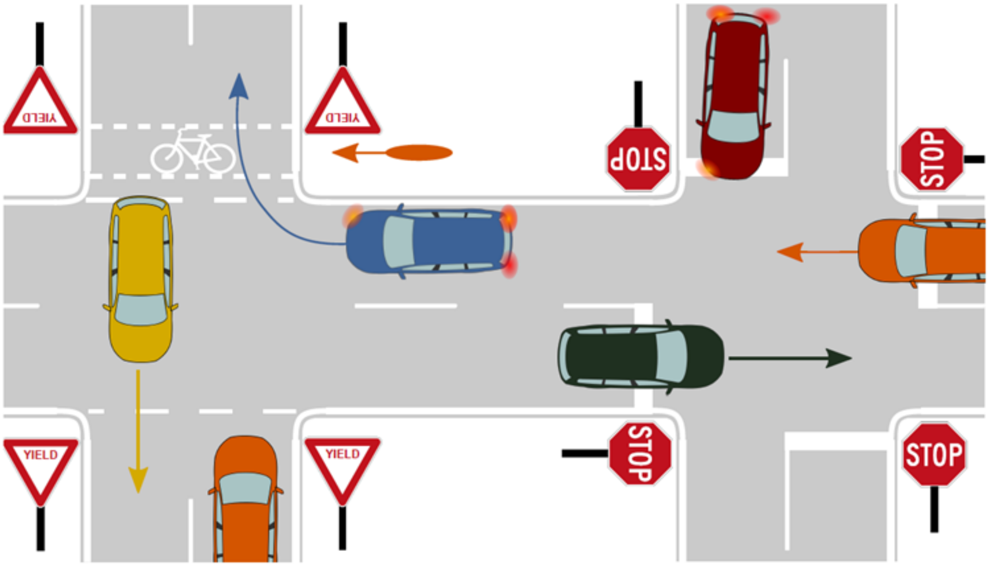

Figure 3: Illustration of a situation (taken from [11]). Automated vehicle is shown in blue.

Scenario

A scenario includes at least one situation [7]. For example "a cross-way scenario, where the driver is supposed to turn right, might have just one or perhaps a number of different situations. The ego-vehicle might decide to keep the velocity while turning right or it might decelerate, accelerate, decelerate, and finally turn right. The first course of events consists of only one situation, whereas the second one consists of four situations." [7]

1.3.2 Navigation and decision making terms

Route

A route is a sequence of edges in a graph representing the topology of the road-network. It leads the ego vehicle from a point on the starting edge A to a point on the target edge B.

Maneuver

A maneuver is a high-level characterization of the motion of the vehicle, regarding the position and speed of the vehicle on the road. Examples of maneuvers are "going straight", "turning", or "overtaking".

Path

A path is a geometric curve that connects two points C and D in the two-dimensional Cartesian space. It can e.g. be used to refine the definition of a certain maneuver. However, it does not contain any timing information.

Trajectory

A trajectory is a path with additional timing information.

Route planning

The route planning tries to find the ”best” route from point A to point B based on some kind of cost function, e.g. the time-to-arrival, distance, mileage, or percentage of automation.

Maneuver planning

The maneuver planning addresses the problem of taking the best high-level decision for the vehicle, taking into account the route that originates from route planning [21].

Path planning

The path-planning is the problem of finding an arbitrarily parameterized geometric curve that connects two points C and D in the two-dimensional Cartesian space with regard to some kind of cost function, e.g. distance or displacement from obstacles. Secondary conditions, like smoothness of the curve, can help to restrict the solution space.

Trajectory planning

The trajectory planning is concerned with finding a time-parameterized geometric curve γ that connects two points C and D in the two-dimensional Cartesian space, based on some kind of cost function, e.g. the distance or the displacement from obstacles. The time parameter defines where on the curve the ego vehicle is located at a certain point in time. Secondary conditions – like minimal velocity ($\dot{\gamma}$), acceleration ($\ddot{\gamma}$), or jerk ($\dddot{\gamma}$) – can help to restrict the solution space. Especially the minimal jerk is an important criterion of ”good” trajectories for automated cars [39].

Note that the curve that results from path planning can be used as a ”reference” trajectory for the trajectory planning step in order to simplify the problem. For example, one may allow only lateral offsets from the reference curve and thus reduce the problem dimension.

1.3.3 Other terms

Parking spot

A parking spot is a well-defined geometric area, designated to park exactly one vehicle.

Parking lot

A parking lot is a certain area containing more parking spots than one.

RoadGraph

A graphical structure representing the topology of the road network, including positions of traffic lights, zebra crossings, etc.

2 Related work

The research in the area of automated driving is currently very popular in academia and also in industrial research. Automated driving has a lot of advantages such as improving safety, reducing congestion, lower emissions and greater mobility [21]. However, automated vehicles must interact with human-driven vehicles. Because of this, the correct estimation of the expected behaviors of other traffic participants is essential.

The automated car consists of two main parts: hardware and software. The hardware is based on a combination of different sensors such as lidar, radar, different types of cameras. The software part is focusing on processing the inputs from the sensors. It is very important to process and evaluate the data received from the hardware correctly. The crucial parts of the software structure are perception, fusion of information from various sensors, planning the path, maneuver, trajectory of the ego vehicle, and the correct intention estimation and motion prediction of other traffic participants such as pedestrians or other cars.

There were enormous advances in the research of automated driving in the last two decades. One of the first attempts in the area of automated driving was the Eureka PROMETHEUS project in the 1990s which dealt with automated lane keeping and cruise control. In this project, the tour from Munich, Germany to Odense, Denmark was made with 95% of automated lane keeping [5]. In a similar project in the USA called "No hands across America", the tour was completed with 98% of automated lane keeping [33].

The Defense Advanced Research Projects Agency (DARPA) hosted the Grand Challenge in 2004 and 2005 and the Urban Challenge in 2007. The Grand Challenge focused on off-road automated driving, where different research groups (e.g. [10]) competed with each other in order to complete a 212 km long off-road route. The DARPA Urban Challenge focused on an urban environment with moving traffic, where intersection and interactions with other vehicles have to be dealt with (see e.g. [4]).

A recent international event in the field of cooperative driving – the Grand Cooperative Driving Challenge 2016 [14] – was held in the Netherlands. In this challenge, ten student teams from six European countries competed for the best performance in cooperative and automated driving. In this challenge, different trials of automated driving were tested, such as merging two rows of vehicles into one, passing a crossing, turning at an intersection, and giving way to an emergency vehicle [14].

2.1 Approaches from a broader automated driving perspective

A lot of different automated cars were designed for the DARPA challenges. Because of the divergent hardware and software approaches, there exist a large variety of solutions towards automated driving. Despite the fact that there exist so many different approaches, there are still many unanswered questions and challenges such as interaction between an automated car and a human-driven car.

In the DARPA Urban challenge, the Tartan Racing team with their car named "Boss" [4] was the winner of the challenge. The hardware architecture consisted of several lidar and radar sensors. The crucial part of their system is the perception and the world modeling, which consist of three main parts: sensor layer, fusion layer and situation assessment layer (Figure 4). In the sensor layer, a list of individual sensors is maintained.

The measurements from each sensor are associated with existing tracked objects. It is very important to correctly stabilize the data from the sensors, i.e. measurements that do not associate with existing objects (traffic participants) will be added into a new object. In the fusion level, each sensor list of associated measurements is added into one global list of tracked objects. The situation assessment layer attempts to estimate the intentions of other traffic participants by integrating the estimates with the knowledge about the road infrastructure.

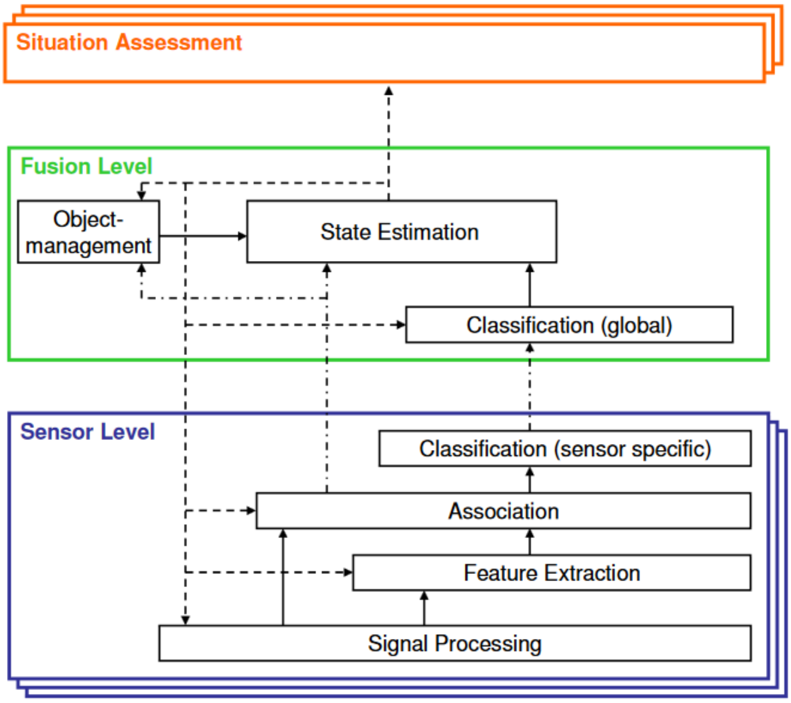

Figure 4: The moving obstacle fusion architecture of the "Boss" car (taken from [4]).

The sensor hardware of the "Junior" automated car, presented by the Stanford University team [6] at the DARPA Urban challenge, consists of five laser rangefinders, an Applanix GPS-aided internal navigation system, five radars, and two computers. The software architecture consists of sensor interfaces, perception modules, navigation modules, a drive-by-wire interface and global services. The behavior hierarchy of this automated car uses a finite state machine, which is used to switch between different driving states such as U-turns and intersections.

The automated car of team AnnieWAY [8] consists of stereo vision cameras, lidar, computers for sensor signal processing, situation assessment, and behavior generation, as well as computers for low level control. In this approach, a human-like understanding of the surrounding world is a key element. The sensor data is transformed into an abstract and holistic scene representation and probabilistic reasoning algorithms are applied subsequently which yield the recommended behavior for the given situation and traffic regulations.

2.2 Object verification & stabilization

One of the steps in scenario understanding for automated driving is verification and stabilization of the objects in the current scene. The verification and stabilization step lies in the fusion of the inputs from sensors and the correct processing of these data.

There exist different input sensors for automated cars such as lidar, radar, or camera. Each of these sensors has different characteristics of it's output data. Some generate raw data, other deliver pre-processed data. The fusion of the input data is responsible for processing the data from different sources. During this fusion, a lot of information about objects in the scene needs to be processed. Some of these information can be noisy and inaccurate due to the inaccurate input from sensors. Because of this, proper object verification and stabilization is very important.

2.3 Motion prediction and intention estimation

One of the most challenging tasks in automated driving is the detection of dangerous situations and the appropriate reaction to avoid accidents. It is also very important to predict the most likely evolution of the current traffic situation.

Authors of the survey [27] were focusing on the motion prediction and the risk assessment in traffic situations. Based on their research, there are three main groups of motion prediction models:

- *Physics-based* motion models
- *Maneuver-based* motion models
- *Interaction-aware* motion models

2.3.1 Physics-based motion models

Physics-based motion models (e.g. [1], [3], [20], or [29]) are currently the most used motion models for trajectory prediction and collision risk estimation. However, these models rely on the low level properties of motion (dynamic and kinematic properties), therefore they are limited to short-term (less than a second) motion prediction. Usually these models are unable to anticipate any change in the motion of the car caused by the execution of a particular maneuver, or changes caused by external factors (e.g. slowing down because of a pedestrian).

2.3.2 Maneuver-based motion models

Maneuver-based motion models (e.g. [25], [31], or [36]) are more advanced than physics-based models. They consider that the future motion of a vehicle also depends on the maneuver that the driver intends to perform. They represent vehicles as independent maneuvering entities, i.e. they assume that the motion of a vehicle on the road network corresponds to a series of maneuvers executed independently from the other vehicles.

Trajectory prediction with maneuver-based motion models relies on the early recognition of the maneuvers that drivers intend to perform. If we can identify the maneuver intention of the driver, we can assume that the future motion of the vehicle will match that maneuver. Based on this information, more relevant trajectories can be derived. They can be more reliable in the long term than the ones derived from physics-based motion models.

Maneuver-based motion models are based on prototype trajectories or based on maneuver intention estimation.

For the trajectory prototypes, the idea is that the trajectories of vehicles on the road network can be grouped into a finite set of clusters, where each cluster corresponds to a typical motion pattern. These trajectories can be learned from previously observed samples or from a digital map including information about the traffic infrastructure. The trajectory prediction is based on the comparison of partial trajectories (executed so far) with learned patterns. For example, in the case of Gaussian processes, the distance is computed as the probability that the partial trajectory corresponds to the Gaussian process. However, handling more subtle variations in velocity (i.e. variations caused by heavy traffic) is still an issue for such models. Another limitation is the hard adaptation to different road topologies, in particular when applied to road intersections.

The maneuver intention estimation consists of two steps: estimating the maneuver intention of the driver (e.g. waiting at the stop line) and then predicting the successive physical states so that they correspond to a possible execution of the identified maneuver. A major advantage over trajectory prototypes is that there is no need to match the partial trajectory with a previously observed trajectory. Instead, higher-level characteristics are extracted and used to recognize maneuvers, which makes it easier to generalize the learned model to arbitrary layouts.

The main limitation is the fact that these models assume vehicles moving independently from each other. Inter-vehicle dependencies are particularly strong at road intersections, where priority rules force vehicles to take into account the maneuvers performed by the other vehicles.

2.3.3 Interaction-aware motion models

Interaction-aware motion models consider the inter-dependencies between maneuvers of multiple vehicles, i.e. the motion of a vehicle is assumed to be influenced by the motion of the other vehicles in the scene. This contributes to a better understanding of the situation. These models are based either on prototype trajectories or on Dynamic Bayesian Networks (DBN).

Models based on trajectory prototypes

For these methods, inter-vehicle influences cannot be taken into account during the learning phase because of a high resulting number of motion patterns. However, it is possible to take into account the mutual influences during the matching phase by assuming that drivers have a strong tendency to avoid collisions when they can (e.g. [23]). Pairs of trajectories which lead to an unavoidable collision are penalized in the matching process, and as a result safe trajectories are always considered to be more likely than hazardous ones. This approach is an elegant workaround for taking into account inter-dependencies when using trajectory prototypes. However, the issue of modeling other types of influences remains, since the influence of one vehicle on the trajectory of another cannot be modeled directly.

Models based on Dynamic Bayesian Networks

Most *interaction-aware* motion models are based on Dynamic Bayesian Networks (DBN). Pairwise dependencies between multiple moving entities can be modeled with coupled Hidden Markov Models (CHMMs). This approach becomes quickly very complex due to the number of entities in the scene, because the complexity of CHMMs is not manageable in the context of complex traffic situations. The solution is to make CHMMs asymmetric by assuming that the surrounding traffic affects the vehicle of interest, but not vice versa. The assumption of asymmetric dependencies has been used in a number of works, in particular when dealing with lane change and overtaking maneuvers [37] or car following [30].

Several works have proposed general probabilistic frameworks for tracking vehicles and predicting their future motion. In [15], authors used mutual influences by using factored states. They modeled the causal dependencies between the vehicles as a function of the local situational context, which reduced the computational complexity.

In [28] and [26] authors model the joint motion of vehicles at road intersection. For this purpose, they introduced an intermediate variable called "Expected maneuver". The situational context influences what the driver is expected to do, which in turn influences what the driver intends to do.

The *interaction-aware* motion models allow longer-term predictions compared to *physics-based* motion models. They are also more reliable than *maneuver-based* motion models because they take into account the dependencies between the vehicles. It is very difficult to use *interaction-aware* models in real-time risk assessment, because the computation of the potential trajectories of the vehicles is computationally very expensive. For this reason, some risk assessment techniques have been proposed recently which do not rely on trajectory prediction.

2.3.4 Risk assessment

The motion models can be used to predict the future motion of vehicles. These predictions can be used to evaluate the risk of a situation. In the concept of intelligent vehicles, the risk assessment is associated with the idea that a situation may be dangerous for the driver, therefore it is natural to consider collisions as the main source of risk.

For risk assessment there are two main groups of approaches: risk based on colliding future trajectories and risk based on unexpected behavior. The risk based on colliding future trajectories can in turn be divided into three main groups: binary collision predictions, probabilistic collision prediction, and other risk indicators.

In risk assessment based on unexpected behavior, unusual events are detected: The risk of a situation can be estimated by defining the nominal behavior of vehicles on the road and detecting events which do not match that nominal behavior. Another approach lies in detecting conflicting maneuvers.

A number of works propose to assess the risk of a situation by estimating the maneuver intentions of the drivers and detecting potential conflicts between them or with the traffic laws. Since these approaches rely on estimated maneuver intentions, and since the concept of maneuver does not exist in physics-based motion models, vehicle motion is usually represented using maneuver-based or interaction-aware motion models.

2.4 Projects and industrial implementations

One of the most well-known automated cars, the Google self-driving car [17], is used to navigate safely through city streets. The sensors – such as lasers, radars, and cameras – are designed for object detection in all directions. They can detect e.g. pedestrians, cyclists, and vehicles but also fluttering plastic shopping bags and birds. The Google self-driving car processes map and sensor information to determine the position of itself in the world. The detection of the surrounding objects is based on their size, shape, and movement pattern. A safe speed and trajectory of the car is chosen based on the intention estimation of other traffic participants. The Google self-driving cars have driven more than 2 million kilometers in the traffic and are currently out on the streets of several cities in the USA.

The Navya ARMA [32] is an electric automated transport vehicle. It is equipped with state of the art sensors capable of communicating between themselves and fusing their data to refine the decision making of the vehicle. They include e.g. LIDAR sensors, RTK-GPS, and stereo vision cameras. The perception layer of this automated vehicle enables the understanding of the environment in which the vehicle is located by detecting obstacles and anticipating the displacements. The decision making layer computes and determines its itinerary and navigation applies and follows the most optimal route computed for the vehicle.

3 Scenario understanding architecture, interchanged data, and use cases

The requirements for the scenario understanding work package have been discussed in Section 1. Here, we present our first draft of a scenario understanding architecture that is able to deliver the desired outputs. The individual architecture components will be briefly described in Section 3.1. Afterwards, in Section 3.2, we will describe the data formats that are interchanged between the individual scenario understanding layers. Finally, in Section 3.3, we will provide some use cases that illustrate the proposed scenario understanding architecture.

3.1 Scenario understanding architecture

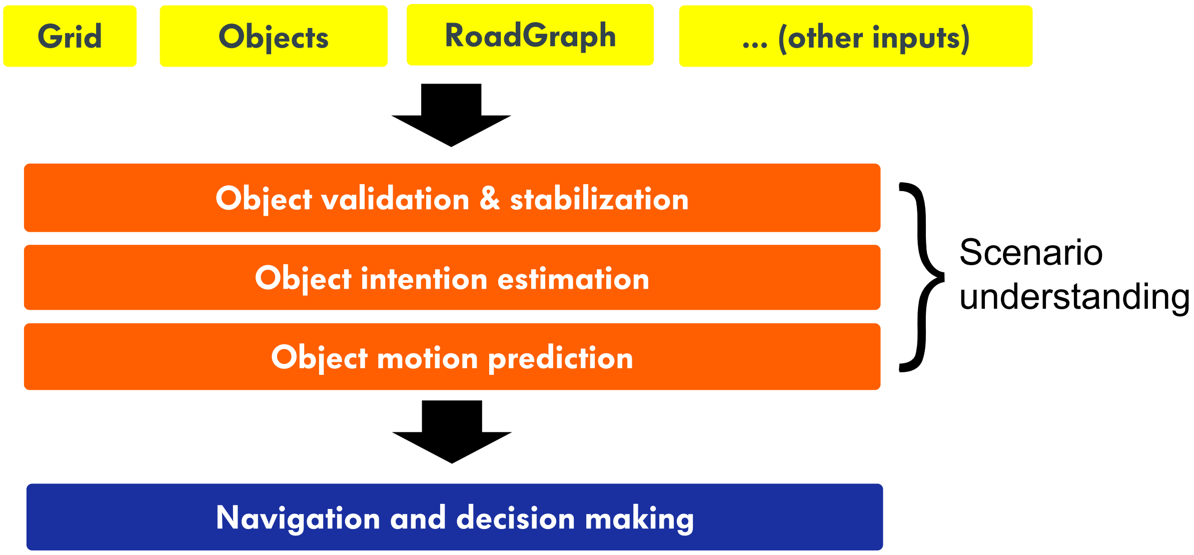
Grid
Objects
RoadGraph
... (other inputs)
Object validation & stabilization
Object intention estimation
Object motion prediction
Scenario understanding
Navigation and decision making

Figure 5: Scenario understanding architecture.

Our proposed scenario understanding architecture is shown in Figure 5. The input is provided by WP4 (Local Perception) and WP5 (Lifelong Localization & Mapping). The expected inputs are:

- Objects
  Physical objects like cars, pedestrians, etc. that have been identified by the local perception.
- Occupancy grid map
  Other (unidentified) objects, represented as a digital image centered around the ego vehicle. The pixel values represent occupied, unknown, or free areas.
- RoadGraph
  A graphical structure, representing the topology of the road network, including positions of traffic lights, zebra crossings, etc.
- Other inputs
  Other inputs from the Local Perception that have to be defined in the course of the project, e.g. detection of the turning-angle of a pedestrian's head.

These inputs have varying importance in each layer of our scenario understanding architecture. The first layer of our architecture is the *object validation & stabilization* layer.

The validation part of this layer verifies that a detected object can be interpreted as indicated by the perception modules. For this purpose, additional inputs (as e.g. raw image data) may be used. The stabilization ensures that objects are continuously tracked over a certain time-frame even if the input from the perception modules is only sporadic/erratic. This will be possible using higher level information about the traffic structure – provided by the RoadGraph – and other additional information.

One exemplary scenario for the object validation and stabilization module may be a group of motorcycles, which are first (wrongly) classified as a big truck (because of limited sensor resolution and corresponding motion patterns) and then split into several objects. The *object validation & stabilization module* will be able to understand such a situation – using statistical models and making assumptions like objects do not just disappear – so that the correct objects are passed to the following layers.

Having modeled the objects in the ego vehicle's environment as detailed as possible (desired goal: simulation-like objects), the *object intention estimation* layer's task is to provide an intention estimate for each object in the scene by using the stored object history, object dynamics, infrastructure information (as e.g. zebra crossings, two lanes narrowing down to one lane, traffic lights, etc.), information about other objects in the vicinity together with their dynamics, and other sensor inputs (as e.g. a blinking turn indicator detection module for cars or a head orientation detection module for pedestrians). For this purpose, it takes into account the outcome of the *object validation & stabilization* layer, additional information provided by various other perception modules, and information about the traffic infrastructure (provided as a road graph). Each object can be attributed with more than one intention, at best with estimated probabilities. Examples for intentions are, e.g. going straight, turning right, etc.

The *object prediction module* takes as input the output of the *object intention estimation module* plus similar inputs as that module. The task is to estimate one or more likely trajectories taking into account the expected intention(s). Information about the road infrastructure, relations, possible interplay among detected objects, and heuristics will also be taken into account.

The interpreted and predicted objects will serve as an input to the *navigation and decision making module* which will be dealt with in WP7. With these additional inputs, the automated vehicle will be able to cope with situations which require a semantic understanding of the scene. An illustrative example is to overtake a standing car ahead of the own car if there is a free lane to the left which requires the car ahead to be detected/understood as an isolated obstacle.

3.2 Interchanged data formats

The data structure that is able to store all the above-mentioned objects, intention estimates, and predictions is shown in Figure 6.

The main output of the scenario understanding work package will be the data stored in the class `PredictionOutputReadWriteLocked`. It is a read-write-locked specialization of the `PredictionOutput` base class that can be used in multi-threaded environments. At the same time, this class acts as an input to the scenario understanding work package from the local perception work package.

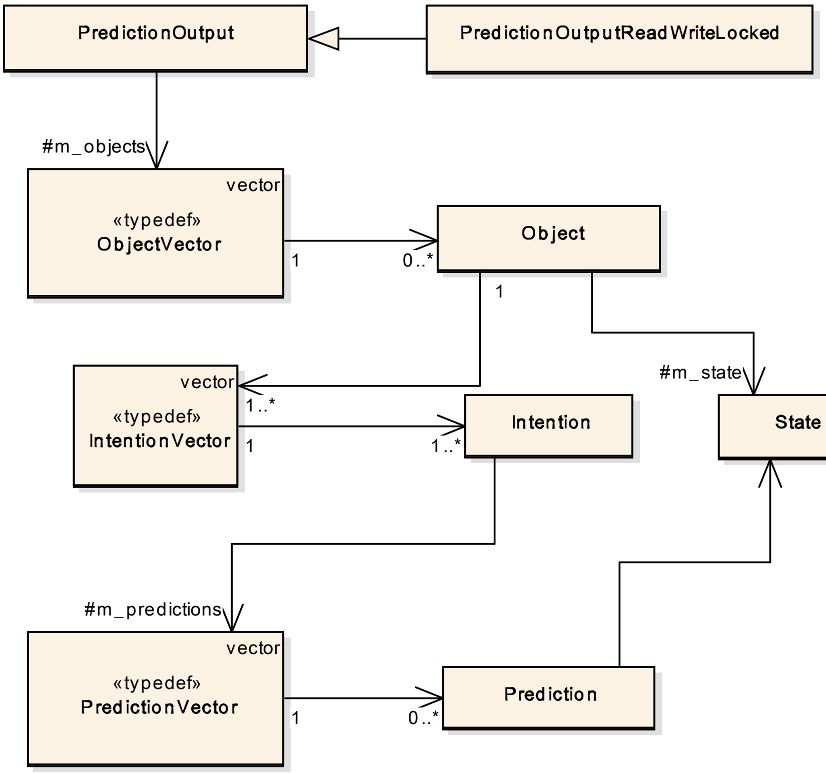

Figure 6: Scenario understanding data types.

The intended usage can be seen in Figure 7. It is as follows: The local perception work package stores all detected objects in the `PredictionOutput` class. Each detected object is represented as an instance of the `Object` class.

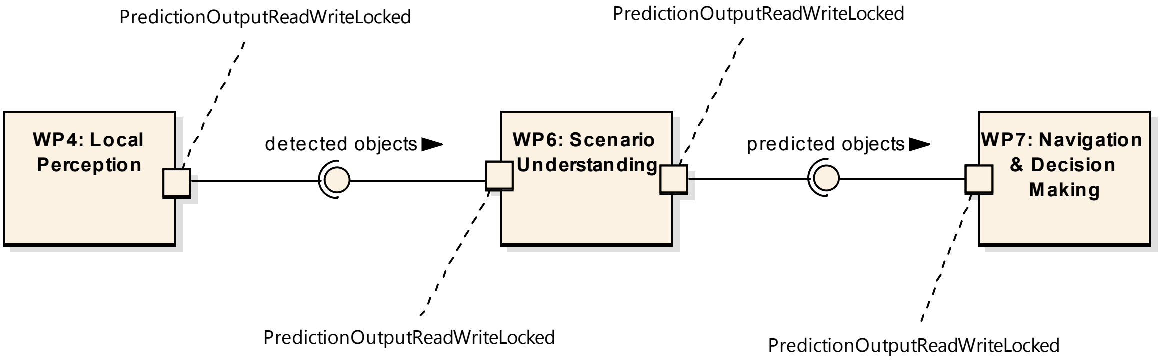
PredictionOutputReadWriteLocked
PredictionOutputReadWriteLocked
WP4: Local Perception
detected objects ►
WP6: Scenario Understanding
predicted objects ►
WP7: Navigation & Decision Making
PredictionOutputReadWriteLocked
PredictionOutputReadWriteLocked

Figure 7: Information flow of detected and predicted objects.

The *object intention estimation* layer of our scenario understanding architecture is going to enhance the detected objects with intention estimates. All possible object intentions are represented as a vector of intentions. This `IntentionVector` is supposed to be empty upon reception of the object list from the local perception work package as it is going to be filled by the scenario understanding work package.

These intention-enhanced objects are then passed on to the *object motion prediction* layer of our architecture, where each intention is translated into one or more possible predictions of the object's future state. They are going to be stored in a `PredictionVector`.

3.3 Use cases for the data structures

The intended use cases for the `PredictionOutputReadWriteLocked` data structure can be seen in figure 8:

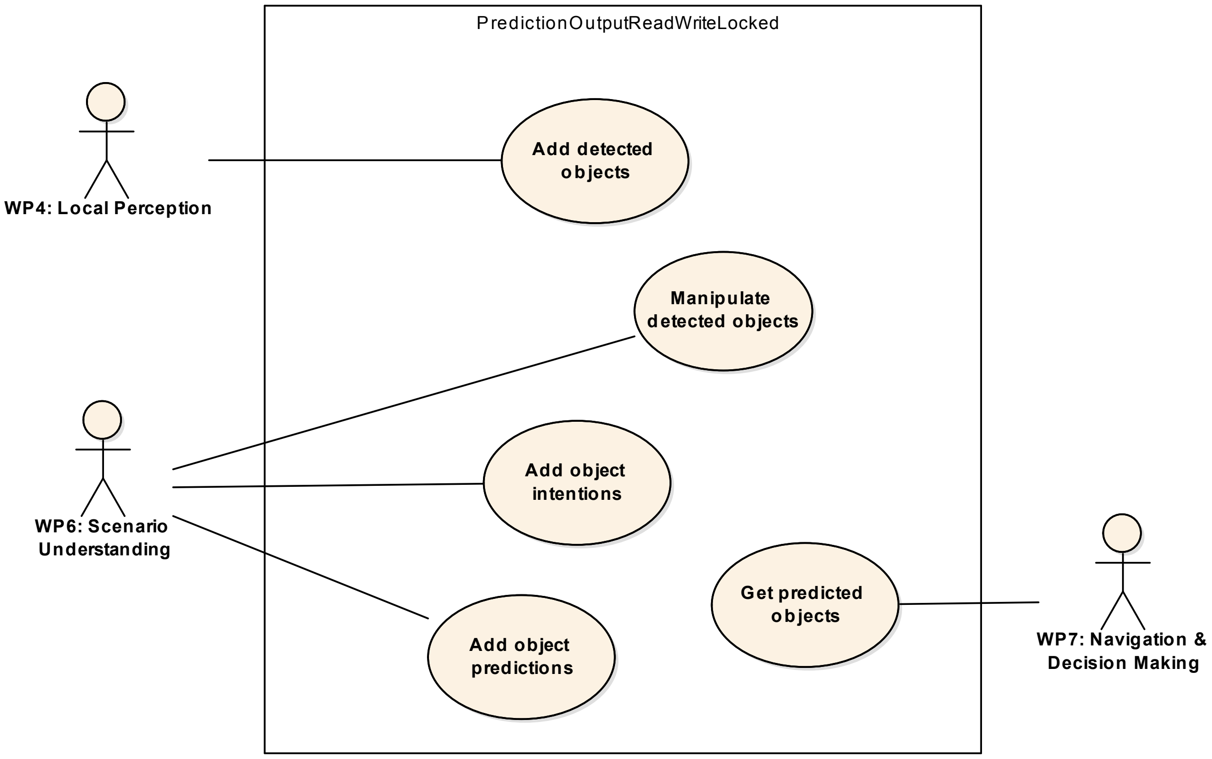
PredictionOutputReadWriteLocked
WP4: Local Perception
Add detected objects
Manipulate detected objects
WP6: Scenario Understanding
Add object intentions
Get predicted objects
WP7: Navigation & Decision Making
Add object predictions

Figure 8: Uses cases for the `PredictionOutputReadWriteLocked` data structure.

- The local perception work package shall store the detected objects in this data structure.
- The scenario understanding work package shall:
  - manipulate the detected objects in order to validate and stabilize them
  - add intentions to the detected objects
  - add predictions to each intention
- The navigation & decision making work package shall take the predicted objects as an input into the decision making process.

4 Possible approaches to scenario understanding

The automated car needs to be able to correctly evaluate the situations in a real-world driving environment. The same scene, in which the automated car is located, can evolve into different situations. These situations are influenced by the goal of the automated car, current traffic situation, and also by actions of other traffic participants.

Scenario understanding is a very challenging task due to the complex, uncertain, continuous, dynamic, and only partially observable traffic situations. The input data, received from the sensors, needs to be fused together with the digital map, where the object validation and stabilization play the most important role.

Another challenge is to correctly distinguish between static objects such as trees, buildings, traffic signs, or traffic lights and dynamic objects such as cars, cyclists, or pedestrians. After the correct classification of these objects, the estimation of their intention and prediction of their motion is crucial for the decision making of the automated car.

One of the main goals of the scenario understanding task is to deliver information about the surrounding – such as intentions, heading, and position of other traffic participants – and also contextual information of the whole situation to the decision making process. The contextual information and information about the intentions of other participants are very important due the different traffic situations. For example, the car ahead of the automated car may stop due to another car, or it may be planning to park (Figure 9). Each of these situation is different and due to the contextual information, the automated car needs to take necessary actions.

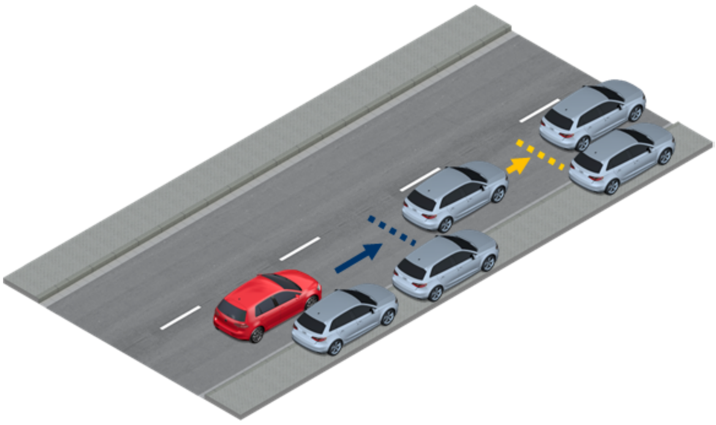

Figure 9: Example of a particular traffic situation in which the context is very important.

Scenario understanding in the *UP-Drive* project consists of several parts, as outlined in Figure 5. As an input, there is information from different sensors such as lidar, radar, and cameras, but also RoadGraph and GPS information. This information needs to be correctly processed for the further use in decision making and navigation of the automated car.

4.1 Object validation & stabilization

The validation and stabilization of the objects is an important step in order to correctly process the input data. The goal of this part is to combine the results from the digital map with results from the perception system. It should detect and handle inconsistencies between the map data and the perception data. This step can be executed in the *UP-Drive* WP4 (Perception) and also in *UP-Drive* WP6 (Scenario understanding), which depends on the input data.

The fusion of the input data lies in the combination of the input data into one common structure for each found object. Within the fusion of information some errors can occur, such as false detection of an object or detection of an incorrect number of objects in the scene. The validation and stabilization step in WP6 is focused on mitigating these problems.

One important sub-task in this layer is recognizing when an object becomes occluded. For this, it is important to have the information about blind spots of the car. Afterwards, using the list of tracked objects, the object's past positions and predicted future positions can be used to determine whether the object entered a blind spot.

4.1.1 Object validation & stabilization in the *UP-Drive* scope

All most commonly used approaches for object stabilization & validation come from the statistical domain. One possible approach can be to use conditional probabilities. In this case, the probability of an object class could be conditioned on measurements obtained by sensors and domain knowledge. The probability distribution could be set by an expert or trained from data. One such approach that models the world via conditional probabilities are Bayesian networks.

The object validation & stabilization step can be solved with a similar approach as for the intention estimation and motion prediction steps, which use probabilistic graphical models. In this case, the variable representing the class of an object would be a direct parent of the real measurements and inputs from the perception module (e.g. estimated object class). To validate the objects class, the additional information, such as object position and its relation to other parts of the scene (sidewalk, road lanes, . . .), as well as the measurements and object class estimate from previous time instances can be used. The Dynamic Bayesian Network (DBN) provides the most probable assignment to the query variables.

4.2 Object intention estimation and motion prediction

Estimating the intentions of other traffic participants (pedestrians, cyclists, and vehicles) is a very difficult task. As was described in Section 2.3, there are many different approaches to motion planning, intention estimation, and motion prediction. The intention estimation part is closely related to the following motion prediction. The estimation of the other traffic participants' behavior is usually based on observations of the surrounding context and previous motion patterns of the other traffic participants. In this section, we will describe some of the possible approaches for intention estimation and motion prediction of other traffic participants.

In [2], the authors proposed ontology-based context awareness for driving assistance systems. The authors used contextual information for the prediction of other traffic participants' behavior. They formulated an ontology about the vehicle, perceived entities, and context. The proposed ontology allows for a coherent understanding of the interactions between the perceived entities and contextual data. However, the approach considers only a one-dimensional driving space. Therefore, more research in this area is needed.

A framework for estimating a driver's decisions near intersections was presented in [13]. The authors suggested an architecture for describing the coupling of vehicle and driver through Hybrid-State-Systems (HSSs). In order to estimate the state of a vehicle, the authors used a framework which consists of HSSs and Hidden Markov Models (HMMs). This framework provides accurate results in comparison to the human observer.

An approach for learning continuous, non-linear, and context-dependent models for the behavior of other traffic participants was presented in [16]. The authors proposed a Bayesian model for the estimation and prediction of traffic situations, where the context-dependent policy model is used to predict the behavior of other traffic participants based on contextual information. The scheme of such a Bayesian model is presented in Figure 10, where solid arcs represent direct dependencies and dashed arcs represent temporal dependencies. An Expectation Maximization (EM) approach for learning the model from unlabeled observations was used. This model can cope better with noisy sensors and uphold a valid estimation even if the traffic participants are occluded for long periods of time. It allows for more precise long-term (up to 6 seconds) predictions without neglecting the uncertainty.

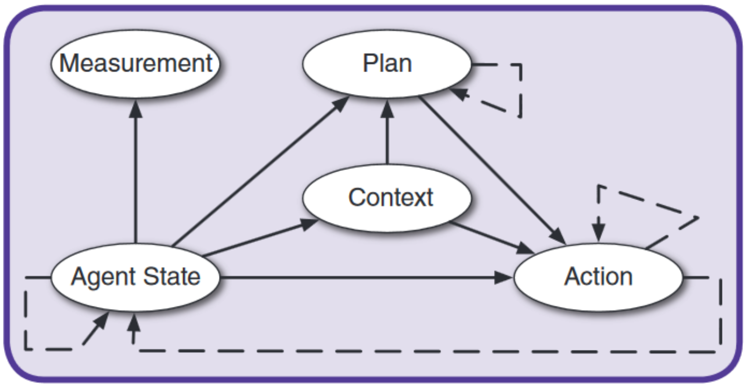

Figure 10: Scheme of the Bayesian model [16].

In [35], the authors proposed an intention-aware decision-making algorithm to solve decision-making problems in an uncontrolled intersection scenario. In order to consider uncertain intentions, the authors developed a continuous HMM to predict the motion intentions of other traffic participants. Consequently, a generative Partially Observable Markov Decision Process (POMDP) framework was built to model the automated decision-making process. However, there are difficulties in solving the POMDP. Therefore, the authors used proper assumptions to simplify this problem. They used a human-like policy generation mechanism to generate possible solution candidates. A model for predicting future motions of human-driven vehicles was proposed to be applied in the state transition process, and the intention is updated during each prediction time step. The authors also designed the reward function, which considers the driving safety, traffic laws, and time efficiency to calculate the optimal policy.

The project presented in [40] introduced a method for developing probabilistic driving models for traffic modeling and automotive safety systems. The authors used dynamic Bayesian networks for representing the distribution over acceleration and turn-rate for the next time step. Real-world driving data were used for learning the graph structure and parameters using existing Bayesian methods.

4.2.1 Object intention estimation and motion prediction in the *UP-Drive* scope

As was described in Section 2.3, there are three main approaches for motion prediction. Physics-based models can reliably predict the motion of other traffic participants for up to 1 second into the future. Nevertheless, they are currently the most widely used. Maneuver-based motion models are more complex, because intentions of other traffic participants are taken into account. However, they do not consider interactions between traffic participants. Interaction-aware motion models are the most complex models. They take into account the interactions between different traffic participants. Therefore, they are most suitable for this task. However, the complexity of these models is very high.

The most often used methods for intention estimation and motion prediction are graphical models such as Bayesian networks (e.g. [16], [24]) and Hidden Markov Models (e.g. [35],[13], [22]). In these models, the observed variables are the measured properties of the agents (such as their speed, heading, . . .) and the hidden variables are, besides the true values of their properties (as it is in case of Bayesian filtering), their actions and intentions. The probability distributions for the policy model (for actions and intentions) can be learned from real-world data, e.g. using the Expectation Maximization (EM) algorithm. Other distributions in the probabilistic network can be modeled either using physics-based models or expert knowledge (e.g. [16]). The model can then be used to estimate the agents' current actions and predict their intentions based on the current measurements.

Another approach in modeling the behavior of the agents are Finite-State Machines (FSM). In this case, it is assumed that the agents are controlled using a FSM where the individual states are the actions that the agent can perform. However, the actual state, in which the agent currently resides in, is hidden. Only the effects of the states, i.e. measurements, are observed, effectively forming an HMM. By learning the probability distributions for the measurements, the current actions of the agent can be estimated.

The open world probabilistic models are an extension of the probabilistic graphical models that allow changing the number of objects in the model dynamically. This is very important in this context since the number of traffic participants in any given scene is not known in advance.

5 Conclusions

This Deliverable 6.1 stated the initial requirement for the scene understanding workpackage (WP6). Based on the published works and VW's good practice, the outline of WP6 functionalities and related requirements was listed. WP6 will be closely interconnected with WP4, WP5, and WP7 in *UP-Drive*.

Besides material already presented in Deliverable 1.1, the conclusions of the scene understanding related workshop held aside of the *UP-Drive* Steering Committee Meeting on June 23, 2016 in Prague have been taken into account:

- Scene understanding refers to understanding the local vicinity of the car. The aim is to predict the future state of involved objects.
- Fixing the time range is difficult, for example when the car is stopped but keeps having the same intention.
- Test data from real drives is needed to understand the tasks in scenario understanding.
- There exist two approaches to prediction: (a) dynamic modeling; (b) from previous experience. Both approaches will be used.
- WP6 and WP7 are treated as decoupled. Nevertheless, predicting the future state in WP6 uses experienced data, which makes the connection between WP6 and WP7 explicit.

The next steps are the following:

- ČVUT team will start a toy implementation of possible approaches to scenario understanding.
- IBM will create an interface providing semantic information from the digital map (see Task 6.1 in the description of action).
- ČVUT and IBM will specify what queries might be useful.
- UTC will contribute to WP6.

References

[1] S. Ammoun and F. Nashashibi. Real time trajectory prediction for collision risk estimation between vehicles. In *Intelligent Computer Communication and Processing, 2009. ICCP 2009. IEEE 5th International Conference on*, pages 417–422, Aug 2009.

[2] A. Armand, D. Filliat, and J. Ibañez-Guzman. Ontology-based context awareness for driving assistance systems. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 227–233, June 2014.

[3] Mattias Brännström, Erik Coelingh, and Jonas Sjöberg. Model-based threat assessment for avoiding arbitrary vehicle collisions. *IEEE Trans. Intelligent Transportation Systems*, 11(3):658–669, 2010.

[4] Ch. Urmson et al. Tartan racing: A multi-modal approach to the DARPA urban challenge. Technical Report CMU-RI-TR-, Robotics Institute, Pittsburgh, PA, April 2007.

[5] J. Ziegler et al. Making bertha drive an autonomous journey on a historic route. *IEEE Intelligent Transportation Systems Magazine*, 6(2):8–20, Summer 2014.

[6] M. Montemerlo et al. Junior: The stanford entry in the urban challenge. In Martin Buehler, Karl Iagnemma, and Sanjiv Singh, editors, *The DARPA Urban Challenge*, volume 56 of *Springer Tracts in Advanced Robotics*, pages 91–123. Springer, 2009.

[7] S. Geyer et al. Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance. *IET Intelligent Transport Systems*, 8(3):183–189, May 2014.

[8] S. Kammel et al. Team AnnieWAY's autonomous system for the 2007 DARPA Urban Challenge. *J. Field Robot.*, 25(9):615–639, 2008.

[9] S. Nedevschi et al. Initial version of requirements definition, system architecture and component specification. Deliverable 1.1, Project UP-Drive consortium, May 2016.

[10] S. Thrun et al. Stanley: The robot that won the DARPA grand challenge: Research articles. *J. Robot. Syst.*, 23(9):661–692, September 2006.

[11] S. Ulbrich et al. Defining and substantiating the terms scene, situation, and scenario for automated driving. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 982–988, Sept 2015.

[12] W. Derendarz et al. Project UP-Drive, Description of Work, Technical Annex, December 2015.

[13] V. Gadepally, A. Krishnamurthy, and U. Ozguner. A framework for estimating driver decisions near intersections. *IEEE Transactions on Intelligent Transportation Systems*, 15(2):637–646, April 2014.

[14] "The Grand Cooperative Driving Challenge". `http://www.gcdc.net/en/`, 2016. Part of European research project i-GAME, Accessed: 2016-06-07.

[15] T. Gindele, S. Brechtel, and R. Dillmann. A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, pages 1625–1631, Sept 2010.

[16] T. Gindele, S. Brechtel, and R. Dillmann. Learning driver behavior models from traffic observations for decision making and planning. *IEEE Intelligent Transportation Systems Magazine*, 7(1):69–79, Spring 2015.

[17] "Google Self-Driving Car Project". `https://www.google.com/selfdrivingcar/`, 2016. Google Self-Driving Car Project, Accessed: 2016-06-10.

[18] S. Harnad. The symbol grounding problem. *Physica D*, 42:335–346, 1990.

[19] C. Hudelot, N. Maillot, and M. Thonnat. Symbol grounding for semantic image interpretation: From image data to semantics. In *Proceeding of the International Workshop on Semantic Knowledge in Computer Vision (in association with ICCV 2005)*, Beijing, China, October 2005. IEEE Computer Society, Los Alamitos, USA.

[20] N. Kaempchen, K. Weiss, M. Schaefer, and K. C. J. Dietmayer. Imm object tracking for high dynamic driving maneuvers. In *Intelligent Vehicles Symposium, 2004 IEEE*, pages 825–830, June 2004.

[21] Ch. Katrakazas, M. Quddus, W. Chen, and L. Deka. Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions. *Transportation Research Part C: Emerging Technologies*, 60:416 – 442, 2015.

[22] R. Kelley, A. Tavakkoli, Ch. King, M. Nicolescu, M. Nicolescu, and G. Bebis. Understanding human intentions via hidden markov models in autonomous mobile robots. In *Proceedings of the 3rd ACM/IEEE International Conference on Human Robot Interaction*, HRI '08, pages 367–374, New York, NY, USA, 2008. ACM.

[23] E. Käfer, C. Hermes, C. Wöhler, H. Ritter, and F. Kummert. Recognition of situation classes at road intersections. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3960–3965, May 2010.

[24] F. P. Kooij, J, N. Schneider, F. Flohr, and Dariu M. Gavrila. *Context-Based Pedestrian Path Prediction*, pages 618–633. Springer International Publishing, Cham, 2014.

[25] P. Kumar, M. Perrollaz, S. Lefèvre, and C. Laugier. Learning-based approach for online lane change intention prediction. In *Intelligent Vehicles Symposium (IV), 2013 IEEE*, pages 797–802, June 2013.

[26] S. Lefèvre, Ch. Laugier, and J. Ibañez-Guzmán. Intention-Aware Risk Estimation for General Traffic Situations, and Application to Intersection Safety. Research Report RR-8379, INRIA, October 2013.

[27] S. Lefèvre, D. Vasquez, and Ch. Laugier. A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH Journal*, 1(1):1–14, 2014.

[28] S. Lefèvre, C. Laugier, and J. Ibañez-Guzmán. Risk assessment at road intersections: Comparing intention and expectation. In *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pages 165–171, June 2012.

[29] Chiu-Feng Lin, A. G. Ulsoy, and D. J. LeBlanc. Vehicle dynamics and external disturbance estimation for vehicle path prediction. *IEEE Transactions on Control Systems Technology*, 8(3):508–518, May 2000.

[30] F. Klanner M. Liebner, M. Baumann and C. Stiller. Driver intent inference at urban intersections using the intelligent driver model. In *Proc. IEEE Intelligent Vehicles Symposium*, page 1162–1167, 2012.

[31] B. Morris, A. Doshi, and M. Trivedi. Lane change intent prediction for driver assistance: On-road design and evaluation. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 895–901, June 2011.

[32] "Navya ARMA". http://navya.tech/?lang=en, 2016. Navya ARMA, Accessed: 2016-06-16.

[33] D. Pomerleau and T. Jochem. Rapidly adapting machine vision for automated vehicle steering. *IEEE Expert*, 11(2):19–27, Apr 1996.

[34] Automated Valet Parking and Charging for e-Mobility (V-Charge). http://www.v-charge.eu/, 2011-2015. European Commission funded project, FP7. Accessed: 2016-06-02.

[35] W. Song, G. Xiong, and H. Chen. Intention-aware autonomous driving decision-making in an uncontrolled intersection. *Mathematical Problems in Engineering*, 25:775–807, 2016.

[36] A. Tamke, T. Dang, and G. Breuel. A flexible method for criticality assessment in driver assistance systems. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 697–702, June 2011.

[37] Ch. Tay. *Analysis of Dynamic Scenes: Application to Driving Assistance.* Theses, Institut National Polytechnique de Grenoble - INPG, September 2009.

[38] "UP-Drive project Glossary". https://git.up-drive.eu/documentation/wiki/wikis/glossary, 2016-2019. European Commission funded project, Accessed: 2016-06-06.

[39] M. Werling, S. Kammel, J. Ziegler, and L. Gröll. Optimal trajectories for time-critical street scenarios using discretized terminal manifolds. *The International Journal of Robotics Research*, 31(3):346—359, 2011.

[40] T. Wheeler. Probabilistic driving models and lane change prediction, 2014. http://cs229.stanford.edu/proj2014/Tim%20Wheeler,%20Probabilistic%20Driving%20Models%20and%20Lane%20Change%20Prediction.pdf.