Real-world Test Drive Vehicle Data Management System for Validation of Automated Driving Systems

Lars Klitzke[1], Carsten Koch[1], Andreas Haja[1] and Frank Köster[2]
[1]*Hochschule Emden/Leer, University of Applied Sciences, Department of Electronics and Informatics, Emden, Germany*
[2]*German Aerospace Center (DLR), Institute of Transportation Systems, Braunschweig, Germany*

Keywords: Scenario Mining, Automated Driving Functions, Validation, Large-scale Test Drives, Data Management System.

Abstract: For the validation of autonomous driving systems, a scenario-based assessment approach seems to be widely accepted. However, to verify the functionality of driving functions using a scenario-based approach, all scenarios that may be relevant for the validation have to be identified. Real-world test drives are mandatory to find relevant and critical scenarios. However, the identification of scenarios and the management of the captured data requires computational assistance to validate driving functions with reasonable effort. Therefore, this work proposes a highly-modularised multi-layer Vehicle Data Management System to manage and support analysing large-scale test campaigns for the scenario-based validation of automated driving functions. The system is capable of aggregating the vehicle sensor data to time-series of scenes by utilising temporal discretisation. Those scenes will be enriched with information from various external sources, providing the foundation for efficient scenario mining. The practical usefulness of the proposed system is demonstrated using a real-world test drive sequence, by finding lane-change scenarios and evaluating an onboard system.

1 INTRODUCTION

In the last decades, the effort for developing new Driver Assistance Systems (DAS) and improving available ones has significantly increased, aiming at an overall more comfortable driving experience. In particular, the domain of DAS validation has gained much attention in the last years. Due to the ever-increasing complexity of designing and testing DAS and the need to manage this complexity economically, recent and current research projects aim to provide methodologies, methods and tools (Haja et al., 2017)(Winner et al., 2018) to reduce the effort for the validation or, in particular, enable to approve automated driving functions w.r.t to functional safety standards such as the ISO26262.

Technical advances in the field of computer graphics and computer simulation during the last decades paved the way for new testing methods to master the growing complexity of driver assistance systems. With more sophisticated models of real-world components becoming available, testing shifted from the real world to the virtual world. This is due to the fact that simulations allow to conduct risky manoeuvres without jeopardising test engineers or other traffic participants(Stellet et al., 2015) and enable to reach a high test coverage economically (Schuldt, 2017).

Nevertheless, test drives with the system under test (SUT) in the real world are currently indispensable due to the lack of other available certified methods for finally proving the functionality of the system (Winner et al., 2018). Simulation-based testing cannot be used for the functional approval process alone since they are currently not able to sufficiently represent the extraordinary complexity of the real world. Due to this, testing results "need to be verified and validated on test grounds and in field tests" (Winner et al., 2018), and real-world test drives are mandatory for finding relevant or critical scenarios which are the basis for scenario-based validation approaches (Menzel et al., 2018)(Bach et al., 2016).

1.1 Motivation

Still, assessing DAS functionality in the real world using Field Operations Tests (FOT) or Naturalistic Driving Studies (NDS) is complex but tedious and thus cost-intensive. That is at first due to the high amount of kilometres that is required to prove that the SUT works reliably (Kalra and Paddock, 2016).

Klitzke, L., Koch, C., Haja, A. and Köster, F.
Real-world Test Drive Vehicle Data Management System for Validation of Automated Driving Systems.
DOI: 10.5220/0007720501710180
In *Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2019)*, pages 171-180
ISBN: 978-989-758-374-2

Furthermore, engineers have to process that massive amount of data collected during such test campaigns in order to examine and verify the response of the system in specific scenarios for proving the system's functionality or fine-tuning the parameter of the SUT. Engineers have to know where to find specific or rather relevant scenarios in the data such as an overtaking sequence on a wet two-lane motorway driving towards sundown.

Discovering such scenarios may become an enormous economic burden and tedious task if analysing the data without computational assistance. That includes labelling the data with additional information for the identification of scenarios, performance assessment of a system or for providing a comprehensive data basis for deep learning. Despite that, the vast amount of data gathered during test campaigns need to be managed somewhere accessible by multiple project participants. Data Management Systems (DMS) have shown to be the right choice for such data management and analysing tasks due to their usage in various domains, e.g. medicine(Fraenkel et al., 2003), ecology (Frehner and Brändli, 2006) or finance (Shavit and Teichner, 1989).

The identification of scenarios in real-world test drives, so-called scenario mining (Elrofai et al., 2018), and analysing the influence of environmental effects on the system performance is still an open research question and the focus of the research project FASva[1]. Besides that, the project aims at conducting and analysing real-world test drives and propose tools and frameworks supporting scenario-based real-world test drive data analysis. Therefore, test drives of approx. 25,000 km were conducted on motorways, cities and rural roads in mainly northern Germany within the last two years. This data is the basis for addressing the following open research question within the research project:

1. How to automatically identify scenarios in real-world test drive data efficiently for setting up a rich catalogue of general driving scenarios and for enabling scenario search?
2. Which scenarios are relevant for the functional approval process of specific driving functions?
3. Which parameters are system-relevant in certain scenarios?
4. How to evaluate the performance of conducted real-world test drives to ensure conducting test campaigns efficiently?

[1]Intelligente Validierung von Fahrerassistanzsystemen (*engl.: intelligent validation of driver assistance systems*) of the Hochschule Emden/Leer.

1.2 Contribution of this Work

This work addresses the design of a system architecture for a Vehicle Data Management System (VDMS) that manages and analyses real-world test drive campaigns helping engineers finding scenarios of interest in the collected data. Hence, this paper discusses a partial solution to the first research question by proposing a system for efficient scenario identification and search. Therefore, the work discusses requirements for such a scenario mining system from different perspectives or roles utilising software-quality standard characteristics defined in ISO/IEC 25010.

Based on the derived requirements, the work proposes a highly-modularised multi-layer VDMS, that is used within FASva, to manage and support analysing large-scale test campaigns for the scenario-based validation of automated driving functions that is capable of aggregating the vehicle sensor data to scenes and enriching these scenes with information from various external sources. The latter is especially helpful if onboard sensors were missing but the analysis demands specific information in detail, e.g. about the weather or road. It also allows generating new aggregated signals like the time-to-collision with other traffic participants or integrating alternative algorithms as a reference for system evaluation. These functions or algorithms can be added to the VDMS to add further knowledge–even after the campaign. Hence, performing test drives with the missing sensors again is not required which cuts down expenses.

1.3 Paper Structure

This work is structured as follows: An overview of related work is given in Section 2. Afterwards, requirements on a VDMS are determined in Section 3. Therefore, relevant user-roles within such projects are defined and based on these, role-specific requirements are derived which are the base for the architecture proposed in Section 5. In Section 4 the processing chain for transforming the multivariate time-series of sensor data to a time-series of enriched scenes is discussed. The proof-of-concept is demonstrated in Section 6 with two use-cases: assessing the performance of an onboard driver assistance system and finding scenarios in real-world driving data. For the first, the onboard Lane Keep Assist System (LKA) of the research vehicle is selected and for the latter, lane change scenarios on highways will be identified using the VDMS.

2 RELATED WORK

Finding scenarios for the validation of automated driving functions obtained much attention since the last years. In particular, the focus is on the identification of relevant situations (Damm et al., 2018) in, e.g. databases of traffic accidents (Pütz et al., 2017), field operational tests (Benmimoun et al., 2011) or naturalistic driving studies (Klauer et al., 2006) aiming at setting up a database of relevant traffic scenarios for the validation of automated driving functions(Pütz et al., 2017). However, for the identification of such relevant scenarios, the data of the conducted test drives need to be managed and analysed.

Schneider et al. utilise a probabilistic approach using a Bayesian network and fuzzy features for the classification of emergency braking situations (Schneider et al., 2008). Weidl et al. optimise the Bayesian networks to recognise driving manoeuvres online (Weidl et al., 2014). In (Roesener et al., 2016) scenario-specific classification algorithms are evaluated for the identification of lane changes, vehicle followings and cut-ins.

All of the approaches have in common that they classify scenarios based on the vehicle sensor data. Thus, one can argue that they perform multivariate time-series analysis, as already pointed out by (Roesener et al., 2016). Taking the vast amount of data gathered during test-campaigns into account, reducing the data without high loss of information would, in turn, reduce the required storage capacity, the classification time and thus validation effort. Furthermore, utilising the definition of the term scenario from (Ulbrich et al., 2015) stating that a scenario describes a particular time interval with environment and traffic conditions, and including the description of the term scene representing a certain point in time, the vehicle sensor data has to be aggregated to a time-series of scenes for describing scenarios.

Therefore, we propose a temporal data discretisation approach to aggregate the raw vehicle sensor data to discrete scenes w.r.t to the time using equal width discretization (Liu et al., 2002) and by applying type-dependent data aggregation functions to reduce the data size (Moskovitch and Shahar, 2015) while at the same time establishing the foundation for scene-based scenario mining.

3 REQUIREMENTS ON VDMS

Due to the high economic effort of conducting real-world test drives, multiple parties usually plan and perform test campaigns. In this work, however, we focus on the roles, people have within such an DAS project, working with the VDMS. A description of these roles of interest is given in this section including role-based functional requirements on the architecture which are the basis for deriving general requirements utilizing software quality characteristics defined in the ISO 25010.

1. The *campaign manager* is responsible for the achievement of the project goals and acts as an interface to the principal or project owner. Consequently, they need up to date status information about the project's progress.

2. On behalf of the *campaign manager*, the *drive planner* plans the conduction of the specific test drives w.r.t to the general campaign goals and the current test drive coverage. They, therefore, require more detailed knowledge about the performed test drives including, for instance, the weather condition on specific trips or the road type distribution.

3. *Test drivers* perform the actual test drives according to the plans of the *drive-planner*. After each drive, they have to verify the fulfilment of drive-specific test requirements. The result may be a report, used by the driver planner to organise follow-up drives.

4. *Test engineers* perform the in-depth validation of the SUT. Based on the defined specification of the system, they verify the performance of the SUT. Hence, they need access to the sensor data collected by the test fleet in case of a system misbehaviour.

5. The last role of interest is the *Algorithm engineer*. People with this role either optimize design and develop new system functions or alternative solutions. The former allows adding additional knowledge to the database which may help in the SUT validation whereas the latter allows evaluating a SUT against a reference system, i.e. evaluate the performance of different traffic-sign-detection systems.

Concluding this overview, it is evident that different roles have various functional requirements on the VDMS w.r.t the grad of information detail or how to access the data or even extending the VDMS functionality. Based on the defined role-specific requirements, general characteristics of the architecture are now defined. Therefore, in order to ensure a software-quality driven design approach, a subset of the software quality characteristics defined in the ISO 25010, the successor of the ISO 9126, is employed.

Scaleability. Test drive campaigns typically have a specific duration spanning from several months up to years. Furthermore, the fleet of test drive campaigns typically consists of multiple vehicles and the test drivers may change during the campaign. Thus, the demand on a scalable framework exists in order to gap-free document the progress of the campaign including information about the drivers, e.g. sex, weight and height which might be used for driver behaviour or system acceptance analysis, or the configuration of each vehicle used within the campaign, such as the dimension of vehicles or sensor configurations.

Compatibility. Unfortunately, there is currently no standard tool to measure the vehicle sensor data. Conclusively, this also applies to the data format used for storing the vehicle sensor data. Thus, besides the file format of ADTF [2] used within this project, the VDMS should be able to support diverse data file formats. Furthermore, it should be able to manage information of external data sources, such as OpenStreetMap, which are required for the system analysis.

Maintainable. In order to support multiple data file formats or extend the functionality of the VDMS, adding new modules to the VDMS is vital. Thus, the architecture has to be highly modular – on different levels of the system. On the top level, where drive-related tasks run, adding further modules is required to process drives that were uploaded by test drivers, i.e. import the drive into the database, query the weather database based on the route in the drive or compress the files of the drive after the processing. Whereas on the level, where the processing of the vehicle sensor data takes places, adding new functions is required to add new facts to the database using, e.g. external data sources or developed algorithms, as stated in the previous description of the *Software engineer* role.

Reliability. In order to ensure that failures or non-normative behaviour of functions or algorithms added by engineers do not affect the whole system process, each module should run in a dedicated context. In case of an error of a module, the system should log this information and appropriately indicate that error.

[2]The Automotive Data and Time-triggered Framework (ADTF) of Elektrobit is used for synchronous data measurement and capturing.

4 VEHICLE DATA PROCESSING CHAIN

For the scene-based identification of scenarios in the real-world test-drive data, this work presents a three-stage process. The data basis are the raw vehicle sensor data that are transformed to series of scenes. Those scenes will be enriched with additional information from external data sources or algorithms as shown in Figure 1. In this section, each stage is described briefly.

4.1 Data Discretisation and Aggregation

The first step is the temporal data discretisation and aggregation utilising equal width discretisation(Liu et al., 2002) and applying type-depend aggregation operations.

Discretisation. At first, the time series of the available sensors are discretisised to a time series of scenes on a per-drive basis.

Therefore, let $D = \{t_b, S, t_e\}$ represent a drive as a set with three elements, whereas $t_b^d, t_e^d \in \mathbb{N}$ giving the beginning and end time of the drive d as milliseconds since epoch and $S = \{s_1, s_2, \ldots, s_k\}$ as the discretisesed and aggregated vehicle sensor data as a set of k scenes. Utilising the definition of multivariate time-series of (Baek and Kim, 2017), let $E = \{e_1, e_2, \ldots, e_m\}$ be a set of m vehicle sensors, each of which generates a finite series of n values $x_1, x_2, \ldots, x_n$. Since the vehicle sensors have different sample frequencies and thus the sensor series vary in length, the definition of (Baek and Kim, 2017) is adapted so that $\mathbf{x}_i = \left[x_1^i, x_2^i, \ldots, x_{n_i}^i\right]$ represents the time series of the ith sensor with n_i values. Furthermore the multivariate time series X is defined as a set of vehicle sensor time series with $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$.

For the discretisation of the multivariate time series X into a time series of scenes, equal width discretisation (EWD) is applied to each sensor series (Liu et al., 2002). Hence, using the previous definition of $\mathbf{x}_i$, the time series of the ith sensor is split up into a series of k scenes $\mathbf{u}_{i1}, \mathbf{u}_{i2}, \ldots, \mathbf{u}_{ik}$ with equal duration Δt so that the jth scene time series of the ith sensor is defined as

$$\mathbf{u}_{ij} = \{x \mid x \in \mathbf{x}_i \wedge \left(t_s^j \leq \Phi(x) \leq t_s^j + \Delta t\right)\} \quad (1)$$

whereas $\Phi(x)$ giving the time of the sample x in the time series and t_s^j stating the beginning of the jth scene s. Furthermore, the number of scenes $k = |S|$ in the drive d is given by

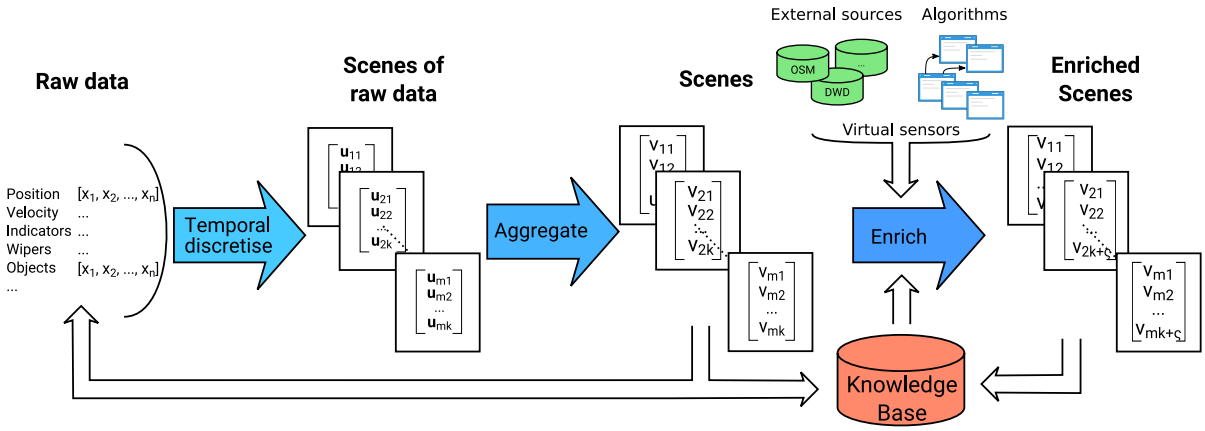

Figure 1: Three stage process for transforming the vehicle sensor data to a series of scenes enriched with additional information from algorithms and external data sources as the foundation for scene-based scenario identification.

$$k = \begin{cases} \frac{\widetilde{d}}{\Delta t} + 1 & \text{if } \widetilde{d} \bmod \Delta t > 0 \\ \frac{\widetilde{d}}{\Delta t} & \text{if } \widetilde{d} \bmod \Delta t \equiv 0 \end{cases} \tag{2}$$

with $\widetilde{d}$ stating the duration of the drive d. Then, utilizing the definition of (1) the series of scenes S of a drive d is formally defined by the $m \times k$ matrix

$$S_d = \begin{bmatrix} \mathbf{u}_{11} & \mathbf{u}_{12} & \dots & \mathbf{u}_{1k} \\ \mathbf{u}_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \mathbf{u}_{m1} & \dots & \dots & \mathbf{u}_{mk} \end{bmatrix}. \tag{3}$$

Aggregation. The next step is to aggregate the sensor series of each scene. Therefore, let $A = \{a_1, a_2, \dots, a_m\}$ represent a set of m aggregation functions for each sensor, mapping the scene time series $\mathbf{u}_{ij}$ of the ith sensor to an aggregated value v_{ij} with $a \in A : \mathbf{u}_{ij} \rightarrow v_{ij}$. Then, using the Equation (3), the series of scenes S of a drive d is formally represented by the $m \times k$ matrix

$$S_d = \begin{bmatrix} a_1(\mathbf{u}_{11}) & a_1(\mathbf{u}_{12}) & \dots & a_1(\mathbf{u}_{1k}) \\ a_2(\mathbf{u}_{21}) & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a_m(\mathbf{u}_{m1}) & \dots & \dots & a_m(\mathbf{u}_{mk}) \end{bmatrix} \tag{4}$$

where the scene $\mathbf{s}_t$ at time t is represented by the column vector $\mathbf{s}_t = [a_1(\mathbf{u}_{1t}), a_2(\mathbf{u}_{2t}), \dots, a_m(\mathbf{u}_{mt})]^T$ or in short $\mathbf{s}_t = [v_1, v_2, \dots, v_m]$.

For the data aggregation, a data-type dependent approach was chosen for selecting the aggregation functions. The supported data types are $T = \{\text{boolean}, \text{integer}, \text{floating point}, \text{string}\}$ and the set of default aggregation functions is $A = \{\text{or}, \text{median}, \text{mean}, \text{concatenate}\}$. The following applies mapping a scene time series $\mathbf{u}_{ij}$ to its aggregated value v_{ij} using the aggregation functions of A:

$$v_{ij} = \begin{cases} \text{mean}(\mathbf{u}_{ij}) & \text{if } \rho(\mathbf{u}_{ij}) \equiv \text{floating point} \\ \text{median}(\mathbf{u}_{ij}) & \text{if } \rho(\mathbf{u}_{ij}) \equiv \text{integer} \\ \text{or}(\mathbf{u}_{ij}) & \text{if } \rho(\mathbf{u}_{ij}) \equiv \text{boolean} \\ \text{concate}(\mathbf{u}_{ij}) & \text{if } \rho(\mathbf{u}_{ij}) \equiv \text{string} \end{cases} \tag{5}$$

with $x = \rho(\mathbf{u}_{ij})$ giving the value type of $\mathbf{u}_{ij}$, whereas $x \in T$. Besides the default aggregation functions, custom ones can be defined for specific signals.

4.2 Data Enrichment

The last step in the processing chain is the enrichment of the scenes with additional knowledge to further de-

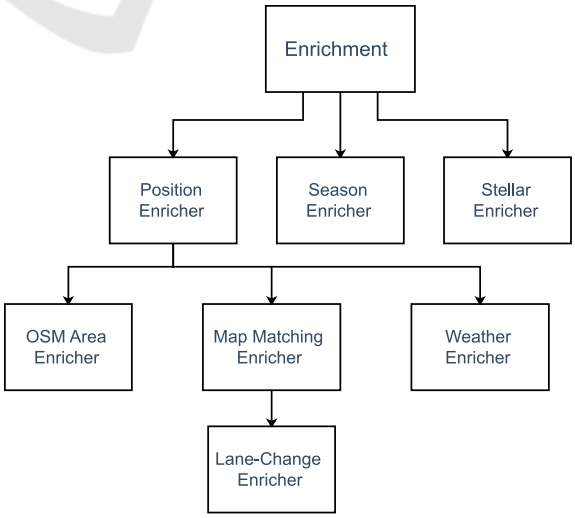

Figure 2: The enrichment module uses an acyclic graph to manage components and their dependencies for scene enrichment visualized as a tree of components.

scribe the vehicle and its environment such as other traffic participants, the weather or road as depicted at right of Figure 1. Hence, the previously introduced set E of vehicle sensors is extended with *virtual scene sensors*.

Each virtual sensor is a component running as a part of the enrichment module (cf. Section 5.2 about modules). Since a virtual sensor may depend on information generated by another sensor, i.e. a map matching algorithm used to map the ego vehicle position to a digital map depends on a accurate ego-position, an acyclic directed graph is used to manage the virtual sensors and the dependencies between them. An overview of the available components is depicted in Figure 2. Choosing an acyclic graph enables to build up complex processing chains with components only being run if their dependants finished processing a particular drive. It also allows running independent components concurrently to speed up the processing.

5 ARCHITECTURE

Due to the different roles defined in the Section 3 and their participation in the project and based on the defined general requirements, an event-driven three-tier architecture was chosen for the VDMS with the layers *Data*, *Modules* and *Interfaces* (see Figure 3).

5.1 Data Layer

The bottom layer *Data* describes every data represented as a file on the disc generated by either the modules or external sources including data in databases. These data files are, for instance, ADTF container collected during test drives by the *test drivers*. Furthermore, the layer includes files generated by components of the *Modules* layer, such as images which are extracted from the ADTF container, thumbnails of the images or JSON files containing the decoded CAN data of the ADTF containers.

The extraction of the CAN signal data from the ADTF container to JSON files is due to the demand to support other measurement tools such as ADTF as well. Because by only supporting a single data format, the usability of the framework is quite limited and thus the flexibility.

The basic properties of the JSON format provide information about the *vehicle*, *driver* and the *time interval* of the measurements. The latter is required because all timestamps of signal values are relative to the defined start time which enables to change the reference system easily, i.e. to synchronize the record times to an absolute reference system even after the test drive. The special property *measurements* contains the signal values. Each signal is defined by its minimum and maximum value, its type (e.g. floating point number or integer) and unit (e.g. kilometre per hour). Besides that, it contains all timestamped values of that signal in the property *value* sorted by the signal value timestamps.

5.2 Modules Layer

The second layer *Modules* entails all modules of the system. Each module has a distinct functional purpose and is highly independent of the other modules ensuring a loosely-coupled design and thus facilitating a scalable and maintainable software system.

The loose coupling of modules is realized by an event-driven file-system based information passing method and by utilising the observer pattern for asynchronously notify about module state changes. The latter is used as the base for the *Module State Control* component of the layer *Interfaces* to asynchronously notify connected clients, e.g. *test engineers*, via WebSocket connections about the progress of modules. Whereas the former is used to trigger modules and thus, start the processing of a certain drive. Therefore, all modules have the following three parameters, *source*, *indicate*, *destination*.

The *source* parameter defines the directory where the module watches for drives to process. If a module successfully processed a drive, it creates a new symbolic link in one or multiple directories, defined with the parameter *indicate*, to inform all other modules watching for the directory defined in *indicate* about its progress. In the case of modules creating new files, the parameter *destination* defines the location where to put those files.

Having the compatibility in mind, the implemen-

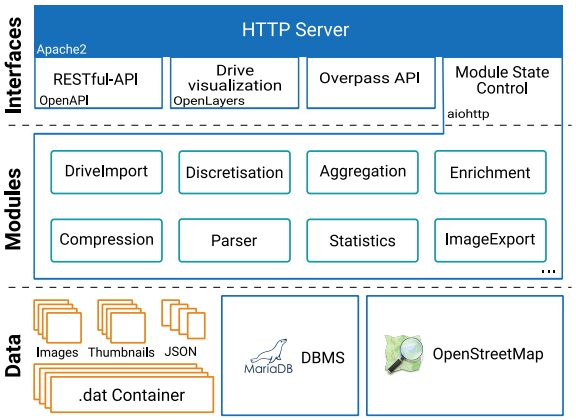

Figure 3: The architecture of the proposed VDMS for managing large-scale test campaigns consists of three layers: *Data*, *Modules* and *Interfaces*.

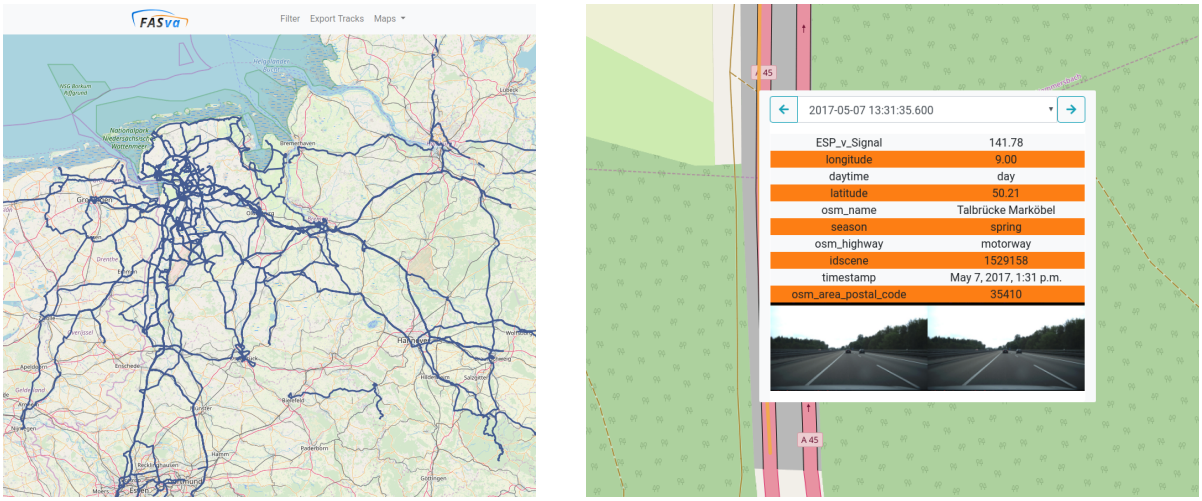

Figure 4: The *drive visualisation* service provides an interactive web-interface of the conducted test drives including filter capabilities for efficient sequence identification and track selection for scene analysis. *Left:* A map of all conducted test drives. *Right:* The same zoomed-in map showing information of a selected scene.

tation language of the module functionality is not restricted to the implementation language of the architecture which currently is Python. Instead, the module class merely works as a wrapper or adapter to the actual functionality for saving the state of the module's progress in the database and to guarantee the reliability of the framework, i.e. robustify the framework against misbehaviour or errors in the modules such as memory leaks.

5.3 Interfaces Layer

The last layer *Interfaces* provides access to the VDMS for different project roles with each component of the layer working as a service interconnected by an HTTP server.

The *Drive visualisation* service utilises OpenStreetMap to show the conducted test drives via an interactive web-based frontend. The left image in Figure 4 shows the conducted drives within FASva. The purpose of this service is to help *drive planner* by planning test drives since it gives a rough overview about the test drive coverage w.r.t the geolocation and *engineers* by finding sequences of interest. Therefore, the web-interface provides filter and selection capabilities. The former enables to search for sequences with specific characteristics, e.g. test campaign, daytime, region of interest or road type and the latter gives access to specific situations or scenes of a drive. That includes information about the vehicle and its environment either from onboard sensors, e.g. velocity or location or any other external sources such as weather, street type or daytime.

The RESTful-API service provides access to the data of conducted test drives, e.g. sensor data or images of cameras and is used by *test engineers* and *algorithm developers*. The OpenAPI specification is used for the description of the API, allowing to generate client applications for various programming languages. The *drive visualisation* service, for instance, uses the *RESTful-API* to retrieve the meta-information and thumbnails of specific situations and the geolocation of the conducted test drives.

The *Module State Control* service allows *algorithm engineers* to interact with the modules of the Modules layer, e.g. to start the processing of a particular drive or getting notified if a module finished processing.

The Overpass API service grant access to the OpenStreetMap (OSM) server for adding information about the infrastructure. This enables *test engineers* or *algorithm developers* to find sequences that took place on specific road types, e.g. motorway or rural roads. The `Map Matching Enricher` of the *Enrichment* module, for instance, uses the OSM server to retrieve information about the road the vehicle is on, which is used by the `Lane Change Enricher` to detect lane changes only on highways.

6 EVALUATION

For the evaluation of the-proof-of concept it is shown that the proposed processing chain can be used for finding scenarios of interest and for evaluating driving functions.

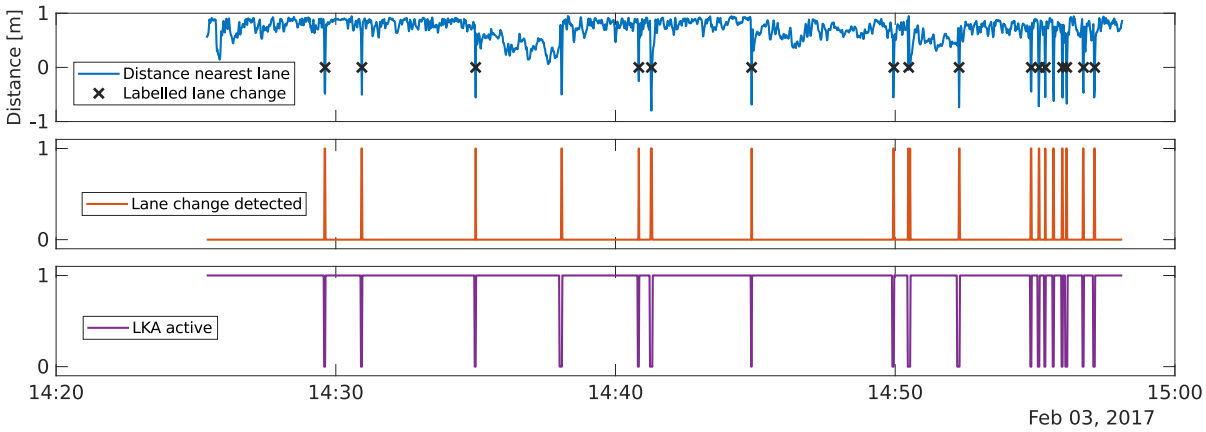

Figure 5: Sequence of a trip on a motorway with two to three lanes for demonstrating the proof-of-concept of the proposed VDMS for scenario identification and system evaluation. *Top:* Distance to nearest line and the manually labelled lane changes. *Middle:* The detected lane changes of the algorithm. *Bottom:* The signal of the onboard Lane Keep Assist System.

6.1 Dataset

The data basis is from a short sequence of approx. 100 km conducted with our research vehicle (see Figure 5). The sequence takes place on a motorway with three lanes. During the trip, 16 lane changes occurred which were manually labelled using the *drive visualisation* interface shown in Figure 4.

The sample rate of the vehicle sensors vary between 25 and 100 Hz. However, during the conversion of the ADTF file format to the described JSON format in Section 5.1, the sample frequency was limited to 25 Hz with equal time interval sampling.

For the aggregation of the multivariate time series to a time-series of scenes as described in Section 4.1 and thus further reduction of the sample frequency, a scene duration $\Delta t = 1$ second was chosen. This is due to the fact that, according to an analysis of real-world drives by (Olsen et al., 2002), the mean duration of lane changes on motorways is $\mu = 6.25$ seconds with a standard deviation of $\sigma = 1.64$. Thus, assuming a Gaussian distribution with the parameter (μ, σ) and ensuring the detection of a lane change using one second scene intervals, the probability of a lane change duration X being greater two seconds $P(X > 2)$ is approx. 99.52% which is adequate for the analysis.

6.2 Results

For the demonstration on how to find scenarios of interest–in this case lane change scenarios–a naive lane change detection algorithm was implemented and added to the VDMS which uses the distance to the nearest line signal for finding lane change events. The evaluation is performed in MATLAB by retrieving the data of the sequence via the RESTful API.

In Figure 5, the orange line indicates the result of the lane change detection algorithm whereas the black crosses mark the manually labelled lane changes. Since classification of lane changes is a binary classification problem, the F_1-score is employed for assessing the performance. The algorithm detects all lane changes correctly but also has four false positives leading to a F_1-score of 88.89%. This demonstrates that even with a significant system-dependent reduction of the sample size, a robust identification of scenarios is possible.

Besides the identification of scenarios, the assessment of a SUT is another typical use-case. A *test engineer* might want to find those situations in which a SUT such as a LKA does not operate. In Figure 5 the purple line represents the state of the LKA. If the LKA actively assist the driver, the signal is one and zero otherwise. Hence, the situations in which the signal is zero are of special interest. From the Figure 5 it is evident that on this sequence, the LKA stops operating if the driver performs a lane change. Hence, this system does not actively assist the driver during lane changes. This demonstration shows that by supporting the addition of algorithms to the VDMS, reference signals with a higher confidence or further knowledge about the vehicle and its environment may help assessing onboard DAS.

7 CONCLUSION

Test drives in the real world are a valuable data source for finding relevant and critical scenarios which are required for the validation of automated driving functions (Damm et al., 2018). Since the analysis of large-scale test campaigns requires computational as-

sistance to identify scenarios efficiently, this work proposes an highly modularized three-tier architecture of a VDMS for the management and analysis of real-world test drives for the scenario-based validation of automated driving functions.

Based on a formal definition of time-series of scenes, a processing chain for transforming the raw vehicle sensor data to a time series of scenes for scenario mining is presented. That processing chain is a central component of the proposed VDMS whereas the design of the architecture follows a requirements-driven approach by analysing the needs of particular project roles and deriving specific and general requirements on the architecture.

The proof-of-concept is finally evaluated by using the RESTful API for identifying lane change scenarios based on real-world data. That demonstration shows that even with a significant reduction of the sample size, robust identification of scenarios is still possible. Conclusively, the demonstrations show the feasibility of the VDMS for identifying scenarios in real-world test drives efficiently. However, in follow-up work, an in-depth analysis of choosing the scene duration Δt on the performance of different scenario-mining algorithm has to be conducted.

Focussing on the compilation of a sophisticated set of scenarios for validating automated driving functions (Damm et al., 2018), several topics need to be addressed in follow-up work. At first, since the set of relevant scenarios depends on the road type of the vehicle, the current map matching algorithm needs to be replaced with a more robust one. For the identification of other scenarios such as overtaking or approaching, information about the road and other traffic participants are required. Thus, in future work, image-based lead vehicle and road detection will be integrated into the VDMS.

Besides the enrichment of scene understanding, follow-up work will address the open research questions defined in Section 1. At first, future work will focus on the identification of scenarios in real-world test drives using the proposed VDMS.

ACKNOWLEDGEMENTS

We thank LG Electronics, Vehicle Solution Company, Republic of Korea, for supporting this project by cooperating in capturing large-scale test drives and providing valuable measurement equipment.

REFERENCES

Bach, J., Otten, S., and Sax, E. (2016). Model based scenario specification for development and test of automated driving functions. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 1149–1155. IEEE.

Baek, S. and Kim, D. Y. (2017). Empirical sensitivity analysis of discretization parameters for fault pattern extraction from multivariate time series data. *IEEE Transactions on Cybernetics*, 47(5):1198–1209.

Benmimoun, M., Fahrenkrog, F., Zlocki, A., and Eckstein, L. (2011). Incident detection based on vehicle can-data within the large scale field operational test "eurofot". In *22nd Enhanced Safety of Vehicles Conference (ESV 2011), Washington, DC/USA*.

Damm, W., Möhlmann, E., Peikenkamp, T., and Rakow, A. (2018). *A Formal Semantics for Traffic Sequence Charts*, pages 182–205. Springer International Publishing, Cham.

Elrofai, H., Paardekooper, J., Gelder, E. d., Kalisvaart, S., and Op den Camp, O. (2018). Streetwise: scenario-based safety validation of connected automated driving. Technical report, TNO.

Fraenkel, D. J., Cowie, M., and Daley, P. (2003). Quality benefits of an intensive care clinical information system. *Critical Care Medicine*, 31(1):120–125.

Frehner, M. and Brändli, M. (2006). Virtual database: Spatial analysis in a web-based data management system for distributed ecological data. *Environmental Modelling & Software*, 21(11):1544–1554.

Haja, A., Koch, C., and Klitzke, L. (2017). The ADAS SWOT Analysis - A Strategy for Reducing Costs and Increasing Quality in ADAS Testing. In *Proceedings of the 3rd International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2017)*, pages 320–325.

Kalra, N. and Paddock, S. M. (2016). Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice*, 94:182–193.

Klauer, S. G., Dingus, T. A., Neale, V. L., Sudweeks, J. D., Ramsey, D. J., et al. (2006). The impact of driver inattention on near-crash/crash risk: An analysis using the 100-car naturalistic driving study data. Technical report, National Highway Traffic Safety Administration.

Liu, H., Hussain, F., Tan, C. L., and Dash, M. (2002). Discretization: An enabling technique. *Data mining and knowledge discovery*, 6(4):393–423.

Menzel, T., Bagschik, G., and a. M. Maurer (2018). Scenarios for development, test and validation of automated vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1821–1827.

Moskovitch, R. and Shahar, Y. (2015). Classification-driven temporal discretization of multivariate time series. *Data Mining and Knowledge Discovery*, 29(4):871–913.

Olsen, E. C. B., Lee, S. E., Wierwille, W. W., and Goodman, M. J. (2002). Analysis of distribution, frequency, and duration of naturalistic lane changes. *Proceedings of*

the Human Factors and Ergonomics Society Annual Meeting, 46(22):1789–1793.

Pütz, A., Zlocki, A., Bock, J., and Eckstein, L. (2017). System validation of highly automated vehicles with a database of relevant traffic scenarios. In *12th ITS European Congress*. ITS European Congress.

Roesener, C., Fahrenkrog, F., Uhlig, A., and Eckstein, L. (2016). A scenario-based assessment approach for automated driving by using time series classification of human-driving behaviour. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1360–1365.

Schneider, J., Wilde, A., and Naab, K. (2008). Probabilistic approach for modeling and identifying driving situations. In *2008 IEEE Intelligent Vehicles Symposium*, pages 343–348.

Schuldt, F. (2017). *Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen*. PhD thesis, Technische Universität Carolo-Wilhelmina zu Braunschweig.

Shavit, E. and Teichner, L. (1989). Interactive market management system. US Patent 4,799,156.

Stellet, J. E., Zofka, M. R., Schumacher, J., Schamm, T., Niewels, F., and Zöllner, J. M. (2015). Testing of advanced driver assistance towards automated driving: A survey and taxonomy on existing approaches and open questions. In *18th IEEE International Conference on Intelligent Transportation Systems*, pages 1455–1462.

Ulbrich, S., Menzel, T., Reschka, A., Schuldt, F., and Maurer, M. (2015). Defining and substantiating the terms scene, situation, and scenario for automated driving. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 982–988.

Weidl, G., Madsen, A. L., Kasper, D., and Breuel, G. (2014). Optimizing bayesian networks for recognition of driving maneuvers to meet the automotive requirements. In *2014 IEEE International Symposium on Intelligent Control (ISIC)*, pages 1626–1631.

Winner, H., Lemmer, K., Form, T., and Mazzega, J. (2018). *PEGASUS—First Steps for the Safe Introduction of Automated Driving*, chapter Vehicle Systems and Technologies Development, pages 185–195. Springer International Publishing, Switzerland.