




Vehicle Data Management System for Scenario-Based Validation of Automated Driving Functions

Lars Klitzke¹ , Carsten Koch¹, Andreas Haja¹, and Frank Köster²

¹ Department of Electronics and Informatics, Hochschule Emden/Leer, University of Applied Sciences, Emden, Germany

{lars.klitzke, carsten.koch, andreas.haja}@hs-emden-leer.de

² German Aerospace Center (DLR), Institute of Transportation Systems, Braunschweig, Germany

frank.koester@dlr.de

Abstract. Proving the functionality of AI-controlled automated vehicles is a challenging task due to the enormous overall complexity. Although a scenario-based validation approach is widely accepted in the literature, the identification of these scenarios is still an open issue.

Real-world test drives are valuable data sources for this purpose. However, an automated system is required for data management and scenario identification to analyze the vast amount of data in a legitimate amount of time and effort. Therefore, this work proposes a modular multi-tier Vehicle Data Management System for large-scale test campaign management and analysis as the basis for scenario-based validation of automated driving functions. For system demonstration, lane-change maneuvers are identified and extracted, and an onboard DAS is evaluated with a real-world test drive sequence.

Keywords: ADAS validation · Maneuver identification · Real-world test drives · Data enrichment · Scenario mining · Data Management System

1 Introduction

Automakers and suppliers are currently competing fiercely to develop advanced driver assistance systems (ADAS) and to be among the first to launch a fully automated driving solution into the market. In addition to the difficulties of solving technical challenges, the high amount of testing and validation required for such systems poses a serious cost factor for existing players and a high entry barrier for new companies which are planning on entering the ADAS segment. Hence, recent and current research projects aim to provide methodologies, methods and tools [8, 25] 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 validating driver assistance systems. With more sophisticated models of

real-world components becoming available, testing shifted from the real to the virtual world. This is due to the fact that simulations allow to conduct risky maneuvers without risking vulnerables such test engineers or other traffic participants [22]. Moreover, they enable to reach a high test coverage more economically [19].

Nevertheless, real-world test drives are still compulsory to finally prove the system functionality since no other certified methods are available [25]. Although simulation-based testing procedures reduce the overall validation effort, they cannot be used for the final system approval. This is due to the fact that they are currently not able to sufficiently represent the extraordinary complexity of the real world. Hence, testing results “*need to be verified and validated on test grounds and in field tests*” [25]. But, since rigorous testing in the real world is economically infeasible, a scenario-driven validation approach aims at reducing the overall effort [1, 13]. For that purpose, real-world test drives are mandatory to find relevant or critical scenarios which are the basis for scenario-based validation approaches [4].

However, due to the high mileage that is required to prove the reliability of the system under test (SUT) [9], assessing ADAS functionality in the real world using Field Operations Tests (FOT) or Naturalistic Driving Studies (NDS) is complex, tedious and cost-intensive. Engineers have to process and manage huge amount of recorded data collected during test campaigns to prove the system’s functionality or fine-tune the parameter of the SUT by examining and verifying the response of the system in specific scenarios. Furthermore, 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 analyzing the data without computational assistance. That includes labeling the data with additional information for the identification of scenarios, performance assessment of a system or for providing a comprehensive data basis for machine learning [17]. Apart from that, the vast amount of data gathered during test campaigns must be managed in such a way that it is accessible by multiple project participants. Data Management Systems (DMS) have shown to be the right choice for such data management and analyzing tasks due to their usage in various domains, e.g. medicine [6], finance [20] or ecology [7].

1.1 Research Project FASva

The identification of scenarios in real-world test drives, so-called scenario mining [5], and analyzing the influence of environmental effects on the system performance is still an open research question and the focus of the research project FASva¹. Besides that, the project aims at conducting and analyzing 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

¹ Intelligent Validierung von Fahrerassistenzsystemen (*engl.: intelligent validation of driver assistance systems*) of the Hochschule Emden/Leer.

the following open research question within the research project as already pointed out in an earlier work [11]:

1. How to automatically identify scenarios in real-world test drive data efficiently for setting up a rich catalog 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?

1.2 Contribution

To help engineers finding scenarios of interest in real-world test-drive data, this work proposes a Vehicle Data Management System (VDMS) that is capable of scenario identification. Hence, this paper addresses the first research question by introducing a system for efficient scenario identification and search. For that purpose, the VDMS presented in an earlier work [11] is extended by a procedure to extract maneuvers as sequences instead of only finding the most likely point in time where the maneuvers occurs, which is the main focus of other works [21,24,26]. The knowledge of the time interval of maneuvers enables to analyze conducted test drives quantitatively in terms of, e.g., total mileage and duration. Apart from that, engineers are able to find sequences in real-world driving data representing a particular maneuver of interest to examine Driver Assistance System (DAS) more efficiently. The feasibility for maneuver extraction is demonstrated and the performance of the approach is evaluated with a motorway sequence of manually labeled lane-change intervals.

1.3 Structure of This Work

This work is structured as follows: Related work and projects are discussed in Sect. 2. Afterwards, requirements on a VDMS are determined in Sect. 3 based on software quality characteristics defined in ISO 25010 and requirements are derived for different users-roles participating in test campaign. For the sake of completeness, the terms scene, maneuver and scenarios are defined in Sect. 4. These definitions are the basis for the proposed processing chain for maneuver identification and extraction presented in Sect. 5. Afterwards, the architecture of the VDMS is described in Sect. 6. To demonstrate the usability of the VDMS and to assess the performance of the proposed maneuver extraction algorithm, a Lane Keep Assist System (LKA) is analyzed with a test-drive on a motorway, and the accuracy of lane-change maneuver extraction is assessed in Sect. 7. The paper concludes with a summary and outlook for further work in Sect. 8.

2 Related Work

Due to the development of AI-driven vehicles, the automotive industry faces new challenges by verifying the functional safety of the systems. Since rigorous testing of the

automated driving systems is economically infeasible, a scenario-driven validation approach aims at reducing the overall complexity. Due to this, identifying scenarios for the validation of automated driving functions obtained much attention in the last years. In particular, the focus is on the identification of relevant situations [4] in, e.g. databases of traffic accidents [16], field operational tests [3] or naturalistic driving studies [10] aiming at setting up a database of relevant traffic scenarios for the validation of automated driving functions [16, 27]. However, for the identification of such relevant scenarios, the data of the conducted test drives need to be managed and analyzed.

Schneider et al. utilize a probabilistic approach using a Bayesian network and fuzzy features for the classification of emergency braking situations [18]. Weidl et al. optimize the Bayesian networks to recognize driving maneuver online [24]. In [17] scenario-specific classification algorithms are evaluated for the identification of lane changes, vehicle followings and cut-ins. Sonka et al. [21] propose an approach for lane change and lane keeping detection by combining a probabilistic approach with fuzzy logic.

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 [17]. 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, utilizing the definition of the term scenario from [23] 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, this work proposes a temporal data discretization approach to aggregate the raw vehicle sensor data to discrete scenes w.r.t to the time using equal width discretization [12] and by applying type-dependent data aggregation functions to reduce the data size [14] while at the same time establishing the foundation for scene-based scenario mining.

To examine the behavior of automated driving functions in specific situations, information about the occurrence of a maneuver in a certain point in time is helpful. Those data annotations allow test engineers to analyze test drives more efficiently since they know where to find probably relevant situations. Hence, they only need to extract sequences of the drives around those relevant situations.

However, instead of only annotating the point in time with high likelihood for representing a maneuver, the time interval representing that particular maneuver would further help on assessing certain driving functions in sequences of interest, e.g., the adaptive cruise control (ACC) system in vehicle following scenarios. Moreover, by partitioning test drives in maneuvers, they can be quantitatively described according to the occurrence of certain maneuvers. For instance, a free driving sequence on a motorway (no vehicle in front of the ego vehicle) may not be relevant for the validation of an ACC system at all. But, if the ACC emits a braking signal in free driving sequences that may require further investigation by the test engineer. Hence, such data annotations enable test engineers to quickly identify sequences of interest.

In the last decades, however, the main focus of other works is on recognizing the presence of a certain maneuver in a specific point in time with high accuracy [21,24], e.g., to predict maneuvers of other vehicles to react accordingly [26]. For that purpose, [21] proposed a method to identify the most probable point in time of a line-crossover by estimating the likelihood for a line crossover for each point in time. The latter serves as the basis for the proposed maneuver extraction approach that is presented in Sect. 5.3.

3 Requirements

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 working with the VDMS in such projects. A description of these roles 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 fulfillment of drive-specific test requirements. The result may be a report, used by the driver planner to organize 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 misbehavior.
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 behavior 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 ADTF² 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 may be 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 behavior 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.

4 Scenario Definition

Before presenting the proposed VDMS for scenario mining, the terms *scene*, *maneuver* and *scenario* used in this work will be briefly described in the following based on a representative sequence. The different terms are illustrated utilizing a hierarchy of timelines depicted in Fig. 1.

The *drive* represents the first timeline. In this work, a drive begins at that moment where the software in the experimental vehicle start recording and ends if the driver stops the recording. A drive may have time gaps caused by the driver pausing and

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

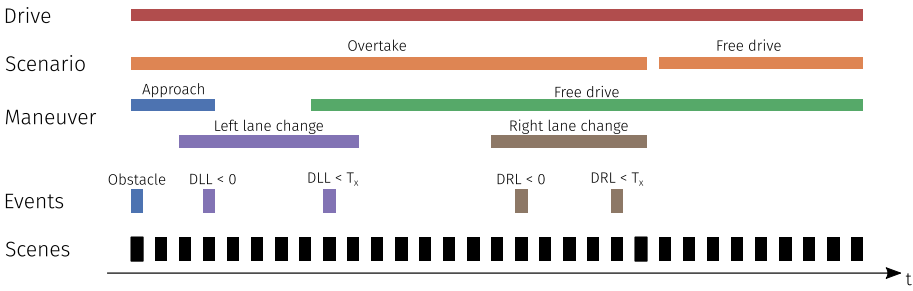


Fig. 1. This figure shows a representative sequence to illustrate the definition of a scenario and scene within this work. The sequence contains an overtaking scenario that consists of multiple subsequence maneuvers, each of which is represented by scenes.

resuming the recording. Each drive consists of multiple consecutive non-overlapping scenarios on the second timeline.

Although different definitions of the term *scenario* exist in the literature, this work employs the one defined by Ulbrich et al. stating that a “scenario describes the temporal development between several scenes in a sequence of scenes. Every scenario starts with an initial scene. Actions & events as well as goals & values may be specified to characterize this temporal development in a scenario. Other than a scene, a scenario spans a certain amount of time.” [23]

Such a scenario may be the *overtake* scenario depicted in Fig. 1, where the driver intends to overtake another vehicle. However, this definition of a scenario allows no distinction between a maneuver and a scenario. Maneuvers also span a certain amount of time, and the driver has a specific intention, too. For instance, the driver aims at closing up to a lead vehicle in an approaching maneuver [1]. So, one can argue that maneuvers and scenarios are the same. Due to this, *maneuvers* are, in this work, certain driving actions represented by series of scenes initiated by events such as an obstacle in front of the ego vehicle or the scene where the vehicle crosses a line. Thus, they represent a time interval in which the vehicle performs a particular driving action. Furthermore, the definition of a scenario is extended by the restriction that a scenario consists of at least one maneuver and that it also includes information about the vehicle’s environment such as other traffic participants or road infrastructure as in [1].

The smallest time unit is a *scene* representing the state of the vehicle and its environment in a short interval, e.g. seconds, depicted on the last timeline as black bars. Hence, in this work, the restriction of [23] that a scene does not span “a certain amount of time” [23] is relaxed to guarantee that all information are available—even if sensors with different sampling frequencies are used.

5 Processing Chain

Based on the definition of a scene, maneuver and scenario in the previous section, the proposed procedure to process real-world test drive data is presented in the following.

This work extends the three-stage process introduced in [11] for the scene-based identification of scenarios in the real-world test-drive data with a maneuver mining step (see Fig. 2). The data basis for the processing are the raw vehicle sensor data collected during test drives that are transformed to series of scenes. Those scenes will be enriched with additional information from algorithms and external data sources. Based on the enriched scenes, maneuvers are extracted. For the sake of completeness, each stage is described briefly in the following.

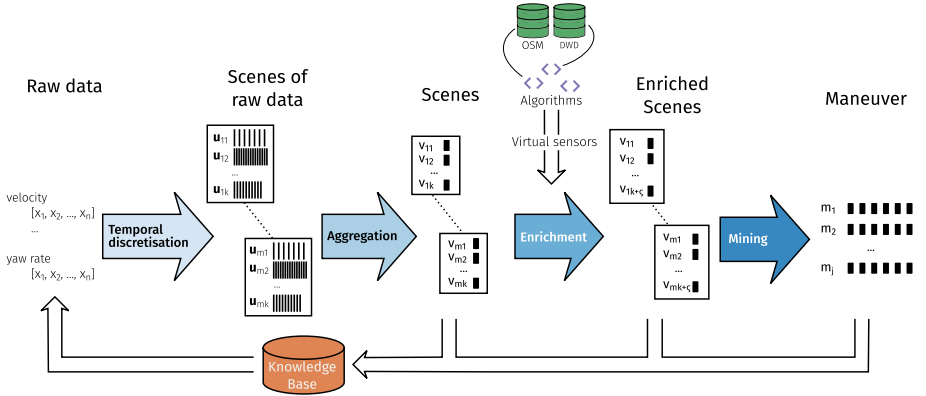


Fig. 2. The four stage process for maneuver identification by transforming the vehicle sensor data to a series of scenes enriched with additional information from algorithms and external data sources.

5.1 Data Discretization and Aggregation

The first step is the temporal data discretization and aggregation utilizing equal width discretization [12] and applying type-depend operations.

At first, the time series of the available sensors are discretized to a time series of scenes on a per-drive basis. Therefore, let $D = \{t_{start}, S, t_{end}\}$ represent a drive as a set with three elements, whereas $t_{start}^d, t_{end}^d \in \mathbb{N}$ giving the beginning and end time of the drive d as milliseconds since epoch and $S = \{s_1, s_2, \dots, s_k\}$ as the discretized and aggregated vehicle sensor data as a set of k scenes. Utilizing the definition of multivariate time-series of [2], let $E = \{e_1, e_2, \dots, e_m\}$ be a set of m vehicle sensors, each of which generates a finite series of n values x_1, x_2, \dots, x_n . Since the vehicle sensors have different sample frequencies and thus the sensor series vary in length denoted in Fig. 2 with the number of black bars per signal in the scene of raw sensor values, the definition of [2] is adapted so that $\mathbf{x}_i = [x_1^i, x_2^i, \dots, x_{n_i}^i]$ represents the time series of the i th sensor with n_i values. Furthermore, the multivariate time series X is defined as a set of vehicle sensor time series with $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$.

For the discretization of the multivariate time series X into a time series of scenes, equal width discretization (EWD) is applied to each sensor series [12]. Hence, using the previous definition of \mathbf{x}_i , the time series of the i th sensor is split up into a series of

k scenes $\mathbf{u}_{i1}, \mathbf{u}_{i2}, \dots, \mathbf{u}_{ik}$ with equal duration Δt so that the j th scene time series of the i th sensor is defined as

$$\mathbf{u}_{ij} = \{x \mid x \in \mathbf{x}_i \wedge (t_{start}^j \leq \Phi(x) \leq t_{start}^j + \Delta t)\} \tag{1}$$

whereas $\Phi(x)$ giving the time of the sample x in the time series and t_{start}^j stating the beginning of the j th scene s . The number of scenes $k = |S|$ in the drive d is given by

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

with \tilde{d} stating the duration of the drive d . Then, utilizing the definition of Eq. (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 i th sensor to an aggregated value v_{ij} with $a \in A : \mathbf{u}_{ij} \rightarrow v_{ij}$. Then, using the Eq. (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 $T = \{\text{boolean, integer, floating point, string}\}$ and the set of default aggregation functions is $A = \{\text{or, median, mean, concatenate}\}$. The following applies mapping a scene time series \mathbf{u}_{ij} to its aggregated value v_{ij} using the aggregation functions of A :

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

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.

5.2 Data Enrichment

The next step in the processing chain is the enrichment of the scenes with information from various data sources to further describe the vehicle and its environment such as other traffic participants, the weather or road as depicted at the right of Fig. 2. Hence, the previously introduced set E of vehicle sensors is extended with *virtual scene sensors*.

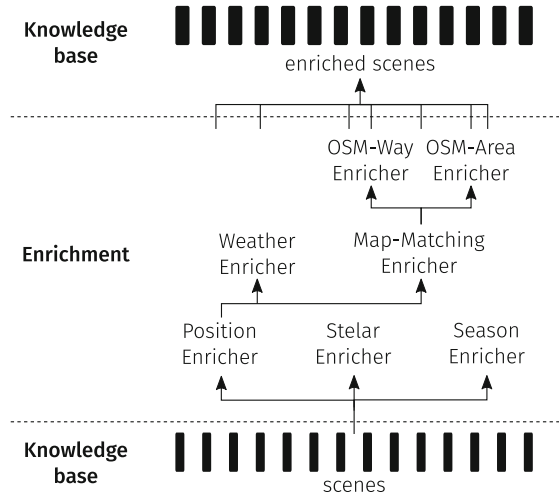


Fig. 3. The enrichment module uses an acyclic graph to manage components and their dependencies for scene enrichment visualized as a tree of components.

Each virtual sensor is a component running as a part of the enrichment module (cf. Sect. 6.2 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 an 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 Fig. 3. Choosing an acyclic graph enables to build up processing chains with components only being run if their dependents finished processing a particular drive. It also allows running independent components concurrently to speed up the processing.

5.3 Maneuver Mining

The last step of the chain is the extraction of maneuvers based on the enriched scenes and depicted in Fig. 4.

Instead of only giving the most likely point in time of the maneuver, the most likely interval is retrieved. Thus, let $M = \{m_1, m_2, \dots, m_j\}$ be the set of maneuvers in the drive with $B = \{b_1, b_2, \dots, b_n\}$ as the available maneuver types, and $G = \{(m, b)_1, (m, b)_2\}$ a set of tuples mapping each maneuver $m \in M$ to at least

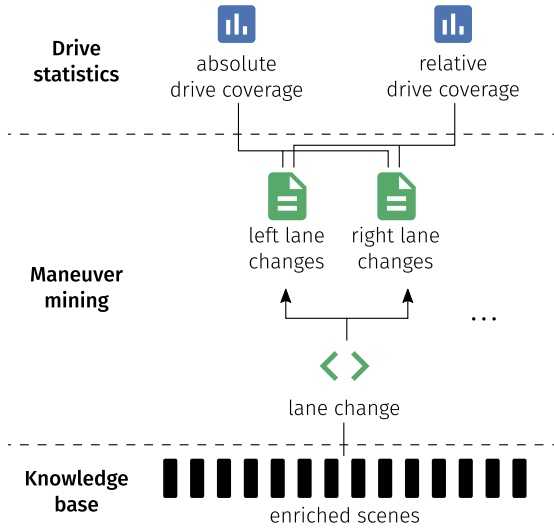


Fig. 4. The procedure for maneuver mining and drive coverage estimation based on the enriched scenes.

one maneuver type $b \in B$. A maneuver m is defined as a series of enriched scenes, $m = s_{t_{start}}, s_{t_{start}+1}, \dots, s_{t_{end}}$ with t_{start} and t_{end} as the start and end of the maneuver. The duration of the maneuver \tilde{m} is estimated by $\tilde{m} = \Delta t * |m|$ with Δt as the chosen scene width in the data discretization step (cf. Eq. (1)). It is assumed that different types of maneuvers (m_{b_i}, m_{b_j}) overlaps, so that $m_{b_i} \cap m_{b_j}$ is not empty in every case. But, maneuvers $(m_{b_i}^i, m_{b_i}^j)$ of the same type b are distinct, i.e. $m_{b_i}^i \cap m_{b_i}^j = \emptyset$ which is also represented in Fig. 1. As depicted in Fig. 4 the framework is currently able to identify and extract lane change maneuvers.

For extracting the maneuver interval, a probability-based approach is used based on the *Fine Search* procedure proposed by [21]. The maneuver extraction process is elaborated in the following for the lane change maneuver depicted in Fig. 5.

Let $f(t)$ represent the distance to the right lane of the scene at time t in a moving time window $w = \{t - \Delta t, \dots, t + \Delta t\}$ with a duration of $2\Delta t$ so that $t \in w$ (see blue line in Fig. 5). The first step is to reduce noise in the signal since this impacts the follow-up steps. Thus, the moving window is smoothed by convolving it with a uniform kernel of size n . Furthermore, let $P(t)$ represent the probability and $P'(t)$ the slope of the probability for a maneuver at time t . For the lane change maneuver, the probability $P(t)$ is defined as

$$P(t) = \frac{|g(t)|}{|g(t)|_{max}} \tag{6}$$

with $g(t)$ as $f(t)$ shifted so that

$$g(t) = f(t) - \tilde{f}(t) \tag{7}$$

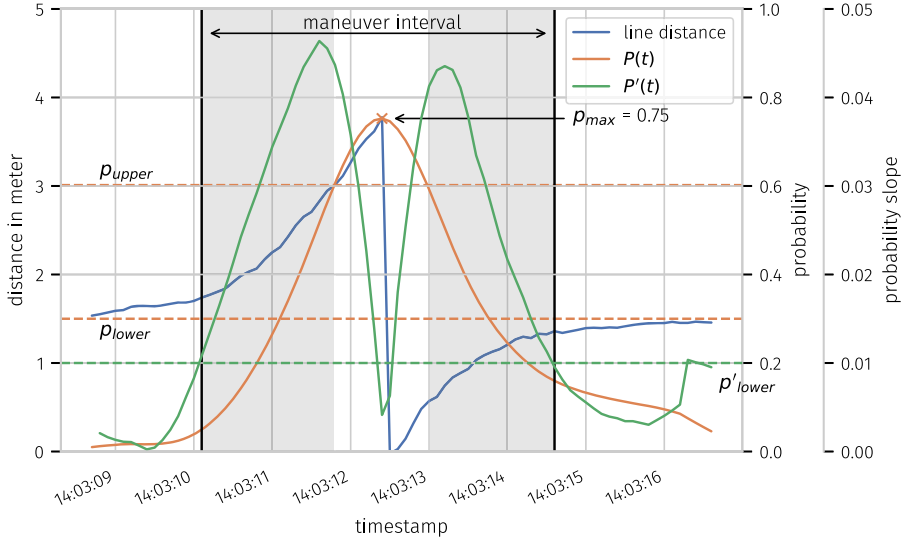


Fig. 5. For maneuver interval extraction, a probability-based approach is utilized and exemplary shown for the lane-change maneuver.

where $\tilde{f}(t)$ is the signal mean in the moving window such that

$$\tilde{f}(t) = \frac{1}{|w|} \sum_i^w f(i) \quad (8)$$

finally leading to

$$|g(t)| = \left| f(t) - \tilde{f}(t) \right|. \quad (9)$$

To find the start and end time of the maneuver t_{start}, t_{end} , the time and value t_{max}, p_{max} of the signal peak is estimated in the moving window. Afterwards, the signal window is split up into the left and right part. The start of the left $t_{l,start}$ and right $t_{r,start}$ window is the point in times where

$$t_{l,start} = \max \left(\arg \max_{t \in w, t < t_{max}} P(t) < p_{max} \cdot \lambda_{upper} \right) \quad (10)$$

and

$$t_{r,start} = \min \left(\arg \max_{t \in w, t > t_{max}} P(t) < p_{max} \cdot \lambda_{upper} \right) \quad (11)$$

that is the maximum in the left window and minimum in the right window where $P(t)$ is smaller than the upper bound $p_{upper} = p_{max} * \lambda_{upper}$ denoting the start of the gray areas in Fig. 5.

This ensures that the windows do not contain the peak of the signal. This is the requirement for the next step: search the start and end of the maneuver. For that purpose, the thresholds p_{lower} and p'_{lower} are introduced. The first defines the maximum

probability and the latter the maximum probability change for the start and end of the maneuver. Hence, the start t_{start} of the maneuver is estimated by

$$t_{start} = \max \left(\arg \max_{t \in w, t < t_{l, start}} P(t) < p_{lower} \wedge P'(t) < p'_{lower} \right) \quad (12)$$

and the end t_{end} by

$$t_{end} = \min \left(\arg \max_{t \in w, t > t_{r, start}} P(t) < p_{lower} \wedge P'(t) < p'_{lower} \right) \quad (13)$$

denoted as black vertical lines in Fig. 5. The performance of the presented approach for maneuver extraction is evaluated in Sect. 7.

6 VDMS Architecture

The presented processing chain consists of multiple components each of which is part of the proposed Vehicle Data Management System (VDMS). For the sake of completeness, the modular and event-driven three-tier architecture presented in [11] described, based on the roles and their participation in the project and on the defined general requirements in Sect. 3.

6.1 Data Layer

The bottom layer *Data* contains all types of data in form of files on the file-system that are either generated or from 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 also includes files generated by components of the *Modules* layer, such as images extracted from the ADTF container, thumbnails of the images or JSON files representing the decoded CAN data of the ADTF containers.

The CAN signal data are extracted from the ADTF container to a generic JSON file 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 (Fig. 6).

The basic properties of the JSON format provide information about the *vehicle*, *driver* and the *time interval* of the measurements to easily match the conducted test drives to a specific test campaign and driver in the database. The time interval of the record is required since, by design, all signal value timestamps are relative to the start time. That enables to change the reference system even after the campaign, i.e. to synchronize the record times to an absolute reference system. This may be helpful if test drives are conducted in different time zones. 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. kilometer per hour). Besides that, it contains all timestamped values of that signal in the property *value* sorted by the signal value timestamps.

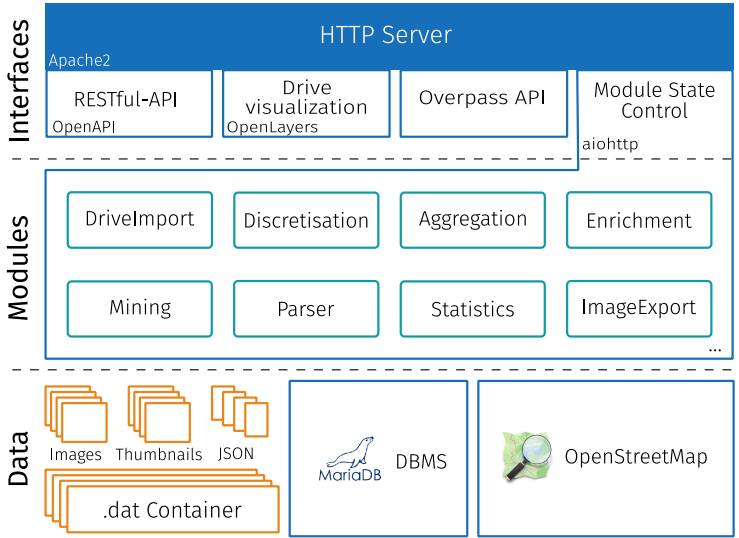


Fig. 6. The architecture of the proposed VDMS for managing large-scale test campaigns consists of three layers: *Data*, *Modules* and *Interfaces* [11].

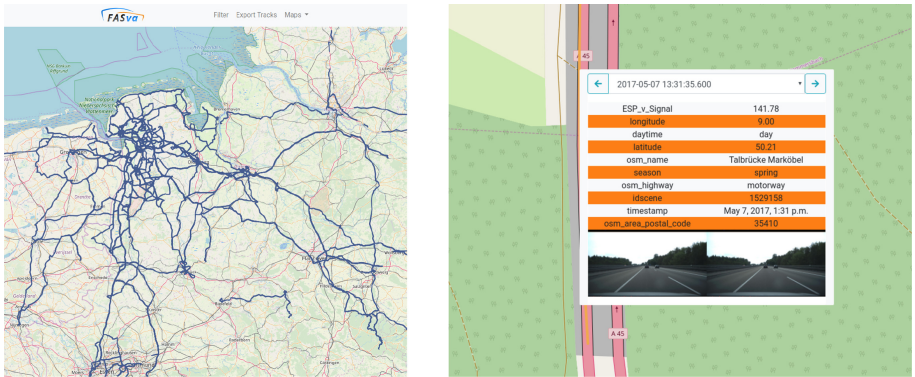


Fig. 7. The *drive visualisation* service provides an interactive web-interface of the conducted test drives including filter capabilities for efficient sequence identification and track selection for scene analysis. *Left:* A map of all conducted test drives. *Right:* The same zoomed-in map showing information of a selected scene [11].

6.2 Modules Layer

The second layer *Modules* entails all modules of the system. Each module has a distinct functional purpose and is 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 utilizing the observer pattern for asynchronously notify about module state changes. The latter is used for the *Module State Control* component of the layer *Interfaces* to notify connected clients, e.g. *test engineers*, about the progress of modules via WebSocket connections asynchronously. Whereas the former is used to trigger modules and thus, start the processing of a specific 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 implementation 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 to save the state of the module's progress in the database and to guarantee the reliability of the framework, i.e. robust the framework against misbehavior or errors in the modules such as memory leaks.

6.3 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 utilizes OpenStreetMap to show the conducted test drives via an interactive web-based frontend. The left image in Fig. 7 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 in the maneuver mining step to detect lane-changes on highways.

7 Experiments and Proof of Concept

To evaluate the proposed VDMS architecture and processing chain, the evaluation of driving functions and scenario identification is demonstrated by extracting lane changes and analyze an onboard DAS.

7.1 Dataset

The data basis for the following experiments are two sequences captured with our research vehicle and retrieved via the RESTful API presented in Sect. 6.3.

The first sequence covers approx. 100 km and takes place on a motorway with two to three lanes (see Fig. 9). During the trip, 16 lane changes occurred which were manually labeled using the *drive visualization* interface shown in Fig. 7. The scene duration within this sequence is 1 s, since according to [15], 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 since all lane changes are represented in the signal (see black crosses in Fig. 9).

The second sequence is another trip on a motorway with a duration of approx. One hour and total mileage of approx. 55 km. The trip contains 28 left and 30 right lane-changes. For each maneuver, the interval was labeled manually.

7.2 Maneuver Extraction

To evaluate the proposed approach for maneuver extraction, lane changes maneuvers are extracted from the second test sequence since the real maneuver intervals are known.

The prerequisite for maneuver extraction is the identification of the lane crossing event. For that purpose, a sliding window-based approach with a window size of 8 s is utilized to search for dominant peaks in the signals representing the distance to the left and right lane as depicted in Fig. 5. The duration of 8 s is used since [15] shows that the mean lane change duration is $\mu = 6.25$ s with a standard deviation of $\sigma = 1.64$. Although the approach is simple, it correctly identifies all available lane changes but has two false positives for left and right lane changes.

To assess the performance of the maneuver extraction approach, the extracted maneuver are matched against the manually labeled intervals. Therefore, a maneuver is correctly matched if the difference between the estimated and real start and end time

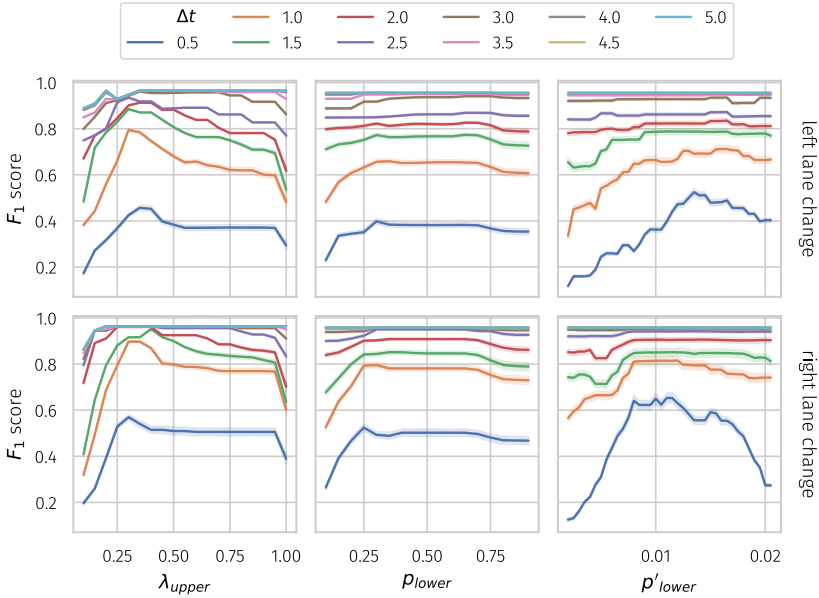


Fig. 8. The performance of the maneuver extraction algorithm depends on multiples thresholds. *Left:* The relative upper bound threshold. *Middle:* The maximum allowed probability p_{lower} for the end and start of the maneuver. *Right:* The maximum allowed probability slope p'_{lower} for the maneuver start and end.

is smaller than a maximum time difference $\Delta t = \{0.5, 1, \dots, 0.5\}$. To quantify the accuracy, the F_1 score with

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{14}$$

estimated. The precision and recall is defined as

$$\begin{aligned} \text{precision} &= \frac{t_p}{t_p + f_p} \\ \text{recall} &= \frac{t_p}{t_p + f_n} \end{aligned} \tag{15}$$

where t_p is the number of correctly matched way changes and f_p the number of unmatched maneuvers.

Since the approach depends on multiple thresholds, their influence on the performance is depicted in Fig. 8. It is evident that, the higher the maximum time difference Δt , the more accurate is the approach. Note that the F_1 score cannot be one since the sliding-window approach detected false positives.

Based on the given results, the optimal parameter for each maneuver can be estimated assuming the following requirements.

1. The offset between the extracted and true maneuver intervals should be as small as possible.
2. The computational complexity for finding the start and end of the interval should be minimized.

By applying these restrictions to the preliminary results, the estimated parameters and performance for left and right lane-change extraction are given in Table 1. The extraction of left lane-changes is accurate up to 3.0 s and right lane-changes can be extracted precisely with a maximum difference of 1.5 s. With this method, the parameters for each maneuver can be derived automatically based on ground truth information.

This demonstration shows that accurate maneuver extraction is possible with the proposed approach if the maneuver identification is precise. That enables *test engineers* to assess driving functions in specific sequences more efficiently, since this approach delivers the sequences and not only the point of time of a certain event.

Table 1. The best parameter constellation for each maneuver estimated by maximizing the F_1 score, minimizing the maximum time difference dt and search window.

	Statistics		Parameters		
	Δt	F_1	λ_{upper}	p_{low}	p'_{low}
Left lane-change	3.0	0.9655	0.95	0.075	0.0205
Right lane-change	1.5	0.9642	0.40	0.090	0.0205

7.3 System Evaluation

Besides the identification of scenarios, the assessment of a system under test (SUT) is another typical use-case. A *test engineer* might want to find those situations in which a SUT such as a Lane Keep Assist System (LKA) does not operate. In Fig. 9 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 Fig. 9 it is evident that on this sequence, the LKA stops operating if the driver performs a lane change (manually marked as black crosses). Hence, this system does not actively assist the driver during lane changes. Besides that, the system is also deactivated in two other cases with no lane-change maneuvers but lane crossing events. Based on this, one can conclude that the system is deactivated if the vehicle is on multiple lanes and not ultimately conducting a lane-change maneuver.

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 and analyzing onboard DAS.

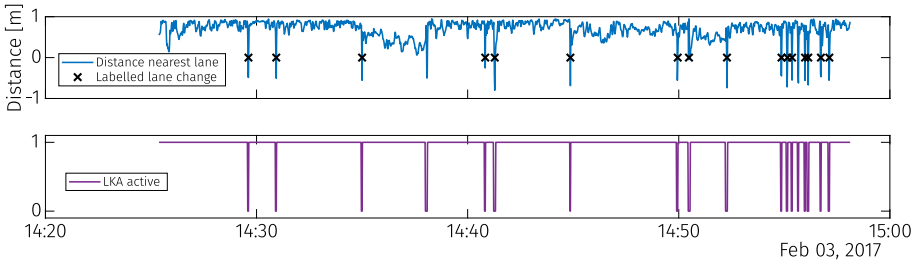


Fig. 9. Sequence of a trip on a motorway with two to three lanes for demonstrating the proof of concept of the proposed VDMS for system evaluation. *Top:* Distance to nearest line and the manually labeled lane changes. *Bottom:* The signal of the onboard Lane Keep Assist System.

8 Summary and Outlook

Test drives in the real world are mandatory to find relevant and critical scenarios for the validation of automated driving functions [4]. However, since the analysis of large-scale test campaigns requires computational assistance for efficient scenario identification, this work proposes a highly modularized three-tier VDMS. That enables to manage and analyses real-world test drives for the scenario-based validation of automated driving functions.

Based on a definition of the terms scene, maneuver, and scenarios, a formal definition of time-series of scenes is given. That is the foundation for the proposed processing chain for maneuver mining. The raw vehicle sensor data are aggregated to time series of scenes enriched with additional information from algorithms or external data sources that may not be available during test drives. The enriched scenes are partitioned into sequences by the maneuver mining component of the processing chain. For that purpose, a novel probability-based maneuver extraction approach is presented. Furthermore, a method is introduced to derive the optimal parameters for the maneuver extraction approach for each maneuver.

That processing chain is a central component of the VDMS, whereas the design of the architecture follows a requirements-driven approach utilizing software-quality characteristics defined in the ISO 25010. Therefore, the needs of particular project roles are analyzed and specific and general requirements on the architecture are derived.

The usability of the VDMS and the performance of the proposed maneuver extraction approach are finally evaluated by assessing an onboard DAS and finding sequences representing lane-change maneuvers. For that purpose, the RESTful API is utilized to retrieve sequences of the conducted test drives and label lane changes based on the images of the front camera.

The evaluation shows that the extraction of left and right lane-changes is accurate up to 3.0 and 1.5 s respectively. That enables test engineers to assess driving functions in specific sequences more efficiently since they can find maneuver sequences instead of only the time a maneuver occurred.

To compile a sophisticated set of scenarios, the focus in follow-up works is to integrate additional algorithms to identify typical maneuver such as vehicle approaching,

vehicle following and free driving [17]. Furthermore, the presented method for maneuver interval extraction needs to be validated with other algorithm.

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

1. Bach, J., Otten, S., Sax, E.: Model based scenario specification for development and test of automated driving functions. In: 2016 IEEE Intelligent Vehicles Symposium (IV), pp. 1149–1155, June 2016. <https://doi.org/10.1109/IVS.2016.7535534>
2. Baek, S., Kim, D.Y.: Empirical sensitivity analysis of discretization parameters for fault pattern extraction from multivariate time series data. *IEEE Trans. Cybern.* **47**(5), 1198–1209 (2017). <https://doi.org/10.1109/TCYB.2016.2540657>
3. Benmimoun, M., Fahrenkrog, F., Zlocki, A., Eckstein, L.: 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 (2011)
4. Damm, W., Möhlmann, E., Peikenkamp, T., Rakow, A.: A formal semantics for traffic sequence charts. In: Lohstroh, M., Derler, P., Sirjani, M. (eds.) *Principles of Modeling*. LNCS, vol. 10760, pp. 182–205. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-95246-8_11
5. Elrofai, H., Paardekooper, J., Gelder, E.d., Kalisvaart, S., Op den Camp, O.: Streetwise: scenario-based safety validation of connected automated driving. Technical report TNO (2018)
6. Fraenkel, D.J., Cowie, M., Daley, P.: Quality benefits of an intensive care clinical information system. *Crit. Care Med.* **31**(1), 120–125 (2003)
7. Frehner, M., Brändli, M.: Virtual database: spatial analysis in a web-based data management system for distributed ecological data. *Environ. Model. Softw.* **21**(11), 1544–1554 (2006)
8. Haja, A., Koch, C., Klitzke, L.: 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)*, pp. 320–325 (2017). <https://doi.org/10.5220/0006354103200325>
9. Kalra, N., Paddock, S.M.: Driving to safety: how many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transp. Res. Part A Policy Pract.* **94**, 182–193 (2016)
10. Klauer, S.G., Dingus, T.A., Neale, V.L., Sudweeks, J.D., Ramsey, D.J., et al.: 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 (2006)
11. Klitzke, L., Koch, C., Haja, A., Köster, F.: Real-world test drive vehicle data management system for validation of automated driving systems. In: *Proceedings of the 5th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2019)*, pp. 171–180, May 2019. <https://doi.org/10.5220/0007720501710180>
12. Liu, H., Hussain, F., Tan, C.L., Dash, M.: Discretization: an enabling technique. *Data Min. Knowl. Disc.* **6**(April), 393–423 (2002). <https://doi.org/10.1023/A:1016304305535>
13. Menzel, T., Bagschik, G., Maurer, M.: Scenarios for development, test and validation of automated vehicles. In: 2018 IEEE Intelligent Vehicles Symposium (IV), pp. 1821–1827, June 2018. <https://doi.org/10.1109/IVS.2018.8500406>

14. Moskovitch, R., Shahar, Y.: Classification-driven temporal discretization of multivariate time series. *Data Min. Knowl. Disc.* **29**(4), 871–913 (2014). <https://doi.org/10.1007/s10618-014-0380-z>
15. Olsen, E.C.B., Lee, S.E., Wierwille, W.W., Goodman, M.J.: Analysis of distribution, frequency, and duration of naturalistic lane changes. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 46, no. 22, pp. 1789–1793 (2002). <https://doi.org/10.1177/154193120204602203>
16. Pütz, A., Zlocki, A., Bock, J., Eckstein, L.: System validation of highly automated vehicles with a database of relevant traffic scenarios. In: *12th ITS European Congress. ITS European Congress (2017)*
17. Roesener, C., Fahrenkrog, F., Uhlig, A., Eckstein, L.: 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)*, pp. 1360–1365, November 2016. <https://doi.org/10.1109/ITSC.2016.7795734>
18. Schneider, J., Wilde, A., Naab, K.: Probabilistic approach for modeling and identifying driving situations. In: *2008 IEEE Intelligent Vehicles Symposium*, pp. 343–348, June 2008. <https://doi.org/10.1109/IVS.2008.4621145>
19. Schuldt, F.: Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen. Ph.D. thesis, Technische Universität Carolo-Wilhelmina zu Braunschweig, April 2017. <https://doi.org/10.24355/dbbs.084-201704241210>
20. Shavit, E., Teichner, L.: Interactive market management system, US Patent 4.799.156, January 1989
21. Sonka, A., Krauns, F., Henze, R., Küçükay, F., Katz, R., Lages, U.: Dual approach for maneuver classification in vehicle environment data. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 97–102, June 2017. <https://doi.org/10.1109/IVS.2017.7995704>
22. Stellet, J.E., Zofka, M.R., Schumacher, J., Schamm, T., Niewels, F., Zöllner, J.M.: 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*, pp. 1455–1462, September 2015. <https://doi.org/10.1109/ITSC.2015.236>
23. Ulbrich, S., Menzel, T., Reschka, A., Schuldt, F., Maurer, M.: Defining and substantiating the terms scene, situation, and scenario for automated driving. In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pp. 982–988, September 2015. <https://doi.org/10.1109/ITSC.2015.164>
24. Weidl, G., Madsen, A.L., Kasper, D., Breuel, G.: Optimizing Bayesian networks for recognition of driving maneuvers to meet the automotive requirements. In: *2014 IEEE International Symposium on Intelligent Control (ISIC)*, pp. 1626–1631, October 2014. <https://doi.org/10.1109/ISIC.2014.6967630>
25. Winner, H., Lemmer, K., Form, T., Mazzega, J.: PEGASUS—first steps for the safe introduction of automated driving. In: Meyer, G., Beiker, S. (eds.) *Road Vehicle Automation 5*. LNM, pp. 185–195. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-94896-6_16
26. Wissing, C., Nattermann, T., Glander, K., Bertram, T.: Probabilistic time-to-lane-change prediction on highways. In: *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1452–1457, June 2017. <https://doi.org/10.1109/IVS.2017.7995914>
27. Zhao, D., Guo, Y., Jia, Y.J.: TrafficNet: an open naturalistic driving scenario library. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–8, October 2017. <https://doi.org/10.1109/ITSC.2017.8317860>