D3.7

Concepts and algorithms incl. V&V results from 3rd cycle

Project Number:	690705
Classification **Deliverable No.:**	Public D3.7
Work Package(s): **Document Version:** **Issue Date:** **Document Timescale:** Start of the Document: Final version due:	WP3 Vs. 1.0 31.12.2018 Project Start Date: September 1, 2016 Month 26 Month 28
Deliverable Overview:	**Main document:** D3.7
Compiled by:	Stefan Suck
Authors:	Gurucharan Srinivas, DLR David Käthner, DLR Juan Pastor Trullos, DLR Mark Eilers, HMT Stefan Suck, OFF Maximilian Graf, ULM
Technical Approval:	Fabio Tango, CRF
Issue Authorisation:	Andreas Lüdtke, OFF

DISTRIBUTION LIST		
Copy type[1]	Company and Location	Recipient
T	AutoMate Consortium	all AutoMate Partners

RECORD OF REVISION		
Date	Status Description	Author
08.11.2018	Deliverable structure	Stefan Suck (OFF)
26.11.2018	DLR input for Enabler 5.1	Gurucharan Srivinivas (DLR); David Käthner (DLR)
07.12.2018	Included Input for E4.1	Maximilian Graf (ULM)
11.12.2018	Included Input for E4.2 and E5.1	Mark Eilers (HMT); Stefan Suck (OFF)
12.12.2018	Input from DLR for internal review	Gurucharan Srivinivas (DLR); David Käthner (DLR); Juan Pastor Trullos (DLR)
21.12.2018	Merge final input and reactions to reviewer comments, from ULM and DLR.	Stefan Suck (OFF)
22.12.2018	Add final content for Risk Assessment and finalize document	Mark Eilers (HMT); Stefan Suck (OFF)

[1] Copy types: E=Email, C=Controlled copy (paper), D=electronic copy on Disk or other medium, T=Team site (Sharepoint)

Table of Contents

List of Abbreviations

A2H .... *Automation to Human*
ACC .... *Accuracy*
DIR .... *Driver Intention Recognition*
EM .... *Expectation Maximization*
FPR .... *False Positive Rate*
IDM .... *Intelligent Driver Model*
IOU .... *Intersection Over Union*
KKT .... *Karush–Kuhn–Tucker*
PRE .... *Precision*
REC .... *Recall*
V&V .... *Verification and Validation*
WP .... *Work Package*

List of Figures

List of Tables

Executive Summary

This deliverable describes the current state of the enablers developed in WP3 (E4.1 - Planning and execution of safe manoeuvre, E4.2 - Learning of intention from driver, E5.1 - Online risk assessment) during the first half of the 3rd cycle to make them ready for the final integration into the demonstrator vehicles.

Section 2 deals with the improvements and changes of the enablers compared to the previous cycles. For E4.1 these are the usage of an intelligent driver model and concept changes that should lead to a much faster planning time, compared to the approach from cycle 2. For E4.2 the main changes are an improved method for online sample generation using forward-backward inference and the possibility to update Gaussian Mixture models distribution inside the Driver Intention Recognition model. For 5.1, there is now a functionality to assess a given trajectory, and the computational efficiency is improved. Additionally for each enabler a statement on possible data privacy issues is given.

Section 3 describes runtime performance, datasets and the experiments conducted to validate the enablers from a technical point of view, according to the validation plan, to the requirements and to the metrics described in D3.6. Additionally it gives a qualitative comparison with the State of the Art.

Section 4 concludes the document and provides for some enablers lessons learned concerning, e.g., dealing with a cold start problem.

1 Introduction

As described in D3.5 the enablers developed in WP3 contribute to the implementation of the “automation to the human” (A2H) direction of support of the AutoMate cooperation concept.

The activities in the Automate project are organized in 3 cycles to guarantee that the maturity of the technologies developed in the project is iteratively increased while assessing that the progresses are consistent with the needs of the demonstrators and, in turn, with the overall concept and objectives of the project.

In deliverable D3.2 “Catalogue of basic driving manoeuvres and associated task distributions”, we had laid out certain principles for driver task allocation. These principles were based on the available literature as well as the result of discussions among the consortium members, and should be understood as a documentation of the state of our discussion at that point in time. Instead, the framework, which assigns control to either the driver or the automation based on believes about their respective competence to handle a current driving situation, should be seen as the starting point for various further deliberations within the consortium, and within different work packages. Some of the framework’s branches suggested a mandatory shift of control to the automation in cases of an incapacitated driver. While such *Human-to-Automation* (*H2A*) shifts of control are being discussed within scenarios involving medical emergencies (e.g. Mirwaldt, Bartels & Lemmer, 2012), they are not the focus of the TeamMate car concept. Within the

AutoMate project, drivers will retain the possibility to overrule the system, in cases of a system malfunction such as sensor errors or even bugs in the software.

This deliverable describes the current state of the enablers developed in WP3 during the first half of the 3rd cycle, the improvements and changes compared to the previous cycles, as well as the experiments conducted to validate them from a technical point of view, according to the validation plan, to the requirements and to the metrics described in D3.6. In such a way, the new Enabler versions should ready for the final integration into the demonstrator vehicles.

The document is structured as follows. The status of the enablers is presented in Chapter 2 including the improvements and latest developments of them. Next, Chapter 3 describes the validation of enablers along with validation methodologies and the results. Finally, Chapter 4 concludes the document.

2 Status of WP3 enablers in the 3rd cycle

2.1 E4.1 - Planning and execution of safe manoeuvre (ULM)

The concept of the trajectory planner was changed in a way that social interaction is incorporated directly via driver models. This modification was made because some problems appeared concerning braking and holding a safety distance to a leading vehicle.

The trajectory planner developed is already integrated and tested in the Ulm vehicle. During the last consortium meeting in Oldenburg, there was WP3 workshop which was used to perform integration of some software modules into the software chain of the demonstrators. During the workshop, the trajectory planner was integrated into the VED architecture. The main problems that appeared are on the one hand the divergence in the used map format and on the other hand the fact that, in Ulm, a Linux operating system is used, while in Vedecom, Windows operating system is used. Nevertheless, with the support of Ulm, some major problems could be solved and VED can continue with the integration task but everything seems to go well so far.

There are some comments of the project officers and reviewers that were raised during the midterm review meeting in Brussels. One major comment is that there is a waste of resources since 2 trajectory planners are developed within the project, one by ULM and one by VED. In fact, the VED planner was not developed for AutoMate but was already existent and adapted for AutoMate. However, it is now attempted that ULM trajectory planner can also be used at VED.

At Ulm, there was already a trajectory planner available. However, there were some problems with computation times and sampling trajectories lateral to the centre line. Therefore, a new concept was developed in AutoMate. This concept has very low computation times and, even if the basic concept is designed for longitudinal traffic, there is enough flexibility to use it for overtaking as well, which is a basic requirement for Automate since overtaking is required within the PETER scenario.

2.1.1 Concept

The main change of the concept is the modification of the cost functional that is now composed as follows:

$$J(\boldsymbol{x}_0, \boldsymbol{x}_{-1}, \dots, \boldsymbol{x}_{N-1}) = \sum_{i=2} L\big(\boldsymbol{x}_i, \boldsymbol{x}_{dd,i}, \boldsymbol{x}_{ddd,i}\big)\Delta t + w_{spatial} \left|\left|\boldsymbol{x}_{ref,N-1} - \boldsymbol{x}_{N-1}\right|\right|_2^2 \Delta t$$

The Langrangian function L is composed as

$$L = w_{spatial}\, j_{spatial} + w_{acc} j_{acc} + w_{jerk} j_{jerk}.$$

With the spatial term

$$j_{spatial,i} = \left|\left|\boldsymbol{x}_{ref,i} - \boldsymbol{x}\right|\right|_2^2$$

The acceleration term

$$j_{acc,i} = \left|\left|\mathbf{x}_{dd,i}\right|\right|_2^2$$

And the jerk term

$$j_{jerk,i} = \left|\left|\boldsymbol{x}_{ddd,i}\right|\right|_2^2.$$

To adhere to kinematic constraints acceleration bounds are included:

$$x_{ddd,i} \leq a_{max}, i = 2, \dots, N-2.$$

Where $\boldsymbol{x}_{ref}(t)$ is referred to as the reference trajectory and is calculated using the intelligent driver model (IDM) equations [1]. Position, acceleration and jerk at the i-th point are indicated as $\boldsymbol{x}_{,i}$, $\boldsymbol{x}_{dd,i}$ and $\boldsymbol{x}_{ddd,i}$ respective. To be able to apply the IDM equations, all traffic participants have to be projected on the centre line. Then the IDM equations are applied to obtain a reference trajectory. The reference trajectory is then transformed back in the Cartesian coordinate system. To overcome the problem that the reference trajectory is only located on the centre line the above optimization problem is solved. It is also important to mention that the first three points of the trajectory are fixed parameters of the problem. This is because we need to ensure that the initial position, orientation, velocity and acceleration as well, correspond to the real state of the ego vehicle. The ensurance by fixing the first three points is due to the fact that the yaw angle is calculated by two adjacent points. Velocity and acceleration are determined by applying finite differences. To smooth the final trajectory and make it comfortable to vehicle passengers, acceleration and jerk terms are inserted. This complements the verification of the requirement *R_EN4_model1.1*.

2.1.2 Improvements

Since many parts of the concept were exchanged as explained in the concept section, it is hard to exactly mention improvements comparing to the concept described e.g. in D3.3. Instead, we like to highlight some strengths of the method. An important point we want to emphasize is the really low computation time of the method, therefore there are not any problems concerning real-time capability. Furthermore we can directly consider leading vehicles using a driver model for social interaction. Further details are to find in [2].

2.1.3 Implementation

The implementation is done in C++. Dependencies used are on the one hand "Eigen" a library for linear algebra application and "Worhp" a library for optimization problems.

Necessary inputs are:

- the map data, which is currently a reference line on the lane it is intended to drive on,
- Vehicle tracks in the form of rectangle boxes,
- Speed limit contained in the digital map,
- the ego vehicle position to determine the initial trajectory states,
- the HMI input to release the overtaking manoeuvre for the PETER scenario.

In Ulm, the planner is integrated within the "ADTF" framework.

Privacy Issues

There are no problems concerning privacy issues since the environmental information that is used for planning does contain any person specific information.

2.2 E4.2 - Learning of intention from driver (HMT)

In the context of AutoMate, the purpose of the algorithms to learn the driving intention from the driver is to enable the system to adapt its automation strategies to the driver's preferences and guarantee a human expert-like driving behaviour.

The Learning of intention from driver is based on the Driver Intention Recognition (DIR) model from WP2 and shall adapt the model parameters of an initial DIR during driving to create an individualized DIR, which should detect intentions of its corresponding driver more robustly than the initial DIR. The main focus is the detection and prediction of lane change intentions during the PETER scenario. In addition, the individualization of a model, to detect the intention to enter a roundabout during the EVA scenario, is a valid option.

For the development of the learning of intention from driver in AutoMate, we started with a pre-existing framework, consisting of libraries and algorithms for the creation and utilization of (Dynamic) Bayesian Networks, originally developed during the former EU project HoliDes[2]. However, for AutoMate several changes and extensions were made. With respect to Enabler 4.2 these are, for example, the general ability to store model parameters in a way that they can be updated during runtime, the update methods for different distribution types used by the DIR, and methods for an online sample generation.

[2] www.holides.eu

In the following, it is described how the TeamMate car learns from the driver during the third cycle of AutoMate.

2.2.1 Concept

The concept for the Learning of intention from driver has already been described in D3.5. The algorithms of this enabler extend the software for the aforementioned DIR model. The DIR model is basically a dynamic Bayesian Network that creates estimations about the intentions of a human driver. As already mentioned the intention to change the driving lane is focussed. The intention itself is considered as a hidden process, which creates observable effects.

The parameters of the initial DIR model are learned offline by using expert annotated multivariate time-series of driving data. In this case, the intention is not hidden in training data and the model is learned with *complete data*. For the case of Enabler 4.2 the expert annotation is not an option since it requires a lot of time and effort. Thus, the intentions remain hidden at first and the algorithm has to handle *incomplete data.* For each point in time t the current observations of the traffic situation $\boldsymbol{o}^t$ are received. To learn the driving intention *lane change* the first problem is to determine if and when a lane change happened, as well as to estimate when the lane change manoeuvre started or the intention was formed. By detecting that the ego vehicle actually changes its lane at time t, the intention I^t can be assumed as known. The estimation of the intention before t can be formulated as a smoothing problem:

$$p(\boldsymbol{I}^{t-x}|\boldsymbol{O}^{1:t}) \propto p(\boldsymbol{I}^{t-x}|\boldsymbol{O}^{1:t-x})p(\boldsymbol{O}^{t-x+1:t}|\boldsymbol{I}^{t-x})$$

Once the intention is estimated, the data for updating the DIR model can be considered as *complete*. We refer to this process as the online sample generation. For more detail on this point, please see the previous deliverables D3.5 and D3.3.

The second problem is the actual update of the distribution parameters of the DIR model. This relies on Bayesian parameter learning and the usage of hyper-parameters Θ. The hyper-parameters describe probability distributions over the model parameters and can be updated as new evidence becomes available through observations. The values of the model parameters can then be inferred. For more details on this point, please see the previous deliverables D3.5 and D3.3.

2.2.2 Improvements

Online sample generation

Until the second cycle the implemented online sample generation relied on the detection of certain events like the actual change of the ego lane mentioned in the previous section. For the third cycle, the detection of certain events is still supported. Like in the cycles before, this requires the definition of certain simple rules like $IF\ ego\_lane^{t-1}! = ego\_lane^{t}\ THEN\ \ I^{t} = 1$. For the estimation of the intention before t, the smoothing problem is solved by performing forward-backward inference utilizing the current DIR model. The forward-backward inference is a two-pass approach. In the first pass, the so called forward pass, the algorithm calculates the forward probabilities by filtering from 1 to $t - x$ to obtain $p(\boldsymbol{I}^{t-x}|\boldsymbol{O}^{1:t-x})$. In the backward pass, the backward probabilities $p(\boldsymbol{O}^{t-x+1:t}|\boldsymbol{I}^{t-x})$ are calculated. By combining forward and back ward probabilities the smoothing result is determined. A more detailed description ca be found in [3, 4, 5].

It is also possible to employ a second model. The second model can be useful, e.g., if it is specialized in the detection of certain manoeuvres or is less complex and thus can be inferred much faster. This could be the case if the second model considers variables that are not considered by the DIR model, since the DIR model only relies on the situational context as described in D2.4 and D2.6. The second model can be defined in the same framework as the DIR, thus it can be updated by the same parameter updating methods. However, additional rules might be needed to translate the inference results of the second model into the sample annotation for the DIR model.

Distribution update methods

Concerning the methods for updating the parameters of the DIR model, during the second cycle it was possible to update discrete variables with their corresponding distributions and continuous variables described by (multivariate) Gaussians. However, the DIR model also supports the usage of Gaussian Mixture Models (GMM) to approximate the underlying densities of continuous variables. Additionally the evaluation in D2.4 has shown that the GMMs are potentially better suited to approximate the densities of the continuous variables than single Gaussians. Thus, for the third cycle update methods for the GMMs used in the DIR models were implemented. Currently a batchwise fixed complexity update is performed similar to the approach described in [6] or sequential EM described in [7]. As already mentioned, a GMM can be used to approximate the underlying probability density function of continuous data. The general assumption is that the data is generated by a set of Gaussian distributions. For offline learning, the parameters of the GMM are often estimated by using Expectation Maximization (EM). For the online parameter updating a single E-step and a special M-step are performed. First the E-step determines the responsibilities, which basically describe how strong a data point belongs one of the Gaussian distributions, of the newly arrived data points, based on the old GMMs parameters. Afterward, the M-step updates the old parameters of the Gaussian distributions with the newly arrived data but weighted with their corresponding responsibilities. This complements the verification of requirement *R_EN4_model2.3*, stating that the model must be able to update the parameters of the DIR model.

2.2.3 Implementation

For the integration into the TeamMate System Architecture and in the demonstrators, the component is exported as a dynamic library. For a simpler handling the functionality for the prediction of the spatial and temporal evolution of the traffic scene, the online risk assessment, and the driver intention recognition were put together into a single C++ Dynamically Linked Library. For integration into the ULM simulator, this library is wrapped in a DPU, which is a format for exchangeable modules of the SiLab simulation software used by ULM. For the integration into the VED demonstrator the library is wrapped into a RTmaps package, which allows a seamless integration into the RTMaps system environment used by VED. This complements the verification of requirement *R_EN4_model2.8*, stating that the module must be integrable in the demonstrators.

As already mentioned in D3.5 and defined in D5.1 "TeamMate System Architecture including open API for 2nd cycle" the Learning of intention from driver requires basically the same data as the DIR from WP2 these are:

- the static environment mode (including a digital road map)l,
- the dynamic environment model (including the state of the TeamMate vehicle and the state of all dynamic objects detected by the TeamMate vehicle)

On an internal level, the enabler operates on the following input:

- The current lane of the TeamMate car (ego_lane).
- Features of other traffic objects dependent on the DIR model.
- Features of the TeamMate car dependent on the DIR model, e.g., position, heading.

In the case of a simulator demonstrator, the above mentioned inputs can be provided directly by the simulation software. In the case of the VED real vehicle demonstrator, the current lane and other features of the TeamMate are provided by the VED real vehicle internal sensors, e.g., a high precision GPS. The features of other traffic object could be provided by external sensors, like e.g., LIDARs, RADARs, or camera. This depends on the setup of the demonstrator vehicle.

The Learning of intention from driver component general works as follows:

- load provided initially offline learned DIR model

- each received data is put into a ring buffer which can hold several seconds of data
    - Evaluate $ego\_lane^{t-1}! = ego\_lane^{t}$ to check if a lane change happened.
    - If a lane change is detected forward-backward inference for buffered data is performed to label data.
    - If no lane change happened check for buffered data that is older than a certain threshold time t_{tresh} and set corresponding non lane change label.
    - If lane change was detected or the buffer is full
        - Transformed labelled buffer data to learning dataset type this can be used for several learning procedures.
        - For every distribution in the loaded DIR model run the corresponding parameter updating procedure.
        - Store the updated DIR model.
        - Inform DIR component about available update.

Privacy Issues

The current implementation for the learning of intention from driver solely updates the single initial DIR model. Since, this model only contains distribution parameters about driving data no personal data about the driver is stored. This complements the verification of requirement *R_EN4_model2.7*, stating that the online learning module must not safe any personal data in an unencrypted manner.

2.3 E5.1 - Online risk assessment (OFF + DLR + HMT)

The purpose of online risk assessment in AutoMate is the calculation of *safety corridors* that quantify the safety of the current and near-future traffic situation according to a metric of risk. Such safety corridors can be used by the TeamMate car to assess and plan safe and feasible trajectories, leading to a set of algorithms that allow identifying safe and reasonable arrangements of the driving process. As described in deliverable D3.5, Enabler E5.1 is required to address the *Peter scenario*. However, regarding the close collaboration with partner VED, the design and development has also closely considered the *Martha scenario*.

Importantly, the algorithms for constructing the safety corridor and trajectory assessment can be divided into two independent parts, corresponding to the handling of other traffic participants in the vicinity of the TeamMate vehicle and the lane boundaries. Within AutoMate, the realization of these two parts is provided by different partners, with individual plans for verification and validation. In the following, we will therefore distinguish between online risk assessment for dynamic objects like other traffic participants and the road boundaries.

Concept, development, and implementation of the algorithm pipeline of the Online Risk Assessment have been entirely developed within the context of AutoMate. No part of the component has been inherited from previous projects nor addressed in any other European projects.

2.3.1 Concept

2.3.1.1 Safety Corridor around Dynamic Objects

The overall concept of the AutoMate enabler for online risk assessment has already been presented in detail in deliverable D3.5 "Concepts and algorithms incl. V&V results from 2nd cycle" and has not changed within the third cycle. For the sake of this document, we will however shortly recapitulate the definition of safety corridors in respect to other traffic participants:

Let Δ denote a temporal step width in seconds and η_{max} denote a desired number of prediction steps, resulting in a desired prediction horizon $\eta_{max}\Delta$ seconds, and $V = \{v_1, \dots, v_{n_V}\}$ denote a set of n_V objects (usually traffic participants) detected by the sensor platform of the TeamMate vehicle at some current time step t.

As previously described in D3.3 "Concepts and algorithms incl. V&V results from 1st cycle", the output of the online risk assessment at each time step t is defined as a set of so-called *safety corridors*

$$\boldsymbol{c}^{t:t+\eta_{max}\Delta} = \left(\boldsymbol{c}^{t:t+1\Delta}, \boldsymbol{c}^{t+1\Delta:t+2\Delta}, \dots, \boldsymbol{c}^{t+(\eta_{max}-1)\Delta:t+\eta_{max}\Delta}\right).$$

For the sake of readability and, as envisioned for online risk assessment, we will silently assume that $\Delta = 1s$ and omit the addition of Δ in the following.

Each safety corridor $\boldsymbol{c}^{i:i+1}, t \leq i < \eta_{max}$ specifies a region over a temporal interval $[i, i+1]$ for which the probability of collision between the TeamMate vehicle and any object $v \in V$ is below a user-defined threshold δ. For this, each safety corridor $\boldsymbol{c}^{i:i+1}$ is defined as a set of *polygons*

$$\boldsymbol{c}^{i:i+1} = \{P_1^{i:i+1}, \dots, P_{n_V}^{i:i+1}\},$$

where each polygon P is composed of a finite number of line segments, specified by a sequence of points $P = (A_1, \dots, A_k)$, where each $A_j \in P$ is defined as a pair $A_j = (x_j, y_j)$ denoting the x- and y-coordinates in a Cartesian coordinate system. Within a safety corridor $\boldsymbol{c}^{i:i+1}$, each polygon $P_j^{i:i+1}, j = 1, \dots, n_V$ denotes a polygonal line that *excludes* the region for which the probability of collision with the corresponding object is below some threshold $\delta_v = 1 - \sqrt[n_V]{(1-\delta)}$. As a result, the overall safety corridor $\boldsymbol{c}^{i:i+1}$ implies a continuous "*safe area*" for the temporal interval $[i, i+1]$, in which the probability of collision with *any* object is below δ. The safety corridor can thus be used by the path planning algorithm to plan current and future trajectories within the temporal interval $[i, i+1]$.

A visual example of a safety corridor is provided in Figure 1, which depicts a safety corridor $\boldsymbol{c}^{t+1:t+2} = \{P_1^{t+1:t+2}, P_2^{t+1:t+2}\}$, over a temporal interval $[t+1, t+2]$. Combined with polylines associated with the lane boundaries, the safety corridor defines a safe area of collision-free travel, shown by the grey hachured area. We note that a safety corridor abstracts from the dimension of the TeamMate vehicle itself.

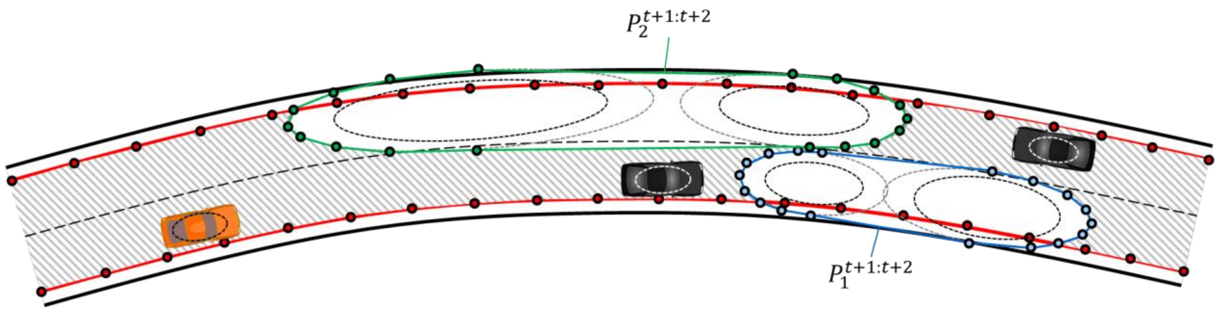

Figure 1: Exemplary visualization of a safety corridor for a temporal interval $[t+1, t+2]$.

To derive the safety corridors, the online risk assessment relies on a prediction of the temporal and spatial evolution of the traffic situation, provided by the Vehicle and Situation Modelling Module (for a detailed description, please refer to D2.4 "Sensor Platform and Models incl. V&V results from 2nd cycle" and D2.6 "Sensor Platform and Models incl. V&V results from 3rd cycle").

2.3.1.2 Safety Corridor between Road Boundaries

During the first cycle (cf. deliverable D3.3) we presented and described a novel approach for extracting the safety boundary corridors between road boundaries. In the second cycle (cf. deliverable D3.5), we considered uncertainty in the TeamMate vehicle position to extract the risk assessed safety boundary corridor. This approach is visualised with the blue and red polylines in Figure 2 and Figure 3.

In deliverable D3.5, we had described that the system uses the OpenDrive format *Digital Map* for parsing localization information, but during the third cycle, a new interface has to be developed in the context of the online risk assessment module to access the proprietary map format from Vedecom.

Nevertheless, the approach and concept for extracting the safety corridor between road boundaries remains the same as we have described in the previous deliverables.

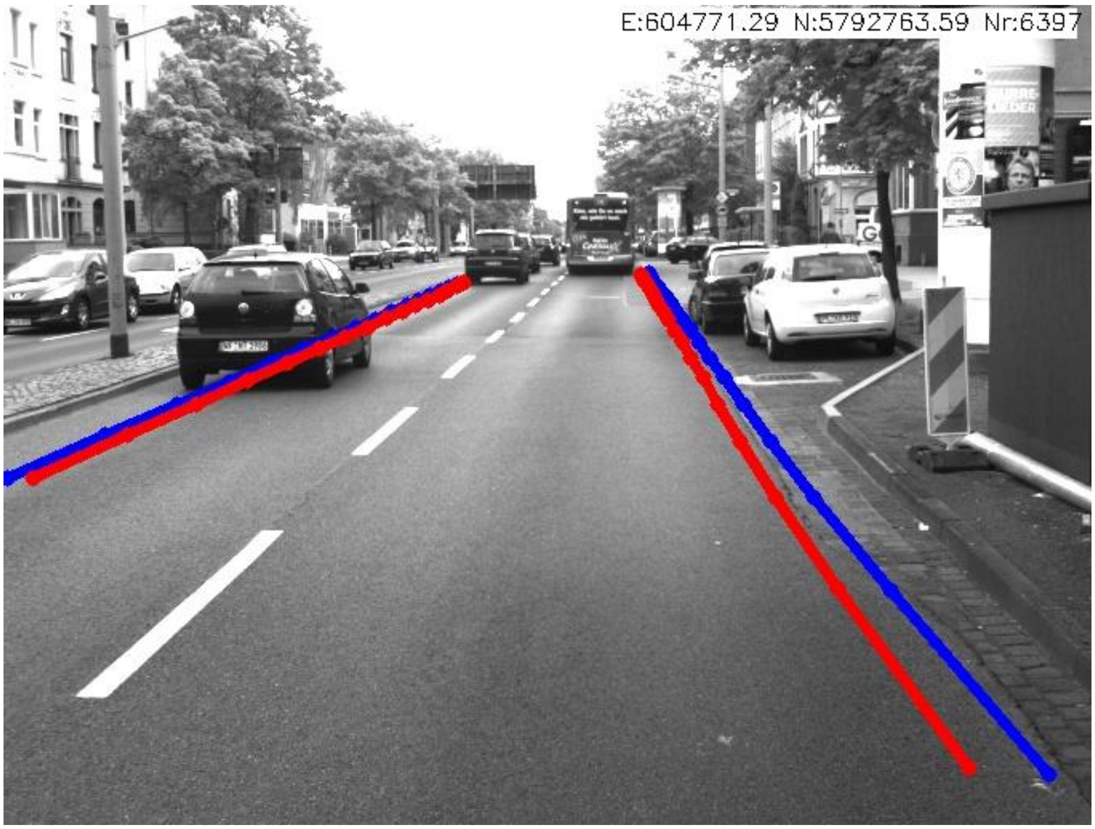

Figure 2: Example of a generated safety corridor assuming that the ego-pose uncertainty is ±0.25m.

The module, first maps the TeamMate vehicle pose and position into map coordinates and extract the required ego lane and lane marker information of those lanes that are associated to the ego lane in the driving direction. Based on the obtained lane marking information, the possible boundary corridor is extracted. Addressing requirement *R_EN5_alg1.1*, the risk of the safety boundary is assessed by taking into account the uncertainty in the

TeamMate's position and the requested degree of "Probability of Collision" (*POC*).

As you see from Figure 2 and Figure 3, the true boundary corridor (blue polyline) is shifted with respect to the quantile z_{δ_V} and $z_{1-\delta_V}$ computed with the help of requested *POC* and uncertainty in position of the TeamMate vehicle. At the end, the shifted polylines that are closest to the vehicle (red polylines) are considered for providing the risk assessed boundary safety corridor ensuring the requested *POC* threshold δ_V is fulfilled.

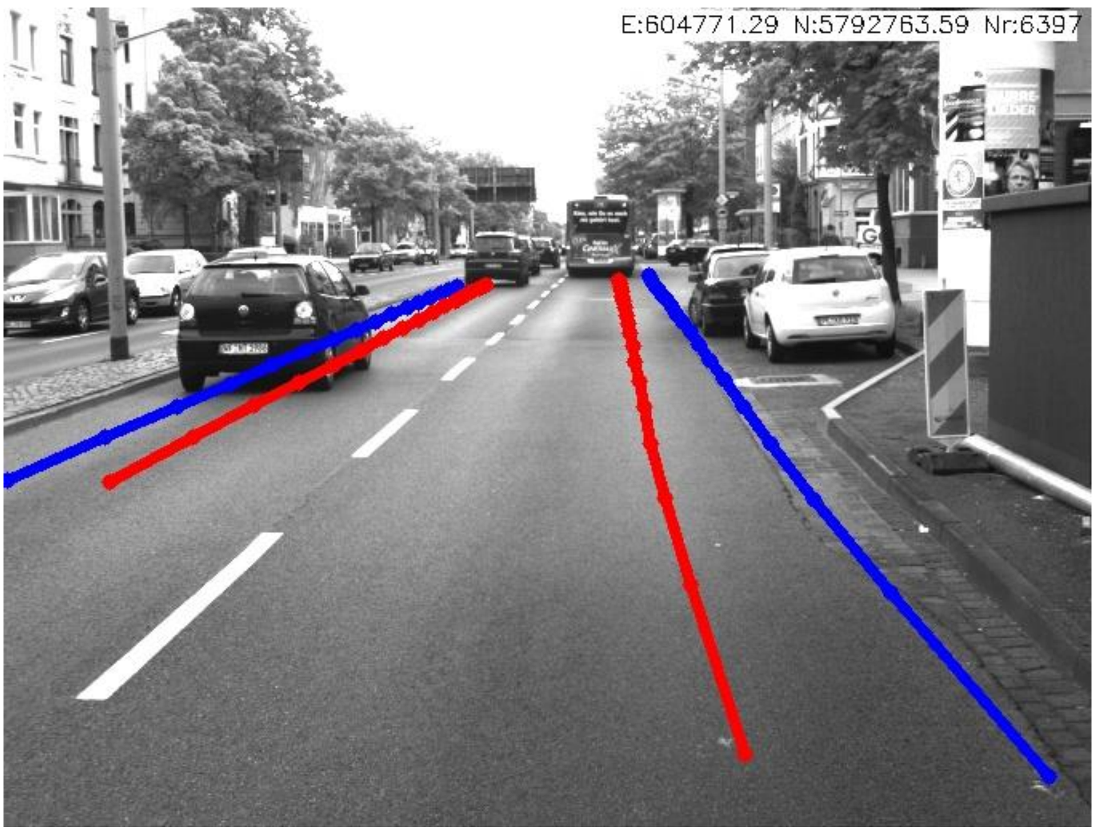

Figure 3: Example of a generated safety corridor assuming that the ego-pose uncertainty is ±1m.

The lane markings from the map do not match the one in the camera image because the map's representation is in a 2 dimensional space. The missing height information into the map leads to a projection error into the image plane. For simplification, we used the height of the camera above the road as the height of the map data.

2.3.2 Improvements

2.3.2.1 Safety Corridor Around Dynamic Objects

The qualitative performance of the online risk assessment is primarily determined by the quality of the predicted spatial and temporal evolution of the traffic scene. As such, the primary focus for improvements in the third cycle was put on the improvement of Vehicle and Situation Modelling Module (c.f., deliverable D2.6 "Sensor Platform and Models incl. V&V results from 3rd cycle"). Exclusively for only risk assessment, we focused on the functionality of trajectory assessment and improving the computational efficiency of the online risk assessment, which shall be described in the following.

2.3.2.1.1 Trajectory Assessment

If at some time t, the online risk assessment has successfully prepared the set of safety corridors $\boldsymbol{c}^{t:t+\eta_{max}\Delta} = \left(\boldsymbol{c}^{t:t+1\Delta}, \boldsymbol{c}^{t+\Delta:t+2\Delta}, \dots, \boldsymbol{c}^{t+(\eta_{max}-1)\Delta:t+\eta_{max}\Delta}\right)$ for some threshold probability of collision δ, these can be used for trajectory assessment, i.e. to decide whether a trajectory planned by the trajectory planner of the TeamMate vehicle is safe or unsafe in respect to other traffic participants.

The input for trajectory assessment is a trajectory R^t defined as a sequence of n_R points

$$R^t = (r_1, \dots, r_{n_R}),$$

where each point $r_i = (t_i, x_i, y_i, \theta_i)$ specifies the planned x_i and y_i coordinates of the center of the bounding box of the TeamMate vehicle and its absolute yaw angle θ_i at some future point in time t_i. For a reasonable assessment, we require that the difference in time between two successive points r_i and r_{i+1} is below 0.5Δ, i.e., $t_i - t_i \leq 0.5\Delta$, where such a requirement is not met, we use linear interpolation to create a trajectory that meets the requirement.

The actual trajectory assessment is then performed as follows: for each point $r_i = (t_i, x_i, y_i, \theta_i)$ in the trajectory, we use the specified coordinates and yaw angle and the dimension of the TeamMate vehicle to derive the corners of the TeamMate vehicle's bounding box. Based on the timestamp t_i, we then search the safety corridor $\boldsymbol{c}^{t+\mathrm{j}\Delta:t+(j+1)\Delta}$ for which $t_i \in [t + \mathrm{j}\Delta, t + (j+1)\Delta]$. For each polygon within the safety corridor $\boldsymbol{c}^{t+\mathrm{j}\Delta:t+(j+1)\Delta}$, we then check whether any corner of the TeamMate vehicle is located within the polygon. If true, we conclude that the trajectory is unsafe for the desired threshold of the maximum probability of collision δ. If false for the complete trajectory, we conclude that the trajectory is safe. The assessment can then be used by the trajectory planner to either trigger the execution of the trajectory or repeat the planning process.

A visual example trajectory assessment is provided in Figure 4, depicting the assessment of a trajectory consisting of $n_R = 4$ points, assumed to fall within the temporal interval $[t + 1, t + 2]$. Green circles depict that the corner of the bounding box lie outside, red circles depict that the corner lie inside of the safety corridor, rendering the overall trajectory as unsafe.

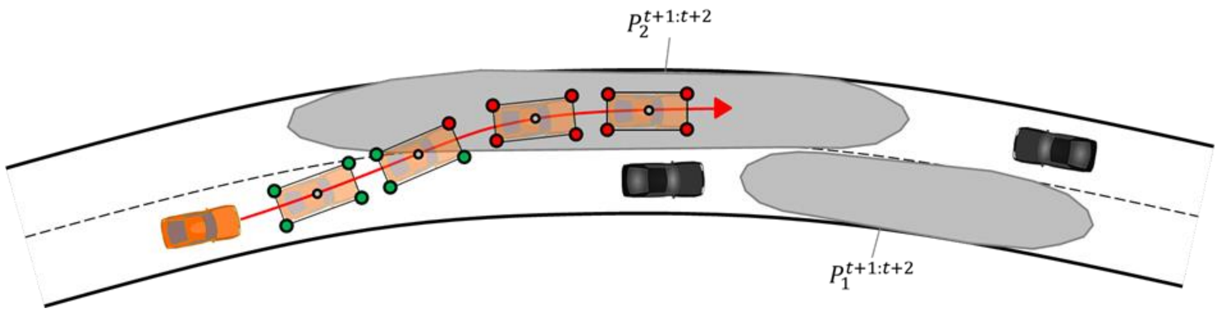

Figure 4: Exemplary visualization of trajectory assessment for a temporal interval $[t+1, t+2]$.

2.3.2.1.2 Computational Efficiency

As the online risk assessment is used to assess the planned trajectories of the TeamMate vehicle, it must comply with strict requirements concerning the required computation time and resources for the construction of safety corridors and trajectory assessment. In this section, we will provide asymptotic boundaries on the computational complexity of online risk assessment, which have been used to improve the overall computational efficiency during the third cycle.

To assess the computational complexity of the construction of safety corridors for other traffic participants, we will shortly recap the construction process as described in deliverable D3.5 “Concepts and algorithms incl. V&V results from 2nd cycle”.

As a reminder, let Δ denote a temporal step width in seconds and η_{max} denote a desired number of prediction steps, resulting in a desired prediction horizon $\eta_{max}\Delta$ seconds, and let $V = \{v_1, \dots, v_{n_V}\}$ denote a set of n_V objects (usually traffic participants) detected by the sensor platform of the TeamMate vehicle at some current time step t. As a prerequisite for the construction of the safety corridors, a prediction of the spatial and temporal evolution of the traffic scene provides as input a sequence of multivariate Gaussian distributions over the Cartesian coordinates of the centre and the orientation of each object $v \in V$ for the prediction horizon, given the available sensor information $\boldsymbol{o}^{1:t}$ obtained by the TeamMate vehicle thus far: $p\left(X_v^{t+\Delta i}, Y_v^{t+\Delta i}, \Theta_v^{t+\Delta i} | \boldsymbol{o}^{1:t}\right), i = 0, 1, \dots, \eta_{max}$.

Given some threshold δ for the maximal probability of collision, determining the internal thresholds for each $v \in V$ as $\delta_v = 1 - \sqrt[n_V]{(1-\delta)}$, the construction of the safety corridor is then performed as follows:

1. For each $v \in V$ and each $i \in \{0, 1, \dots, \eta_{max}\}$, using available sensor information providing the length and width of the object, the multivariate Gaussian $p\left(X_v^{t+\Delta i}, Y_v^{t+\Delta i}, \Theta_v^{t+\Delta i} | \boldsymbol{o}^{1:t}\right)$ is converted into each one polygon for each corner of the bounding box of the object, in the following referred to as a *corner polygons*, enclosing $(1-\delta_v) \cdot 100\%$ of the probability mass for the location of the respected corner.
2. The resulting four corner polygons are then combined in a single polygon, referred to as a *coverage polygon*, by deriving the complex hull of the four corner polygons. Once constructed, each coverage polygon obtained for some multivariate Gaussian $p\left(X_v^{t+\Delta i}, Y_v^{t+\Delta i}, \Theta_v^{t+\Delta i} | \boldsymbol{o}^{1:t}\right)$ can be interpreted as enclosing $(1-\delta_v) \cdot 100\%$ of the probability mass of the location of the complete vehicle.
3. Finally, each two consecutive coverage polygons obtained from the multivariate Gaussians $p\left(X_v^{t+\Delta i}, Y_v^{t+\Delta i}, \Theta_v^{t+\Delta i} | \boldsymbol{o}^{1:t}\right)$ and

$p\left(X_v^{t+\Delta(i+1)}, Y_v^{t+\Delta(i+1)}, \Theta_v^{t+\Delta(i+1)} | \boldsymbol{o}^{1:t}\right)$ are combined into a single polygon $P_v^{t+\Delta i:t+\Delta(i+1)}$ by deriving the complex hull of the two coverage polygons.

Given these steps, we can attempt to provide the approximative boundaries of the computational complexity.

1. Let each corner polygon be composed of a number of n_P vertices, the derivation of a single corner polygon can be performed in linear time complexity, $\mathcal{O}(n_P)$. Under the assumption of n_V considered objects and a maximum number of prediction steps η_{max}, this step has to be performed $4n_v(\eta_{max}+1)$ times in total.

For the construction of the convex hulls in step 2 and 3, we rely on the so called "Andrew's monotone chain convex hull algorithm". Let n denote the number of points for which the convex hull shall be constructed. The algorithm first sorts all n points lexicographically, with a quasilinear time complexity of $\mathcal{O}(n \log n)$, and then constructs an upper and lower hull of the points in $\mathcal{O}(n)$ time, resulting in an overall quasilinear time complexity of $\mathcal{O}(n \log n)$.

2. For deriving a single coverage polygon, we need to construct a convex hull from $4n_P$ vertices. Assuming n_V considered objects and a maximum number of prediction steps η_{max}, this step must be performed $n_v(\eta_{max} + 1)$ times in total.

3. Each two consecutive coverage polygons will then be combined into a single polygon using the convex hull algorithm. In the worst case, each coverage polygon will consist of $4n_P$ vertices, in practice however, resulting coverage polygon will average on roughly $n_P + 4$ vertices.

Assuming n_V considered objects and a maximum number of prediction steps η_{max}, this step must be performed $n_v \eta_{max}$ times in total.

As a result, the computational complexity of the construction of the safety corridors is dominated by the quasilinear time complexity of the convex hull algorithm, which has to be performed $2n_v\eta_{max} + n_v$ times in total. As such, the computational complexity grows quasilinear with the number of vertices n_P per corner polygon, but only linear with the number of objects n_v considered and the maximal number of prediction steps η_{max}, which easily allows for specific calibrations based on the requirements in a current situation.

Assuming that the maximum number of prediction steps η_{max} have been chosen thoughtfully and that all n_V considered objects are crucial, potential optimizations are limited to a reduction of the number of vertices n_P per corner polygon. The number of vertices n_P per corner polygon provide a trade-off between computational complexity and the correctness of the resulting corner polygon in respect to the underlying Gaussian. More specifically, the error induced by approximating the area of the ellipse by a polygon can easily be obtained from the reference circle. Here we have that the area of circle with $r = 1$ is given by $A = \pi r^2$, while the area of a polygon with n equidistant vertices along the arc is given by:

$$A = \frac{1}{2} n r^2 \sin \frac{360°}{n}.$$

During the second cycle, we opted for a number of $n_P = 100$ vertices per corner polygon to obtain an approximation error of $< 0.1\%$. To improve the overall computational performance, we tested a reduction of the number of vertices down to $n_P = 10$ vertices per corner polygon. Although this introduces a substantial approximation error of 6.45% for each individual corner polygon, it turns out that the approximation error of the whole temporal polygon in practice shrinks with the chosen temporal step width Δ. To assess the resulting gain in computational performance and potential loss in quality, we performed validations for both $n_P = 100$ and $n_P = 10$, which are reported in Section 3.3.1.

For trajectory assessment, the computational complexity is dominated by the necessary assessments of whether a corner of the bounding box of the TeamMate vehicle is located within the polygons comprising the safety corridors, for which we rely on a simple line crossing algorithm with linear time complexity based on the number n_P of vertices in the polygon, $\mathcal{O}(n_P)$. Now, let n_R denote the number of points in the trajectory and n_V denote the number of considered objects in the vicinity of the TeamMate vehicle. Morever, let's assume that the number n_P of vertices in each polygon is constant for simplicity, as each point in the trajectory results in four required assessments (one for each corner of the bounding box) of exactly one polygon for each considered object (ignoring the possibility that a timestamp is located on the boundaries of two successive temporal intervals). In such a way, trajectory assessment requires $4n_Rn_V$ of such assessments in the worst case (obviously, trajectory assessment can be aborted early once the trajectory has been classified as unsafe).

Assuming that both all n_R points on the trajectory and all n_V considered objects are crucial, we once again need to reduce the number of vertices n_P for reducing the computational complexity of trajectory assessment.

2.3.2.2 Safety Corridor between Road Boundaries

During the third cycle, we have focused on extending the module to perform online risk assessment for planned trajectories. Figure 5 provides an overview how the trajectory assessment is performed. The risk assessed safety boundary corridor between road boundaries is used to assess the planned trajectory. Those planned trajectories that do not fall within the left and right boundary points of the safety corridor are classified as unsafe trajectories, and those that are within the limits of the left and right safety corridor boundaries are classified as safe trajectories. The surrogate safety measure “Time To Collision” (TTC) is appended for those trajectories that are classified as unsafe.

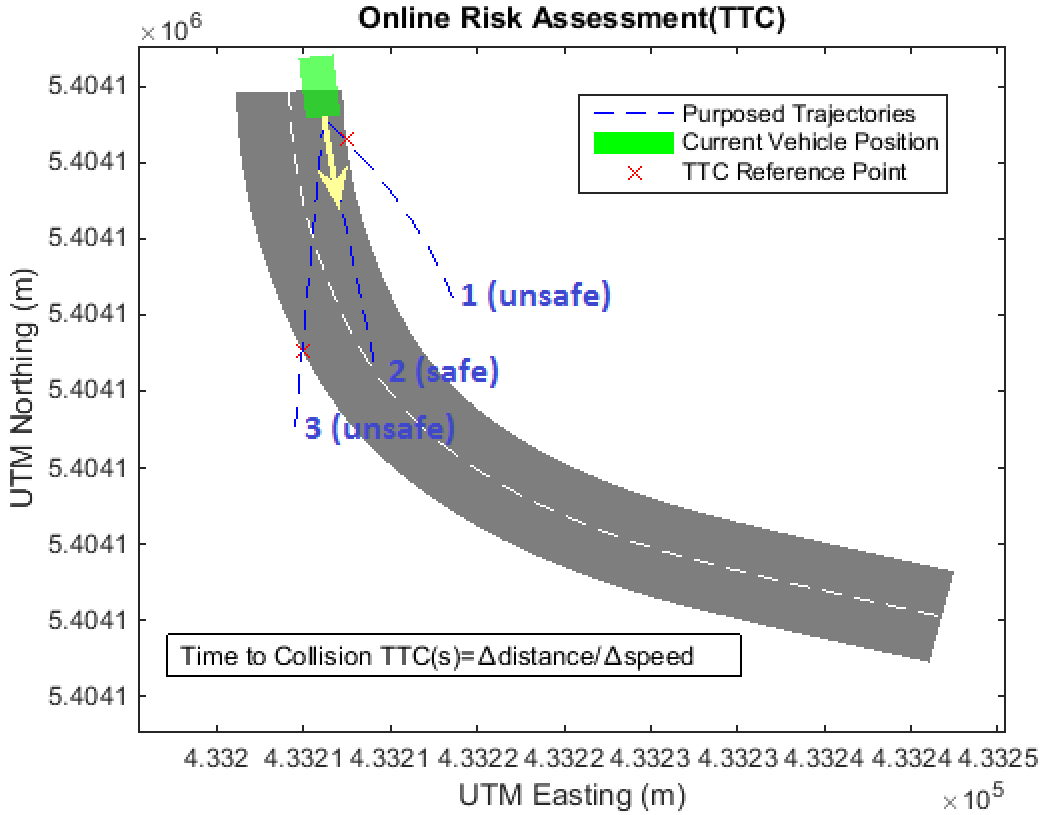

Figure 5: Example how the online risk assessment of planned trajectory is performed Implementation.

2.3.3 Implementation

2.3.3.1 Safety Corridor around Dynamic Objects

Concerning the implementation, online risk assessment in respect to other traffic participants has been integrated together with the functionality for the prediction of the spatial and temporal evolution of the traffic scene, the driver intention recognition, and the online learning into a single C++ Dynamically Linked Library. Within the second and third cycle, this DLL was embedded into functional modules for the simulation environment SILAB, used by the ULM simulator demonstrator, and the third-party software RTMaps, used by the VED real vehicle demonstrator, enabling the utilization of these functionality in the resp. demonstrators.

The required input of the overall module conforms to the TeamMate system architecture and consists of

- the static environment model,
- the dynamic environment model,
- and the planned trajectory,

as defined in deliverable D5.1 "TeamMate System Architecture including open API for 2nd cycle". On an internal level, the online risk assessment in respect to other traffic participants purely operates on the following input:

- A prediction of the spatial and temporal evolution of the traffic scene in terms of a sequence of multivariate Gaussian distributions over the Cartesian coordinates of the centre and the orientation of each object $v \in V$ for the complete prediction horizon, given the available sensor information $\boldsymbol{o}^{1:t}$ obtained by the TeamMate vehicle thus far: $p\left(X_v^{t+\Delta i}, Y_v^{t+\Delta i}, \Theta_v^{t+\Delta i} | \boldsymbol{o}^{1:t}\right), i = 0, 1, \ldots, \eta_{max}$. Within AutoMate, this input is readily provided by the Vehicle and Situation Modelling Module developed in WP2.
- The dimension (i.e. width and length) of each considered object $v \in V$. In the case of simulator demonstrators, these can be provided directly by the simulation software, in the case of the VED real vehicle demonstrator, this are provided as a common output of laser scanners.

- For trajectory assessment, a trajectory to be assessed. Within AutoMate, this input is readily provided by the algorithms for trajectory planning.

As apparent from this list, the module for online risk assessment does not process any personal or private data. Furthermore, the online risk assessment does not make use of the state of the driver, e.g., intention, distraction, or drowsiness. This can be justified as follows: In its current implementation, the online risk assessment is envisioned as providing relevant information to the TeamMate vehicle concerning the potential risk or safety of trajectories planned by the trajectory planning modules. As these are intended to be performed autonomously, without intervention of the driver, online risk assessment operates independently of the driver.

2.3.3.2 Safety Corridor between Road Boundaries

Integration of components into the TeamMate System Architecture is planned in RTMaps environment. As part of the fulfillment of requirement *R_EN5_alg1.4*, the component is exported as 32-bit windows DLL with necessary callback function. This approach allows a seamless integration into the RTMaps system environment. In a previous corresponding deliverable D3.5 we presented that the prediction of the spatial and temporal evolution of the traffic scene and online risk assessment where coupled as single module, but due to the complexity and resource overheads, this particular submodule of online risk assessment is exported as a DLL with callback function.

```
#ifdef ONLINERISKASSESMENTDLL_EXPORTS
#define ONLINERISKASSESMENTDLL_API __declspec(dllexport)
```

```
#else
#define ONLINERISKASSESMENTDLL_API __declspec(dllimport)
#endif

#pragma once

namespace VEDECOM
{
	enum TRAJECTORYASSESMENT{SAFE=100,UNSAFE=50,NOTCOMPUTED=-1};
	enum ISSAFETYBOUNDARYVALID{NOTVALID=-1, VALID=1};
	class CallBackFunc
	{
	public:
		//------callback functions that are exported---------//
		static ONLINERISKASSESMENTDLL_API void checkCallBack();

		static ONLINERISKASSESMENTDLL_API bool
WraperfuncLoadMapEnvironment(const char*  _pathToDigitalPath, const int &
n_RaodpointsInMap);
		/* One time CallBackFunc for loading Digital Map environment*/

		static ONLINERISKASSESMENTDLL_API int
WraperfuncGetSafeBoundaryCorridor(double* _safetyCorridorLeftPnts_io, double*
_safetyCorridorRightPnts_io,  int& _nbPnts_shiftedleft,  int&
_nbPnts_shiftedright, const double* _egoPositionUTM_xy, const double*
_egoUncertainity_xy, float POC = 1.0);
		/* For Safe Boundary Extraction Between Road Boundaries */

		static ONLINERISKASSESMENTDLL_API bool
assesTheRiskOfPlannedTrajectory(const double* trajectoryPntsinUTM,const int&
_nofPnts, const float& vehilceVelocity, float&
TTC,VEDECOM::TRAJECTORYASSESMENT& assesmentResults);
		/*call to the function "assesTheRiskOfPlannedTrahjectory" should
always be follwed by "WraperfuncGetSafeBoundaryCorridor" function;
		 1. The function returns enum type (TRAJECTORYASSESMENT) whether
the Trajectory under test is safe or unsafe
		 2. Other output "float& TTC", is the Time-To-Collision for those
trajectory that are unsafe
		 3. For safe Trajectories the TTC is not computed (TTC is only
compute for unsafe trajectory)*/
	};
}
```

The above code illustrates the callback functions that are exposed by the DLLs. As an input, the component requires a highly accurate *Digital Map* of the road network, along with vehicle absolute velocity and position with involved uncertainties.

Assuming the provided sensor data is accurate, the component is intended to work as follows:

- Map data is initialized only once when the system is first booted. At boot time, the module loads the map data and represents it as a meaningful data structure. During subsequent calls to extract the safety corridor, the map which has been loaded into the the static memory of the component is used.
- Every time the function is called to provide the risk assessed safety corridor for a given *POC*, the following steps are executed: The component first matches the TeamMate vehicle orientation and position to the map. Then it extracts the boundary corridor within the vicinity of the vehicle position based on the adjacent lane marking information to the ego lane. To provide the risk assessed safety corridor with requested *POC*, the component uses the vehicle position as the mean μ, and uncertainty of the position as parameters of normal distribution function. Using the concept of a *CDF* (*cumulative distribution function*) the quantile for the requested *POC* is computed which in turn provides the basis for translating the boundary lanes. Those lines that are close inwards to the TeamMate vehicle are considered as the limitations for the vehicle.

3 Validation of WP3 enablers

3.1 E4.1 - Planning and execution of safe manoeuvre (ULM)

To validate the new concept of the trajectory planner we use the value of the KKT condition that describes how optimal the solution of the optimization problem is and the response time, which is the cycle time (namely the time needed to evaluate on trajectory).

3.1.1 Dataset for validation

To validate the planner, a simulation environment is written within MATLAB SW tool. Therefore, the planner is embedded in a “mex” function file, a possibility to directly call C++ coded functions within matlab.

The simulation environment uses the centre lines of the digital map of Ulm to describe the road infrastructure. Figure 6 shows the lane on which the evaluation is done.

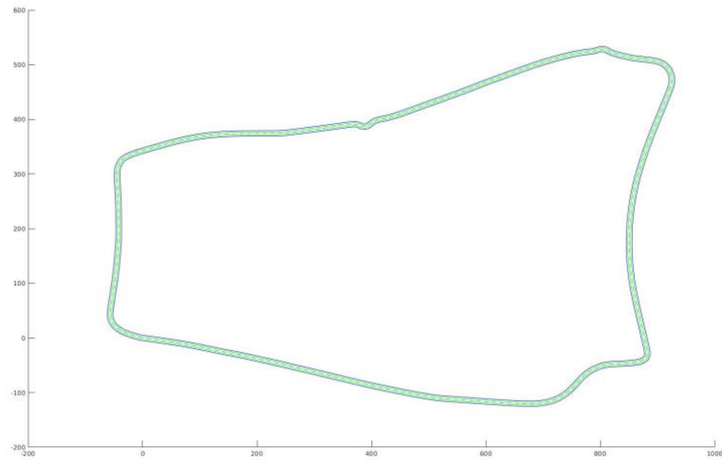

Figure 6: Track used for evaluation

3.1.2 Results

To obtain expressive results the algorithm was called 100 times.

The averaged computation times together with the averaged value of the KKT conditions is shown in the table below. The solver used is WORHP [8].

The number of trajectory support points used for discretization is indicated by N. The time needed for computation is t and the values of the KKT condition is indicated by KKT. The temporal horizon for each simulation is set to $T = 2.95s$

Runtime Performance

Table 1: Runtime performance and KKT values for different number of trajectory support points

N	60	100	200	300
t[ms]	83.82	90.67	110.24	130.17
KKT	<1.67e-7	<3.93e-5	<9.21e-4	<9.69e-4

In Ulm 60 support points of the trajectory are used, thus the method is fast enough to run on real cars. In addition the KKT conditions take always small values < 1e-3, therefore the problem can be solved well. This complements the validation of *R_EN4_model1.10* and *R_EN4_model1.11.*

Details of implementation and evaluation as well are listed in the following table.

Table 2: Hardware details and uses software libraries

CPU	i7-6700K, 4GHz
RAM	32GB
Language	C++
External Software	Eigen, WORHP

3.1.3 State of the Art

Compared to many existing state of the art techniques the planner directly incorporates driver models into planning to generate interaction aware behaviour. Further details are to find in [2].

3.2 E4.2 - Learning of intention from driver (HMT)

According to the validation plan described in D3.6 “Metrics and plan for V&V of the concepts and algorithms in the 3rd cycle”, the Learning of intention from driver is validated by considering the performance of the updated model on a test dataset D_{Test}, in comparison to an initial model, to assess how well the recalibrated driver model recognizes the intentions of the current individual driver. The initial model was trained on a data set D_{Init} while the recalibration is done by using a further dataset D_{New}. The datasets D_{Test} and D_{New} origin from one and the same driver, while D_{Init} may contain data of multiple drivers. For each utilization of the models on the test set D_{Test} a binary confusion matrix is created and recall (REC), False positive rate (FPR), precision (PRE), accuracy (ACC), and F_1-score are derived.

In addition, as described in D3.6, to evaluate the update methods for the GMMs without the influence of the online sample generation the initial model is updated with D_{New} by using the ground truth. A second model is learned offline by using D_{Init} and D_{New}. The updated model and the new offline learned model are evaluated on D_{Test} and compared in terms of the parameters mentioned above.

3.2.1 Dataset for Validation

The mentioned data sets D_{Test}, D_{Train}, and D_{Train_Online} were obtained from the simulator study conducted for training and evaluation of the DIR model. For a detailed description of the experiment and the gathered data, please see D2.4 “Sensor Platform and Models incl. V&V results from 2nd cycle” and D2.5

"Metrics and Experiments for V & V of the driver, vehicle and situation models in the 3rd cycle".

3.2.2 Results

Table 3 shows the validation results for the updating methods for DIR models with distributions approximated by Gaussian mixture models. $M_{GMM}init$ is the initial model which was trained only with a dataset D_{Train}. The training dataset contains in this case data of about 40 minutes of driving on a rural road with 142918 samples. $M_{GMM}off$ is an offline learned DIR model which was trained with D_{Train} and D_{Train_Online}. The data set D_{Train_Online} contains 167522 samples covering about 46 minutes of driving. $M_{GMM}upd$ is the resulting model of updating $M_{GMM}init$ with D_{Train_Online} and the contained annotation using the updating methods for GMM described in Section 2.2.2.

Table 3: Performance of an initial model, an offline "updated", and the updated model for the evaluation on D_{Test}

Model	REC	FPR	PRE	ACC	F_1-score
$\boldsymbol{M_{GMM}init}$	0.658	0.228	0.100	0.767	0.174
$\boldsymbol{M_{GMM}off}$	0.133	0.032	0.139	0.937	0.136
$\boldsymbol{M_{GMM}upd}$	0.149	0.039	0.127	0.931	0.137

The results from the table show that except for the recall and the F_1-score $M_{GMM}off$ and $M_{GMM}upd$ outperform the initial model on the data set D_{Test} . This could be expected, since the initial model had not seen data from the driver whose data is contained in D_{Test}, while the other models both have seen D_{Train_Online} which contains data from the same driver as in D_{Test}. Nevertheless the data shows that the updated model $M_{GMM}upd$ nearly shows the same performance as $M_{GMM}off$. This is an indicator that the GMMs in both models are very similar and the fixed complexity update for the GMMs is working properly.

Table 4 shows the validation results for the evaluation of the initial model M_{init} and the updated model M_{upd} on the dataset D_{Test}. For the updating procedure the annotations in the update training data D_{Train_Online} were ignored. Instead the online sample generation described in Section 2.2.2 was utilized.

Table 4: Performance of initial and updated model on test data set

Model	REC	FPR	PRE	ACC	F_1-score
$\boldsymbol{M_{init}}$	0.844	0.197	0.055	0.804	0.103
$\boldsymbol{M_{upd}}$	0.292	0.068	0.142	0.908	0.191

The ACC and PRE values of the updated model increase in comparison to the initial model while the FPR value reduces, which indicates a better performance of the updated model. Therefore, the updated model outperforms the initial model on the test set in terms of indicating more seldom a lane change when there is no intention to perform one, which results in lesser false alarms than the initial model. The results validate the requirements R_EN4_model2.2 stating that the model must be able to learn (online) the driver's preferred decisions in specific situations. While here the preferred decision is interpreted as the intention to perform a lane change.

Runtime Performance

Considering the validation of the runtime performance of this enabler an empirical approach is used. The real running time of the Learning of intention from driver is measured on an exemplary system. This running time depends on:

- the model which has to be updated;
- the size of the batch which is processed for updating;
- the amount and type of the distributions that are affected by the data contained in the updating batch.

For example, the initial Gaussian model $\boldsymbol{M_{init}}$ has six categorial distributions and 25 multivariate Gaussian distributions which are subject to updates, while the complex model $\boldsymbol{M_{GMM}init}$ contains 10 categorial distributions and eight distributions with Gaussian mixture models.

On a system equipped with an Intel i5-3570 CPU @ 3.4 GHz the average running time is 88ms for $\boldsymbol{M_{init}}$ and 76ms for $\boldsymbol{M_{GMM}init}$. This validates requirement *R_EN4_model2.9* stating that the update procedure must be sufficiently fast and the updating time is below 500ms. It has to be noticed that this refers to the process of sample generation, data preparation and actual updating of the parameters for a mean amount of about 980 samples.

About 80% of the time is consumed during the sample generation, since the forward-backward inference is quite complex. However, for further developments this point for optimizations. For example, the forward and the backward could be parallelized.

3.2.3 State of the Art

In the automotive domain there are a few approaches which apply online learning to driver models, none of them employs dynamic Bayesian Models. In [9] a manoeuvre forecast for other road users at intersection based on a Bernoulli-Gaussian Mixture Model is described. An update of the model is realized by means of sequential EM. In contrast to our approach, updating of the model while driving and an online sample generation are not covered. The approach presented in [10] employs fuzzy Case-Based Reasoning and Situation-Operator modelling to individualize and learn situation recognition for lane-changes. In contrast to our approach, the initially offline learned models are already individualized for one driver and are then trained further online. The case base might also grow over time leading to an increased time to check for known cases. In [11] GMMs trained via EM are used to model lane-changes and car following behaviour. In order to make the model responsive to individual drivers and behaviour changes the EM training is started again whenever a sufficient amount of new samples is available. Since the retraining consumes many resources the GMMs are retrained only on a certain batch of recent data. In contrast to our approach this leads to the effect that the model only represents recent driving data and ignores older experiences.

3.3 E5.1 - Online risk assessment (OFF + DLR + HMT)

3.3.1 Safety Corridor around Dynamic Objects

Following the plans presented in deliverable D3.6 "Metrics and plan for V&V of the concepts and algorithms in the 3rd cycle", validation of online risk assessment in respect to other traffic participants was performed using a set of independent test data, representing ground truth time-series of traffic situations.

To recapitalize the overall validation process and metric used, let D_{Test} denote the test data, composed by a number of m trials, where each trial j, $j = 1, \dots, m$, is a time-series consisting of a number of n_j data samples $d_j^k, k = 1, \dots, n_j$, describing the ground truth traffic situation at time k.

For each sample d_j^k, and each object v considered for online risk assessment, we used the Vehicle and Situation Modelling Module to predict a sequence of (the current and) future states $p\left(X_{j,v}^{k+\Delta i}, Y_{j,v}^{k+\Delta i}, \Theta_{j,v}^{k+\Delta i} | \boldsymbol{o}^{1:k}\right), i = 0, \dots, \eta_{max}$, and derived the region that included the expected position of the vehicle with a probability of $(1 - \delta_v)$, choosing δ_v such that $\delta_v = 1 - \sqrt[n_V]{(1 - \delta)}$.

Based on this prediction, the online risk assessment component was used to calculate a corresponding set of safety corridors $\boldsymbol{c}^{k:k+\eta_{max}\Delta} = \left(\boldsymbol{c}^{k:k+\Delta}, \boldsymbol{c}^{k+\Delta:k+2\Delta}, \dots, \boldsymbol{c}^{k+(\eta_{max}-1)\Delta:k+\eta_{max}\Delta}\right)$. For each safety corridor $\boldsymbol{c}^{i:i+1}, k \leq i < \eta_{max}$, we then used the current and subsequent samples in the trial corresponding to the resp. temporal interval $[i, i+1]$ and checked for each such sample, whether any object $v \in V$ penetrated the implied safety region defined by the conjunction of the polygons. Denoting such an occurrence as a failure and resp. as a success otherwise, we used the metric

$$CR_{\delta}^{i} = \frac{\#_s}{\#_s + \#_f},$$

representing *the ratio of successes $\#_s$ and the sum of successes $\#_s$ and failures $\#_f$*, in the following referred to as the correct classification rate, for a prediction horizon i and a specific level of δ for assessing the quality of online risk assessment.

As mentioned in deliverable D3.6 “Metrics and plan for V&V of the concepts and algorithms in the 3rd cycle”, the metric has the undesired drawback that the correct classification rate can be arbitrarily increased by (artificially) increasing the area in which another traffic participant is likely to be located for a specific prediction interval, therefore decreasing the size of the overall safe area. To account for this drawback, we therefore calculated the mean area $\bar{A}_{\delta}^{i}$ of each safety corridor for a prediction horizon i and a specific level of δ as a supplementary measure for assessing the quality of online risk assessment.

Repeating the validation process of the second cycle for comparison, we computed these metrics using a temporal step width $\Delta = 1s$ and a maximal number of prediction steps $\eta_{max} = 10$, resulting in a prediction horizon of $\eta_{max}\Delta = 10s$, for five different levels of δ, $\delta_{0.5} = 0.5$, $\delta_{0.25} = 0.25$, $\delta_{0.1} = 0.1$, $\delta_{0.05} = 0.05$, and $\delta_{0.01} = 0.01$, expecting a ratio of $1 - \delta$ for the correct classification respectively. As requirement R_EN5_alg1.5 requires a correct rate of classification above 90% to be fulfilled, we define the requirement to be fulfilled for a specific temporal interval i and level of δ, when:

$$CR_{\delta}^{i} > 0.9(1 - \delta).$$

Finally, for an empirical approach for assessing the computational performance of the online risk assessment, we measured the runtime performance for the construction of safety corridors and trajectory assessment using a temporal step width $\Delta = 1s$ and a maximal number of prediction steps $\eta_{max} = 10$ for different numbers of vertices per corner polygon.

3.3.1.1 Dataset for Validation

As the validation process requires the knowledge of ground truth, we performed the validation on simulator data. To allow for an easier comparison with the evaluation results obtained during the second cycle, we reused the same test set D_{Test} as previously described and used in Deliverable D3.5 "Concepts and algorithms incl. V&V results from 2nd cycle".

Due to the test set arising from a simulator study (a detailed description of the experiment is provided in deliverable D2.4 "Sensor Platform and Models incl. V&V results from 2nd cycle") in which the traffic flow was automatically controlled by a traffic simulation, the resulting behaviour of traffic participants in the vicinity of the TeamMate vehicle is highly predictable and unrealistic, leading to overly optimistic results. For a more realistic assessment of humanly controlled traffic participants, we therefore additionally perform our validation on the safety regions for the humanly controlled "TeamMate" vehicle.

3.3.1.2 Results

As the primary result of the validation, Table 5 shows the correct classification rate CR_{δ}^{i} and mean area $\bar{A}_{\delta}^{i}$ for the humanly controlled vehicle for the different temporal intervals i (corresponding to a temporal interval $[k+(i-1)s{:}\,k+is]$) and different levels of δ, using a number of vertices $n_P = 100$. Underlined values indicate that the ratio is above $0.9(1-\delta)$, therefore fulfilling R_EN5_alg1.5, the total sum of successes $\#_s$ and failures $\#_f$ is denoted as # (in thousands). The construction of the safety corridors is based on a number of $n_P = 100$ vertices per corner polygon. As apparent, the current version of online risk assessment fulfils requirement R_EN5_alg1.5 for the whole prediction horizon for $\delta_{0.5}$, $\delta_{0.25}$, and $\delta_{0.1}$, up to eight seconds for $\delta_{0.05}$ and up to six seconds for $\delta_{0.01}$.

Table 5: Mean ratio of successes $\#_s$ and the sum of successes $\#_s$ and failures $\#_f$ and mean area for human participants ($\Delta = 1s, \eta_{max} = 10, n_P = 100$).

i	#	$CR^i_{\delta_{0.5}}$	$\bar{A}^i_{\delta_{0.5}}$	$CR^i_{\delta_{0.25}}$	$\bar{A}^i_{\delta_{0.25}}$	$CR^i_{\delta_{0.1}}$	$\bar{A}^i_{\delta_{0.0}}$	$CR^i_{\delta_{0.05}}$	$\bar{A}^i_{\delta_{0.05}}$	$CR^i_{\delta_{0.01}}$	$\bar{A}^i_{\delta_{0.01}}$
1	8528	$\underline{0.994}$	73.58	$\underline{0.999}$	81.75	$\underline{1.000}$	90.08	$\underline{1.000}$	95.45	$\underline{1.000}$	106.2
2	8521	$\underline{0.937}$	86.88	$\underline{0.957}$	101.5	$\underline{0.969}$	116.7	$\underline{0.974}$	126.7	$\underline{0.982}$	146.9
3	8515	$\underline{0.879}$	95.42	$\underline{0.916}$	114.8	$\underline{0.933}$	135.3	$\underline{0.940}$	149.0	$\underline{0.949}$	177.2
4	8508	$\underline{0.832}$	105.6	$\underline{0.884}$	130.6	$\underline{0.907}$	157.6	$\underline{0.917}$	175.6	$\underline{0.928}$	213.4
5	8501	$\underline{0.794}$	117.5	$\underline{0.860}$	145.0	$\underline{0.886}$	183.3	$\underline{0.897}$	206.5	$\underline{0.910}$	255.4
6	8495	$\underline{0.765}$	130.9	$\underline{0.829}$	169.9	$\underline{0.868}$	212.7	$\underline{0.881}$	241.8	$\underline{0.896}$	303.5

7	8488	0.740	146.0	0.822	193.6	0.854	246.2	0.867	282.2	0.883	358.8
8	8481	0.720	163.1	0.807	220.4	0.842	284.3	0.856	328.2	0.872	422.0
9	8474	0.703	182.2	0.794	250.6	0.832	327.4	0.850	380.4	0.864	494.1
10	8468	0.687	203.7	0.783	284.7	0.823	376.4	0.837	439.8	0.857	576.4

As a comparison for the quality of the online risk assessment when reducing the number of vertices per corner polygon n_P, Table 6 shows the result to $n_P = 10$. Here, R_EN5_alg1.5 is still fulfilled for the whole prediction horizon for $\delta_{0.5}$, $\delta_{0.25}$, and $\delta_{0.1}$, but only up to seven seconds for $\delta_{0.05}$ and up to five seconds for $\delta_{0.01}$. As such, we would prefer the original number of vertices $n_P = 100$, if allowed by the required execution time.

Table 6: Mean ratio of successes $\#_s$ and the sum of successes $\#_s$ and failures $\#_f$ and mean area for human participants ($\Delta = 1s, \eta_{max} = 10, n_P = 10$).

i	#	$CR^i_{\delta_{0.5}}$	$\bar{A}^i_{\delta_{0.5}}$	$CR^i_{\delta_{0.25}}$	$\bar{A}^i_{\delta_{0.25}}$	$CR^i_{\delta_{0.1}}$	$\bar{A}^i_{\delta_{0.0}}$	$CR^i_{\delta_{0.05}}$	$\bar{A}^i_{\delta_{0.05}}$	$CR^i_{\delta_{0.01}}$	$\bar{A}^i_{\delta_{0.01}}$
1	8528	0.994	73.26	0.999	81.25	1.000	89.39	1.000	94.64	1.000	105.1
2	8521	0.935	86.26	0.956	100.5	0.968	115.4	0.973	125.0	0981	144.7
3	8515	0.876	94.50	0.914	113.3	0.932	133.3	0.939	146.5	0.948	173.8
4	8508	0.825	104.4	0.881	128.7	0.905	154.8	0.912	172.3	0.926	208.7
5	8501	0.786	116.1	0.855	146.7	0.883	180.0	0.894	202.4	0.919	249.4
6	8495	0.754	129.3	0.834	167.2	0.865	208.7	0.878	236.8	0.894	296.3
7	8488	0.728	144.2	0.816	190.4	0.850	241.4	0.864	276.2	0.881	350.0

8	8481	0.708	161.0	0.801	216.7	0.838	278.5	0.852	320.9	0.870	411.3
9	8474	0.690	179.8	0.787	246.3	0.827	320.6	0.842	371.8	0.861	481.3
10	8468	0.675	200.9	0.776	279.6	0.818	368.2	0.834	429.5	0.854	561.1

For the sake of comparison with the results reported in D3.5 “Concepts and algorithms incl. V&V results from 2nd cycle”, Table 7 shows the results for the other traffic participants, controlled by the driving simulation for $n_P = 100$, Table 8 shows the results for $n_P = 10$. As expected, R_EN5_alg1.5 is fulfilled for all δ considered for the complete prediction horizon.

Table 7: Mean ratio of successes $\#_s$ and the sum of successes $\#_s$ and failures $\#_f$ and mean area for automatically controlled traffic participants ($\Delta = 1s, \eta_{max} = 10, n_P = 100$).

i	#	$CR^i_{\delta_{0.5}}$	$\bar{A}^i_{\delta_{0.5}}$	$CR^i_{\delta_{0.25}}$	$\bar{A}^i_{\delta_{0.25}}$	$CR^i_{\delta_{0.1}}$	$\bar{A}^i_{\delta_{0.1}}$	$CR^i_{\delta_{0.05}}$	$\bar{A}^i_{\delta_{0.05}}$	$CR^i_{\delta_{0.01}}$	$\bar{A}^i_{\delta_{0.01}}$
1	20746	0.994	88.66	0.994	96.35	0.995	104.2	0.995	109.3	0.995	119.7
2	20176	0.990	108.9	0.991	122.6	0.993	136.7	0.993	146.0	0.994	165.3
3	19616	0.987	123.4	0.988	141.9	0.989	161.3	0.989	174.2	0.990	201.4
4	19067	0.983	140.5	0.985	164.7	0.987	190.4	0.987	207.7	0.988	244.2
5	18531	0.979	160.3	0.983	191.1	0.985	224.1	0.986	246.4	0.988	294.0
6	18006	0.978	182.7	0.983	221.1	0.985	262.5	0.987	290.6	0.988	350.8
7	17490	0.979	207.9	0.985	254.9	0.987	305.9	0.988	340.8	0.990	415.5
8	16984	0.980	236.1	0.987	293.0	0.989	354.9	0.989	397.4	0.990	488.8

9	16491	0.981	267.6	0.987	335.6	0.989	410.0	0.990	461.2	0.991	571.5
10	16007	0.981	302.7	0.987	383.3	0.989	471.7	0.990	532.7	0.991	664.6

Table 8: Mean ratio of successes $\#_s$ and the sum of successes $\#_s$ and failures $\#_f$ and mean area for automatically controlled traffic participants ($\Delta = 1s, \eta_{max} = 10, n_P = 10$).

i	#	$CR^i_{\delta_{0.5}}$	$\bar{A}^i_{\delta_{0.5}}$	$CR^i_{\delta_{0.25}}$	$\bar{A}^i_{\delta_{0.25}}$	$CR^i_{\delta_{0.1}}$	$\bar{A}^i_{\delta_{0.0}}$	$CR^i_{\delta_{0.05}}$	$\bar{A}^i_{\delta_{0.05}}$	$CR^i_{\delta_{0.01}}$	$\bar{A}^i_{\delta_{0.01}}$
1	20746	0.994	88.14	0.994	95.66	0.995	103.3	0.995	108.3	0.995	118.5
2	20176	0.990	107.9	0.991	121.2	0.992	134.9	0.993	144.0	0.994	162.7
3	19616	0.987	121.8	0.988	139.7	0.989	158.5	0.989	171.0	0.990	197.3
4	19067	0.982	138.5	0.985	161.9	0.986	186.7	0.987	203.4	0.988	238.6
5	18531	0.978	157.8	0.982	187.6	0.984	219.5	0.986	241.1	0.987	286.9
6	18006	0.977	179.8	0.982	217.0	0.985	257.0	0.986	284.1	0.988	342.1
7	17490	0.977	204.5	0.954	250.1	0.987	299.3	0.988	332.9	0.989	404.9
8	16984	0.978	232.2	0.986	287.2	0.988	347.0	0.989	388.0	0.990	476.0
9	16491	0.979	263.0	0.987	328.8	0.989	400.6	0.990	450.0	0.991	556.2
10	16007	0.979	297.3	0.987	375.3	0.989	460.7	0.990	519.5	0.991	646.4

Runtime Performance

For an empirical approach for measuring the computational performance of the online risk assessment, we measured the runtime performance for the construction of safety corridors and trajectory assessment using a temporal step width $\Delta = 1s$ and a maximal number of prediction steps $\eta_{max} = 10$ for different numbers of vertices per corner polygon, more specifically $n_P = 10000$, $n_P = 1000$, $n_P = 100$, and $n_P = 10$.

Execution times were calculated as the average of 27000 example executions using an i7-6700 CPU @ 3.40GHz, 16GB desktop computer, running a Microsoft Windows 10 64-Bit operation system. The algorithms were compiled as 64-Bit applications using Visual Studio 2017 and we measured individual execution times in nanoseconds using the high_resolution_clock provided by the std::chrono library.

To allow for a better extrapolation to the usually variable number of considered objects n_V in the vicinity of the TeamMate vehicle, we limited the assessment to a *single* object by only measuring the construction of the safety corridor for the TeamMate itself. These results measure the construction of safety corridors and safety assessment in isolation, and do not include the necessary time for the required prediction of the spatial and temporal evolution of the traffic scene or graphical user interfaces.

The results for the construction of the safety corridors are shown in Table 9. As apparent, the original selected number of vertices $n_P = 100$ per corner polygon still allows for a construction of the complete safety corridor below 1ms per considered object in the vicinity of the TeamMate vehicle. Extrapolating these results, this would allow for the construction of safety corridors in the presence of up to $n_V = 20$ considered objects within the duration of approx. 10ms, which fits well into the intended frequency of execution of 10Hz.

Table 9: Average computation time for the construction of safety corridors for a single object for different number of vertices per corner polygon.

Number of vertices per corner polygon	Average computation time (ms)
10000	64.109
1000	5.168
100	0.525
10	0.111

To assess the runtime performance of trajectory assessment, we created random trajectories, consisting of $n_R = 100$ points over a temporal length of $10s$ by sampling the coordinates and yaw angles from a zero mean normal distribution with standard deviation $\sigma = 100$ and measured the average time for the assessment of this trajectory when using different number of vertices per corner polygon. To assess the worst-case scenario, trajectory assessment was not aborted once the trajectory was assessed as unsafe. As for the construction of the safety corridors, we measured the time using the safety corridors of a single object. The results are shown in Table 10.

Table 10: Average computation time for trajectory assessment against the safety corridors of a single object for different number of vertices per corner polygon.

Number of vertices per corner polygon	Average computation time (ms)

10000	3.604
1000	0.368
100	0.046
10	0.012

Extrapolating these results, the original selected number of vertices $n_P = 100$ per corner polygon would allow for trajectory assessment in the presence of up to $n_V = 20$ considered objects within the duration of just 1ms.

Privacy Issues

Finally, addressing requirement R_EN5_alg1.7, the algorithms for online risk assessment in respect to other traffic participants do not process or retrieve any personal data of the driver.

3.3.1.3 State of the Art

In the context of intelligent driving systems, the notion of risk assessment is commonly associated with the idea “that a situation may be dangerous for the driver, i.e. may result in harm or injury” [12]. Approaches for risk assessment have been broadly classified into two families [12], approaches that relate risk to unexpected behaviour of traffic participants and approaches that relate risk with potential physical collisions between entities (e.g., vehicles) in the traffic scene.

Concerning the former, [13] represented the nominal behaviour of driver by Gaussian mixture models which could then be used to detect “unusual” situations by assessing the likelihood of a driver’s behaviour. [14] proposed to compare expectations about a driver's behaviour with estimated intentions, which allows the computation of the probability of a mismatch between expectation and intentions to indicate risk. Unfortunately, such approaches only allow to assess whether a situation is critical, but provide no additional information concerning the exact circumstances.

Approaches that associate risk with potential collisions usually combine a prediction of future trajectories for all entities in the traffic scene with the assessment of these trajectories to detect potential collisions [12]. Many of such approaches in the literature focus on "Time-To-X" measures, e.g., the "Time-To-Collision", representing the remaining time to a collision under the assumption of constant velocities, or "Time-To-React" measures, representing e.g., the remaining time to initiate a braking or steering manoeuvre, which can be used as an indication of what action should be taken or to identify the least dangerous intervention manoeuvre [12].

For assessing whether a future trajectory of the driver or the automation is safe, the most popular measure of risk is based on the notion of the probability of collision [15], [16], [17], [18] based on the predicted trajectories of the driver and other traffic participants, which we adopted for online risk assessment in AutoMate. Assessing the probability of collision under uncertainty requires the integration over all possible trajectories and dimensions of all traffic participants [15], [16], [17], [18]. Due to the unsolvable nature of this integration in closed form, one must usually resort to Monte Carlo methods, limiting the real-time capacity of such approaches. Unfortunately, actual computation times are seldom reported. [15] limited the prediction horizon to a maximum of three seconds to achieve real-time capacity. [17] considered predictions up to four seconds by limiting the number of samples to a very low number of just 100. [18] proposed a novel approach, suitable if all vehicles are perfectly aligned with the road, that enables online risk assessment via testing for collisions of pairs of trajectories with average computation times of approx. just 0.007ms for each tested pair. Unfortunately, when dealing with uncertainties, one must once again resort to Monte Carlo methods, cancelling the computational advantages.

Although based on the same probability of collision, our approach for online risk assessment differs from these approaches by transforming the prediction of the temporal and spatial evolution of the traffic scene into polygonal safety corridors over time spans, removing the need for integration. The transformation comes with the caveat of an inability to provide an "exact" probability of collision for a given trajectory. Our safety corridors only provide the upper bound on the probability of collision, allowing the assessment whether a trajectory is safe in relation to a desired probability of collision. We argue however that an "exact" assessment is unnecessary, if it finally used to test it against a threshold, in which case this inability is of no effect. When testing against a threshold is sufficient, our approach allows the assessment of trajectories without the need for Monte Carlo methods, allowing for a much greater prediction horizon (e.g., 10s) and traffic density than the state of the art.

3.3.2 Safety Corridor between Road Boundaries

As documented in D3.5, in the second cycle we used *Intersection Over Union* (*IOU*) to quantify the quality of the extracted safety corridor. Therefore, we consider the same metric during third cycle for validation of the extracted safety corridor between road boundaries. *IOU* measures the amount of overlap with respect to the ground truth information, as defined in Equation 1:

Equation 1: Definition of Intersection over Union.

$$IOU = \frac{TP}{TP + FP + FN}$$

where *TP*, *FP* and *FN* are *true positive*, *false positive* and *false negative*, respectively.

During D3.6, we have introduced *Precision* and *Recall* as new metrics to evaluate the performance of the algorithm that asses planned trajectories for risk. These metrics specifically target the verification/validation of requirements *R_EN5_alg1.5* and *R_EN5_alg1.6*.

As defined in Equation 2, *Precision* provides the positive predictive rate and *Recall* measures the ability to classify the relevant instances.

Equation 2: Definition of Precision and Recall.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

Finally, with respect to runtime of the implemented algorithm, the average execution time is provided.

3.3.2.1 Dataset for Validation

Validation of our algorithm was performed on test data provided by Vedecom. The structure of the *Digital Map* representing road networks was specified in a common agreement between DLR and Vedecom. To give an impression of the data format involved, Figure 7 shows a screenshot of the *Digital Map* information available for the component. The single fields are documented in Table 11 below this figure.

VEDECOM_MAP.Excel.xlsx

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	id	nblanes	Lane1								Lane2							
2			Avalability	x	y	half width	Speed max	right marking	left marking	heading	Avalability	x	y	half width	Speed max	right marking	left marking	heading
3	1	3	1	433209,119	5404151,284	1,75	30	1	0	0	1	433209,119	5404151,284	1,75	30	0	1	0
4	2	3	1	433209,08	5404151,077	1,75	30	1	0	-100,669783	1	433209,08	5404151,077	1,75	30	0	1	-100,669783
5	3	3	1	433209,027	5404150,847	1,75	30	1	0	-102,976422	1	433209,027	5404150,847	1,75	30	0	1	-102,976422
6	4	3	1	433208,977	5404150,631	1,75	30	1	0	-103,033356	1	433208,977	5404150,631	1,75	30	0	1	-103,033356
7	5	3	1	433208,928	5404150,43	1,75	30	1	0	-103,700399	1	433208,928	5404150,43	1,75	30	0	1	-103,700399
8	6	3	1	433208,873	5404150,203	1,75	30	1	0	-103,619771	1	433208,873	5404150,203	1,75	30	0	1	-103,619771
9	7	3	1	433208,806	5404149,949	1,75	30	1	0	-104,776868	1	433211,7268	5404149,264	1,75	30	0	1	-104,776868
10	8	3	1	433208,752	5404149,741	1,75	30	1	0	-104,553573	1	433211,6655	5404149,026	1,75	30	0	1	-104,553573
11	9	3	1	433208,689	5404149,522	1,75	30	1	0	-106,049005	1	433211,5959	5404148,78	1,75	30	0	1	-106,049005
12	10	3	1	433208,624	5404149,289	1,75	30	1	0	-105,587505	1	433211,5253	5404148,526	1,75	30	0	1	-105,587505
13	11	3	1	433208,554	5404149,046	1,75	30	1	0	-106,069884	1	433211,4501	5404148,263	1,75	30	0	1	-106,069884
14	12	3	1	433208,479	5404148,794	1,75	30	1	0	-106,574008	1	433211,3682	5404147,986	1,75	30	0	1	-106,574008
15	13	3	1	433208,403	5404148,533	1,75	30	1	0	-106,234912	1	433211,2884	5404147,712	1,75	30	0	1	-106,234912
16	14	3	1	433208,319	5404148,265	1,75	30	1	0	-107,402704	1	433211,1977	5404147,421	1,75	30	0	1	-107,402704
17	15	3	1	433208,233	5404147,991	1,75	30	1	0	-107,425436	1	433211,1082	5404147,135	1,75	30	0	1	-107,425436
18	16	3	1	433208,147	5404147,707	1,75	30	1	0	-106,847199	1	433211,0193	5404146,841	1,75	30	0	1	-106,847199
19	17	3	1	433208,087	5404147,514	1,75	30	1	0	-107,269468	1	433210,9566	5404146,639	1,75	30	0	1	-107,269468

Figure 7: Digital Map Information from Vedecom.

Table 11: Description of the Digital Map's Fields

ID	Road point ID
Nblanes	Number of lanes at the particular road point
Availability	Availability of lane in Driving direction • 0 –Not available • 1 – Available

Half width(m)	Half the entire width of the lane
Speed max(km/h)	Associated max speed for particular lane
Right/Left marking	Right and left lane marking associated to the lane • 1-Solid Lane marking • 0-Dashed Lane marking
Heading(rad)	Heading of the each road point at the particular Road ID

For validating the safety corridor extraction between road boundaries as specified in the D3.5, the *IOU* metric is used on the Vedecom test data. This data set, denoted as $D^{Vedecom}$, was recorded within the scope of the *Digital Map*. It contains the following variables:

- TeamMate's position in UTM
- Unix timestamp for synchronization
- heading of the TeamMate car
- absolute velocity of the vehicle
- uncertainties in the positon of the vehicle

Further, to show the sensitivity of the safety corridor extraction between road boundaries, towards the uncertainty of the vehicle position and requested *POC*, we synthesised a new test data ($D^{synthetic}$) by varying the position uncertainty in the Vedecom test data and different *POC* requirements. The test trajectories were generated by fitting random extrapolation functions between reference points. Considering both the test data allowed us to perform an unbiased validation of the component.

As specified in D3.6 we have proposed *Precision* and *Recall* as the new metrics to provide the validation measures for assessing the risk of planned trajectories.

3.3.2.2 Results

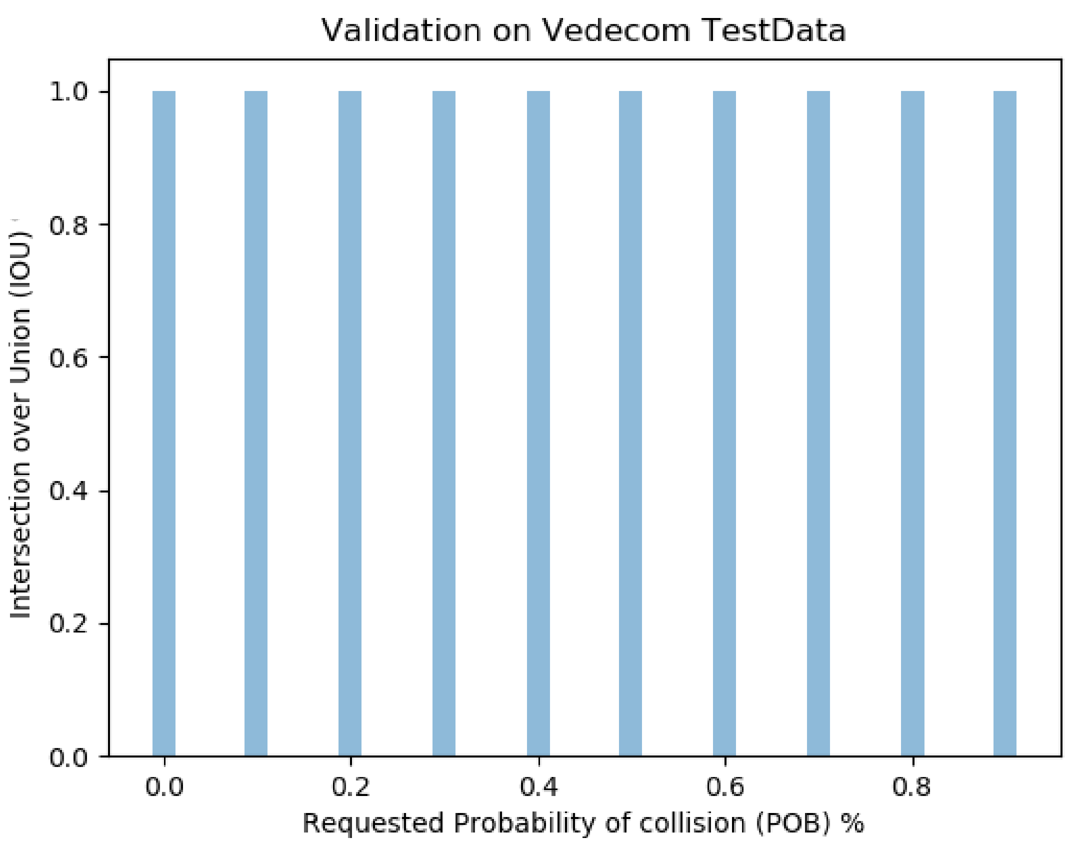

Figure 8: Validation results on Vedecom Test data ($D^{Vedecom}$), *IOU* vs. *POC*.

Due to very low lateral uncertainty (.02 m) in the position of the TeamMate's vehicle, the resulting shift value, which determines the amount of shifting required to obtain risk assessed safety corridor under the requested *POC* results is negligible. To recap the shift value is the result of inverse *CDF* for a particular quantile, which is computed using the requested *POC*. The *CDF* is modelled using the vehicle position and the uncertainty in the position of the vehicle. Therefore, the shift due to the uncertainty at different *POC*'s is very small, this explains why the *IOU* is nearly 100 percent for different *POC*.

As stated earlier, to perform an unbiased validation, new test data was synthesised on top of the test data received from Vedecom. Where the uncertainty involved in the position of the vehicle varied, which is the prime factor that determines the degree of shifting necessary to satisfy for different *POC*'s and to obtain risk assessed safety corridor. Figure 17 shows the effect *of *IOU* with respect to the uncertainty involved in the position of the vehicle.

From Figure 9, we can notice that the *IOU* at *POC* ~0.35 varies for different uncertainty in the vehicle position. In comparison to the results shown in Figure 8, *IOU* is not anymore 100 percent. This clearly explains the component is sensitive to the position uncertainty of the TeamMate vehicle. Referring to the requirement table in D1.5 the algorithm implemented for extracting safety corridor between road boundaries by taking into account the uncertainties and requested *POC* ensures the results to be always safe and acceptable. This fulfils the requirement *R_EN5_alg1.2*.

To show the sensitivity of the GPS information, we had to consider different uncertainties to emphasise the effect of extracted safety corridor information with respect to the uncertainties of the received GPS signal. However, in the obtained test data (real data), the uncertainty in the GPS position was negligible. Henceforth, the IOU remains almost same for different requested POC.

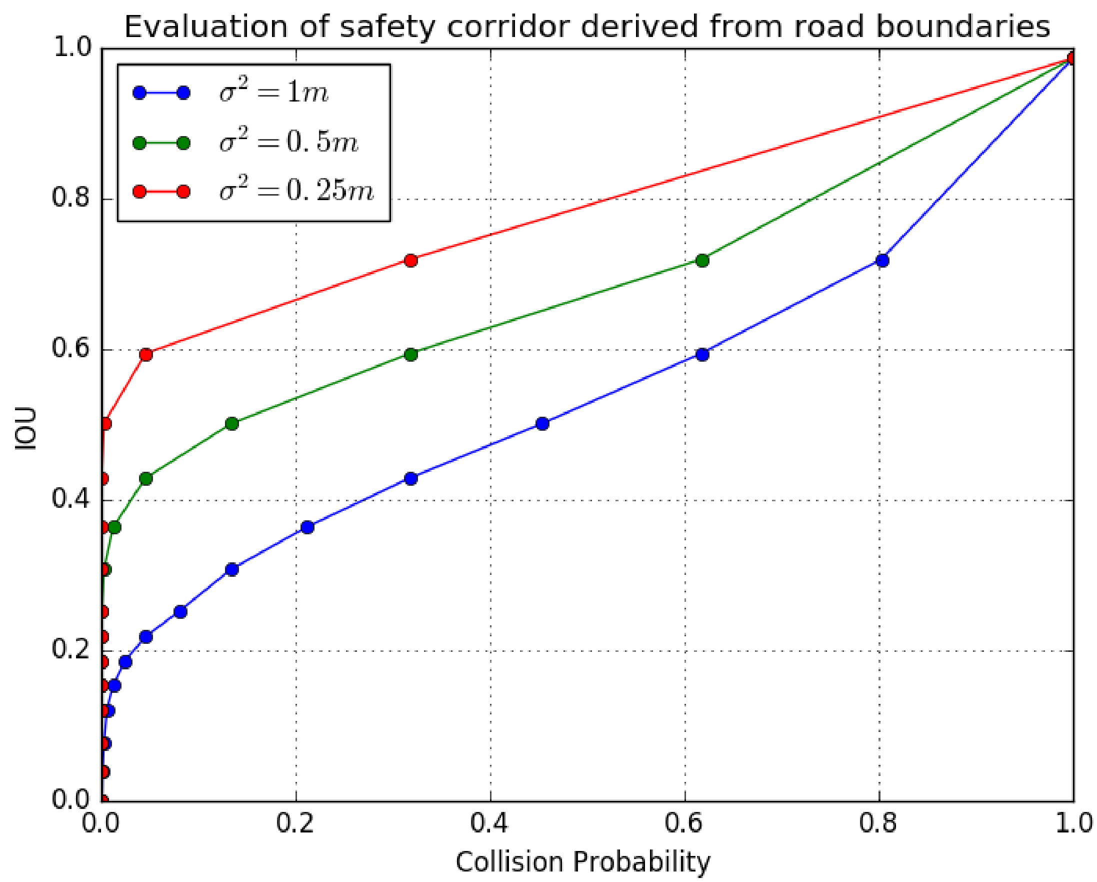

Figure 9: *IOU* vs. *POC* plot on synthesised data ($D^{synthetic}$). The blue, green and red curves denote the *IOU*s obtained for different uncertainties in vehicle position at 1 m, .5 m, and .25 m, respectively.

Regarding the online risk assessment of the planned trajectories as described in the section Safety Corridor between Road Boundaries, the performance metrics of the assessment algorithm is shown in Table 12. *Precision* and *Recall* of the trajectory risk assessment is always 100 percent, due to the extracted safety corridor's sensitivity to the uncertainty of the vehicle position.

From the test data we have seen that the uncertainty of the vehicle position is relatively small and henceforth the extracted safety corridor between the road boundaries is almost the same as the road boundary corridor. Therefore, all the planned trajectories (i.e. the *Ground Truth*) that are safe will be classified as safe by the assessment algorithm, which results in very high *Precision*. And also the 100 percent *Recall* justifies all the planned trajectories are covered when assessing or risk at particular timestamp.

Table 12: Precision and Recall by the Online risk assessment for planned trajectories.

Precision	1.0.
Recall	1.0

Runtime Performance

Release version of the component was measured for its execution time on the development machine for ensuring requirement *R_EN5_alg1.3* from D1.5. The component was able to extract a safety corridor and asses the planned trajectories at a frequency of ~15 milliseconds on an `Intel core i7-6820HQ cpu @ 2.50ghz`.

Finally, addressing requirement *R_EN5_alg1.7*, the algorithm pipeline to extract the safety boundary corridor between road boundaries and risk assessment of planned trajectories, does not process or retrieve any personal data of the driver.

3.3.2.3 State of the Art

The fundamental functionality of online risk assessment methods is the safe navigation of the Autonomous Vehicles (AV), both for single driving as well as when interacting with other road users. This is mainly performed by the Trajectory Planner [19].

Via the data collected by the sensors installed in the vehicle and infrastructure, and topological information (e.g. from open source maps), both dynamic and contextual (road information) inputs are obtained to ensure providing reliable data to the trajectory planner.

As a framework to create an inference out of these inputs, Dynamic Bayesian Networks are often used. This method is able to overcome uncertainties of the data delivered by these embedded systems such as noise, delays and missing input. Furthermore, so called surrogate safety measures are taken into account to evaluate the criticality of both the current status of the Ego vehicle with respect to the environment and the purposed paths by the Trajectory Planner [20].

Building upon this state of the art, our computation of the safety corridor works as follows. Using a wanted *Probability of Collision* (*POC*) and the uncertainty of the position from the GPS receiver, a normal distribution is modelled. Considering the *POC* as a weight to influence the uncertainty coefficient, a safety corridor for the Ego Vehicle is assessed.

As an enhanced measure for estimating the risk of the purposed trajectories of the *Trajectory Planner*, this generated corridor (based on the contextual information above mentioned) is later on used together with the *Time to Collision* (*TTC*) and *Distance to Collision* (*DTC*) surrogate safety metrics. This allows establishing a likelihood hierarchy among the selected paths.

[21] have also suggested such a modification of the traditional approach to trajectory planning. In addition to using *TTC* alone as a risk parameter, each node of the set of possible future trajectories is taken as a reference to compute the surrogate measures.

As [20] further mention, innovative research should try to overcome the limitations of *TTC*, where linear references are taken into account to estimate the risk. This might lead to uncertainties and false positives involving the collision avoidance system. Currently ongoing as well as planned improvements to the *Safety Corridor between Road Boundaries* submodule are addressing this issue. This involves a linear discretizing of the planned curved paths, to yield a more accurate *TTC* estimation.

4 Conclusions

All enablers in WP3 (E4.1, E4.2 and E5.1) have been developed to implement the A2H cooperation, both in action and in perception.

The planning and execution of trajectories enabler has experienced big change to provide well planned trajectories incorporating driver models while also being real time capable.

The module for Learning of intention from driver has been greatly improved to handle updating of complex distributions present in the DIR model and to generate labelled samples required for processing of observed data while driving.

Finally, the concept for the online risk assessment was modified to directly assess the safety of provided trajectories with respect to a boundary Safety Corridor and surrounding dynamic object while keeping a low computation time.

All enabler are ready for the final integration into the demonstrators of the third cycle.

Lessons learnt

- E4.2
    - To address the Cold start problem related to the Learning of intention from driver during the first project cycle it proved helpful for testing and validating the concept to have already

some driving and lane change data obtained during the former EU project HoliDes.

- For a real world application it could be expected that the TeamMate car is used by multiple drivers. This case is currently not handled by Learning of Intention from driver but there are basically two options, either to adapt a single driver model rapidly to a new driver or to store multiple driver models. The first option would require that new evidence is weighted much higher than older observations, or that only recent observations from a certain time frame are considered for the creation of the model, like the approach from [11]. However, this would also mean that old knowledge becomes irrelevant in a way that model parameters for a previous driver basically have to be relearned after another driver has used the car. The second option would enable the system to remember previous drivers but would also require some kind of driver identification. The most simple driver identification could be implemented by using separate keys for individual drivers, which is already used by manufactures to store individual settings for, e.g., comfort functions. Other options would be to give the driver a possibility to register at the system, e.g., via a username or by using a smartphone, to identify the driver by the driving behaviour [22] or by sensors inside the car, e.g., the camera of the TeamMate driver monitoring system.

- E5.1

- For the development and simulation in the first cycle, synthetic data had been used. This data was then replaced by real data recorded by the partner VED on their test track.

5 References

[1] M. Treiber, A. Hennecke und D. Helbing, „Congested traffic states in empirical observations and microscopic simulations,“ *Physical Review E,* Bd. 62, Nr. 2, pp. 1805-1824, 2000.

[2] M. Graf, O. Speidel, J. Ziegler und K. Dietmayer, „Trajectory Planning for Automated Vehicles using Driver Models,“ in *21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[3] K. Murphy, „Filtering, Smoothing and the Junction Tree Algorithm,“ 1999.

[4] S. Russell und P. Norvig, Artificial Intelligence A Modern Approach 3rd Edition, Upper Saddle River, New Jersey: Pearson Education, 2010.

[5] P. Smyth, D. Heckerman und M. I. Jordan, „Probabilistic Independence Networks for Hidden Markov Probability Models,“ *Neural Computation,* Bd. 2, Nr. 9, pp. 227-269, 1997.

[6] O. Arandjelovic und R. Cipolla, „Incremental Learning of Temporally-Coherent Gaussian Mixture Models,“ in *Proceedings of the British Machine Vision Conference*, 2005.

[7] A. S. H. Bashar und J. Q. Gan, „Sequential EM for Unsupervised Adaptive Gaussian Mixture Model Based Classifier,“ in *Machine Learning and Data Mining in Pattern Recognition*, 2009.

[8] C. Büskens und D. Wassel, „The ESA NLP Solver WORHP,“ in *Modeling and Optimization in Space Engineering*, Springer, 2012, pp. 85-110.

[9] J. Wiest, M. Karg, F. Kunz, S. Reuter, U. Kressel und K. C. J. Dietmayer, „A probabilistic maneuver prediction framework for self-learning vehicles with application to intersections,“ *2015 IEEE Intelligent Vehicles Symposium (IV),* pp. 349-355, 2015.

[10] A. Sarkheyli-Hägele und D. Söffker, „Online learning for an individualized lane-change situation recognition system applied to driving assistance,“ in *IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*, Savannah, GA, 2017.

[11] V. Butakov, *Personalized driver assistance systems based on driver/vehicle models,* Los Angeles,

California: University of Southern California, 2014.

[12] S. Lefèvre, D. Vasquez und C. Laugier, „A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH Journal,* pp. 1-14, 2014.

[13] J. Salas, H. Jimanez, J. Gonzalez und J. Hurtado, „Detecting unusual activities at vehicular intersections," *Proceedings of the IEEE international conference on robotics and automation,* pp. 864-869, 2007.

[14] S. Lefèvre, C. Laugier und J. Ibanez-Guzmán, „Risk Assessment at Road Intersections: Comparing Intention and Expectation," *Proceedings of the IEEE Intelligent Vehicles Symposium,* pp. 165-171, 2012.

[15] M. Schreier, Bayesian Environment Representation, Prediction, and Criticality Assessment for Driver Assistance Systems, Technische Universität Darmstadt: Fachbereich Elektrotechnik und Informationstechnik, 2015.

[16] G. Xie, X. Zhang, H. Gao, L. Qian, J. Wang und U. Ozguner, „Situational Assessments Based on Unertainty-Risk Awareness in Complex Traffic Scenarios," *Sustainability,* pp. 1-17, 2017.

[17] A. Houénou, P. Bonnifait und V. Cherfaoui, „Risk assessment for collision avoidance systems," *Proceedings of the IEEE 17th International Conference on Intelligent Transportation Systems (ITSC 2014),* pp. 386-391, 2014.

[18] A. Lawitzky, D. Wollherr und M. Buss, „Maneuver-based Risk Assessment for High-Speed Automotive Scenarios," *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS),* pp. 1-6, 2012.

[19] C. Katrakazasa, M. Quddusa und W.-H. Chen, „A new methododolgy for collision risk assessment of autonomous vehicles," in *Transportation Research Board 96th Annual Meeting*, Washington, D.C., 2017.

[20] C. Katrakazas, M. Quddus, W.-H. Chen und 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,* Nr. 60, pp. 416-442, 2015.

[21] L. Garrote, C. Premebida, M. Silva und J. N. Urbano, „An RRT-based navigation approach for mobile

robots and automated vehicles," in *12th IEEE International Conference on Industrial Informatics (INDIN)*, 2014.

[22] D. Hallac, A. Sharang, R. Stahlmann und Lamprecht, „ Driver Identification Using Automobile Sensor Data from a Single Turn," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems* , 2016.