Lanelet2: A high-definition map framework for the future of automated driving

Fabian Poggenhans[1], Jan-Hendrik Pauls[2], Johannes Janosovits[2], Stefan Orf[1], Maximilian Naumann[1], Florian Kuhnt[1] and Matthias Mayr[1]

***Abstract*— Although accurate and comprehensive maps are indispensable for highly automated driving, especially in complex urban scenarios, there are hardly any publications in which requirements for these maps are discussed. In our opinion, such maps must meet high demands in terms of accuracy, completeness, verifiability and extensibility, so that the resulting complexity can only be handled by an enclosing, carefully designed software framework. In this paper we therefore introduce the open-source map framework Lanelet2 implemented in C++ and explain the underlying concept. The goal of Lanelet2 is not only to be usable for typical, isolated applications such as localization or motion planning, but for various potential applications of maps for highly automated driving. On the basis of both abstract and real examples we show the concrete structure of Lanelet2 maps and its use for automated driving.**

I. INTRODUCTION

There is no doubt that maps are essential for highly automated driving. High-definition maps, which provide accurate information about the surroundings of a vehicle, were an essential component in all major automated driving projects (cf. [1]–[4]) so far. The reason for this is not only that it compensates for the inadequacies of the sensors, but also that maps are important for providing information about regions that cannot be observed by sensors, whether due to occlusion or insufficient sensor range. In addition, a correct interpretation of the observed data is required, which copes with the uncertainties of the sensors. Maps provide the reliable information needed to understand a scene correctly. Finally, maps allow to transfer knowledge from previous journeys and thus represent an additional level of redundancy.

With increasing progress in the field of automated driving, the maps have to be refined more and more and at the same time meet higher and higher quality requirements. While the coarse road course is sufficient for navigation devices and the position and number of lanes can be sufficient for partially automated driving on highways and most country roads, maps for the city center must provide considerably more information. The term "high-definition" (HD) map is often misleadingly used for the latter two, but it is not precisely defined. In this paper we therefore use the term *lane-level accurate map* for the first, while maps that enable safe highly automated driving (HAD) even in complex city center scenarios will be referred to as *HAD maps*.

[1]FZI Research Center for Information Technology, 76131 Karlsruhe, Germany {poggenhans, orf, naumann, kuhnt}@fzi.de

[2]Institute of Measurement and Control Systems, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany {jan-hendrik.pauls, johannes.janosovits}@kit.edu

For highly automated vehicles, not only the lane itself is relevant. Information about the complex environment of a vehicle such as bicycle lanes and sidewalks must also be made available and should therefore be provided by a map. As a result, many parts of the processing chain of a highly automated vehicle depend on precise map information, but each with a different focus (see Fig. 1). The elements of the road on which the map is based and their exact position, such as markings, traffic signs and curbs, are relevant for localizing as well as validating the map. During behavior generation, however, the relevance of these elements for the vehicle is important, e.g. which traffic light is valid for a particular lane. For the prediction of other road users it must be clear which traffic rules apply to them and where they can move to. This is especially true for pedestrians, whose behavior is difficult to predict with sensors alone [5]. Detailed routing requires knowledge of all lanes and where they lead. It must be clear where lane changes are possible or even mandatory. Other requirements are special situations such as emergency or parking maneuvers. For this purpose, the map must provide information beyond the actual lanes. For the simulation of automated vehicles, maps must be very detailed to be able to test under the most realistic conditions. In addition, high quality standards must be met: maps must be complete and accurate, up-to-date and verifiable within a few centimeters to ensure safe automated driving.

In view of the increasing complexity of HAD maps, it is also clear that it is not sufficient to simply store the relevant information in a map. In order to be able to consistently interpret the information and to determine its implications for the ego vehicle and other road users, good software support in form of a special software framework is necessary to ensure a consistent view of the map. Ideally, such a framework should provide simple interfaces for the various different perspectives from which the data contained in a map can be viewed.

Given the high demand for highly accurate maps for many different applications, it is all the more surprising that there are hardly any publications discussing requirements for maps for HAD. Accordingly, there are no map frameworks that meet the demand. Commercial map providers claim to already provide HAD map data, but we will show that these maps do not meet the requirements. In addition to the completeness of the map information, an easy evaluation of the data during the journey is also decisive. We would like to deal with both aspects in the following pages which form the basis of the map framework *Lanelet2*.

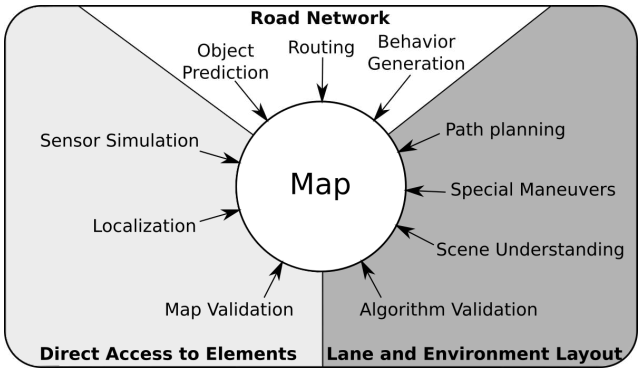

Fig. 1: Applications and required interaction with HAD maps

II. RELATED WORKS

A. Existing Map Formats and Frameworks

Most available map formats used for automated driving have their origins in the time before this became a popular topic. These include, for example, the maps of commercial providers TomTom and Here [6], [7], but also the free mapping project OpenStreetMap (OSM) [8]. Due to the increased requirements, providers are inevitably pursuing a top-down approach: roads are represented by an imaginary center line, as is already used in navigation devices. More information about lanes, for example, is added by adding certain attributes to this center line. For example, the position of traffic lanes and the form of the road border are also stored. As requirements increase, additional attributes are added. This leads to an extremely complex, implicit representation of the information, since, for example, the absolute position of the roadsides can only be determined indirectly by offsetting the center line left and right with the width of the road. This complexity is further increased in intersections, as there is no clear center line. In OSM as an open format, the quality of the data varies greatly [9], as these are created and maintained by volunteers. However, suitable, freely available editors exist to supplement existing maps. For the commercial map providers, no editors or software libraries are publicly available.

The OpenDRIVE [10] framework occupies a similar position. This format was originally developed for a uniform description of the road and its surroundings in driving simulators, but is now also used for HAD. The information is represented very similarly to the formats already mentioned. Although the specification is open, there are no freely available libraries for interpreting and processing the data.

The above-mentioned map formats with lane-level accurate information have also been successfully used for driving on motorways. An example of this is [4]. However, the authors themselves conclude that “there is still a lot of work to be done, especially in the area of validation/certification and the generation of large-scale digital maps”.

Due to the shortcomings of existing map formats, map representations have been introduced and used in recent years that are specially tailored for a particular application or publication. Consequently, there are few further publications in which these formats are used. Examples include the *Unified Map* [11], which was developed for trajectory planning. Here, moving objects, road boundaries and traffic rules, such as traffic lights, are uniformly represented as obstacles that restrict vehicle movement.

Another example can be found in [12]. The aim of this publication is to show uncertainties in the representation of a map created by the sensors while driving. Similar to the already mentioned map formats, the road is represented here as a projection of all elements onto the center line. Intersections are not covered.

Most of the remaining publications where new map formats are introduced deal with localization. Most likely because accurate localization requires very detailed information about observable landmarks. The map formats mentioned above do not provide sufficient support for this. An outstanding example is [13], because it not only introduces a map representation, but also formulates requirements that a map format suitable for HAD must meet. In addition to landmarks suitable for localization, the main requirements listed are information on the geometry and direction of travel of the lanes, especially in intersections, information on height profiles and the resulting occlusions, and information on the prediction of other road users.

The map framework *Liblanelet* [14] follows a very different approach. For the first time used for Bertha-Benz drive [1], this is a map format that was specially developed for HAD, especially for motion planning. It is based on atomic lane sections, *Lanelets*, which together form the road through their neighborhood relationships. This format made a simple, explicit representation of the lanes possible. Traffic rules are represented by so-called *Regulatory Elements*, which represent, for example, right of way, traffic lights and stop lines. *Liblanelet* also offers a library in C++ with interfaces for routing, motion planning and prediction of other vehicles. However, the format has certain weaknesses related to the fact that it was designed for use with a specific, previously known route. For example, lane changes are only possible at predefined points. Overtaking is therefore also not possible. Furthermore, many other fields of application described in Fig. 1 are not supported, such as localization or special maneuvers. In this paper we therefore present a generalization of this map format that eliminates these weaknesses.

B. Usage of Maps

To find out how maps are used in literature and what requirements have been placed on them, we have evaluated 32 publications of the last five years from various research directions in HAD that explicitly state the use of high-resolution maps. Most of them (13) dealt with the topic of localization, five with the topic of map generation, four described a system architecture, four dealt with motion planning and four with prediction. One dealt with scene understanding, one with behavior generation and detection of map deviations. Most approaches (13) used a self-created map format, especially for localization methods, where the map was often created using the same technique. This

already indicates that there is a certain lack of suitable map formats. Most of the other publications used OSM (7), although in three cases the format was extended to provide more information. In five approaches *Liblanelet* was used. They dealt with motion planning, localization and prediction of other road users. Two approaches used OpenDRIVE, one maps from HERE. In the other four cases, the map format was not described.

This small investigation suggests that maps are indeed an important component in many applications. However, the fact that mostly self-created map formats are used for this or existing ones are adapted, clearly indicates a deficiency. The work that is spent in the ever new construction of own, specialized map formats could be used much more meaningfully – for example by contributing to a common, public library. Therefore, in the following section we will give some thought to the requirements that a map framework for highly automated driving must meet and how this can be solved with *Lanelet2*.

III. DESIGN CONSIDERATIONS

As already mentioned in Section I, HAD places many demands on maps and how information is represented in them. In this section we will present essential requirements and our proposed solutions by looking at common applications in automated driving that make use of maps and identifying their requirements. These applications can be divided into three groups (see Fig. 1): Applications that focus more on the road network, applications that need detailed information about lanes and their environment, and applications that need direct access to the individual physical elements of a map. Each of these three groups has different requirements.

A. Road Network

One of the most obvious applications is *routing*. This is of course already a standard feature of many map formats, but HAD requires greater accuracy. It is not enough to know which roads make up a route. It must also be clear which lanes within the road can be used and which, for example, lead to turning lanes or are reserved for other road users such as buses. Previous knowledge of the course of the individual lanes can also help to minimize the number of lane changes required during a journey. In addition, it must be clearly visible where lane changes are possible. Especially on urban roads, and complex intersections, lane changes are often temporarily forbidden in one or both directions. Also, it must be easy to find alternative lanes or routes if the selected lane is blocked.

In *behavior generation*, a choice is made between different maneuvers, such as overtaking, merging or braking maneuvers in front of traffic lights and pedestrian crossings. This requires precise knowledge of right of way rules in order to generate behavior that is consistent with traffic regulations. Therefore, an exact knowledge of a chosen route is required, together with the traffic rules that apply to it.

A similar task is the *prediction* of other road users. Under normal conditions it can be assumed that they either continue to move within their lanes or change lanes if traffic regulations permit this. Similarly to behavior generation, the set of possible maneuvers must be determined. For this purpose, however, the rules must be taken into account depending on the type of road user. For example, special traffic rules often apply to buses, which allow them to use additional turn-offs or separate lanes. A map should therefore contain traffic rules for any type of road user, not only for the ego vehicle. This includes in particular pedestrians, cyclists or trams. These do not move on lanes, but a map must be able to tell where, for example, pedestrians can be expected and in which directions they can move.

B. Lane and Environment

Precise knowledge of the lane geometry is crucial for *path planning*. Knowing the center line alone is not sufficient, as the trajectory should be able to be adjusted depending on the speed, for example in curves for comfortable driving. In the city center, obstacles such as parking vehicles protruding into the lane often have to be avoided without endangering oncoming traffic. Since the localization and the actuators of a vehicle are also not arbitrarily precise, it must also be known which distance to the center line can still be tolerated without leaving the lane. As already suggested in [14], it therefore makes sense to map the lanes completely with their left and right borders.

An often forgotten problem, however, is that highly automated driving does not only involve normal driving on a public road, but also *special maneuvers*. An example of this is parking. To do this, appropriate parking facilities must first be known, but also extensive information must be known for reaching a parking space. For example, which areas can be used on the way to a parking space or which areas are suitable for maneuvering without obstructing oncoming traffic. Another example are emergency situations where the vehicle has to evade another vehicle or a safe position has to be reached after a sensor failure. For this purpose, map data may not be limited to the lanes themselves, but must also contain detailed information about the surroundings of a road.

C. Physical Elements

As already mentioned in Section I, *localization* is a typical case where observable elements are important. For this purpose, a map must contain elements that can be observed by as many sensors as possible in order to be usable for vehicles with different sensor setups. At the same time a sufficient density of such elements must be offered so that an exact localization is possible everywhere. Elements such as markings, crash barriers and roadsides can be used for this, as they can be found on many roads and are also a basic component of corridors, so that a map offers inherently consistent information for localization and motion planning.

A major challenge for maps is however to ensure that the data is *up to date*. The more details a map contains, the more likely it becomes that a map needs to be updated. Today's map formats often do not contain the underlying

data, but only an abstract, pre-processed form. However, if the environment changes, it is no longer possible to trace which parts of the map are affected. For example, removing a speed limit sign may mean that this speed limit no longer applies to the following road section. But there may also be another sign (for example on the other side of the road) that still enforces this speed limit. If only the derived information (the speed limit) is stored on the map, it is no longer possible to determine whether the traffic sign was the source of the traffic rule or not. The map should therefore not only contain the derived information, but also allow conclusions to be drawn as to where this information came from.

The same applies to lanes: Structural changes to a road can only be detected in time by *change detection* algorithms if it is known what limited the lane. In a subsequent step, maps could also be updated automatically by deriving the new, correct lanes from the changed lanes that a vehicle has observed.

Besides local changes, there is another challenge to ensure the map's correctness over time: Physical elements need to be stored and referenced in a way that is robust against environmental changes such as continental drift. The obvious solution is using a locally fixed reference frame instead of using global coordinates (like GPS).

D. Consequences for Maps

We call the detectable elements mentioned above the *physical layer* of a map. It consists of real, observable elements, such as markings, curbstones, traffic lights, etc. All other elements, such as lanes, are only associations of these elements to a more abstract representation. Likewise, traffic rules are associations between the source of a rule, such as traffic lights, and the lanes to which they apply. This representation allows the information on a map to be traced. We call this layer the *relational layer* of a map.

This results in further advantages: For traffic rules it becomes transparent, to which type of road user they refer. For this purpose, it must only be comprehensible whether the element associated with a rule is also valid for this road user. Accurate, map consistent localization can also be ensured, since elements of the physical layer can be detected by definition. By matching these detections with the map, the position within it can be determined or at least validated. Furthermore, the information can also be used to simulate and therefore validate highly automated vehicles on real routes. Ideally, sensor observations can also be simulated based on this.

However, not all elements of a map can be associated to observable objects, as implicit rules also occur in traffic. At intersections, for example, there are traffic lanes whose borders are not defined by markings. Nevertheless, human traffic participants move within an imaginary corridor, which must therefore also be part of a map. Still, elements in a map based on such purely conventional behavioral patterns must be avoided as much as possible, as they are difficult to verify and behavioral patterns can change, too.

The requirements for maps for HAD are constantly changing as new algorithms and processes are being developed. At the same time, the map formats must be valid for many different countries with differing traffic rules. It must therefore be ensured that the map format is expandable and flexible so that it can be adapted to changing requirements. This applies in particular to the traffic rules contained therein.

The high degree of complexity of the resulting maps requires suitable software interfaces to ensure a uniform, consistent interpretation. We have described above that the representation of the elements of a map should be separate from their interpretation. This requirement also applies to the software implementation of such a framework. This makes it possible to interpret the traffic rules stored in a map dynamically, depending on the road user and her abilities. In the following sections we would therefore like to present how we designed *Lanelet2* to follow these principles.

In summary, we consider the following principles to be indispensable for future HD map formats for HAD:

- verifiability of all information in the map while driving by associating it to observable objects,
- coverage of all potentially passable areas, even off the road,
- the interactions between the individual lanes and regions on the map must be identifiable and comprehensible. It must be possible to find out between which lanes lane changes are possible or where conflicts can arise due to crossing lanes,
- information on areas used by other road users and the rules applicable to them,
- separation between sources of traffic rules and their implication on road users,
- expandability/modularity,
- modifiability to easily perform updates.

IV. ARCHITECTURE OF LANELET2

The map framework *Lanelet2* has been developed with the requirements of Section III in mind. It is an extension and generalization of the map format *Liblanelet* [14] developed for the Bertha-Benz journey. As indicated in the previous section, the map is divided into a *physical layer*, which contains the usually real observable elements and a *relational layer*, in which the elements of the physical layer are connected to lanes, areas and traffic rules. A third layer results implicitly from the contexts and neighbourhood relationships of the relational layer: the *topological layer*. Here, the elements of the relational layer are combined to a network of potentially passable regions depending on the road user and situation. It would be conceivable and possible to derive the elements of the relational layer from the physical layer (as humans do), but currently no methods exist to make this possible. However, such a procedure is at least made possible by this presentation.

The format is based on the format known from *Liblanelet* and was designed to be representable on the XML-based OSM data format, for which editors and viewers are publicly available. However, we consider the actual data format of a

map to be irrelevant and interchangeable as long as it is ensured that this format can be transferred to the internal representation without loss. A conversion to other formats would therefore be easy.

We assume that all elements in the map can be described predominantly by a projection onto a flat ground plane. Usually, this requirement is met for all elements in the vicinity of a road. Nevertheless, height information is important, for example, to determine the height profile of a road [13] or to prevent a crossing bridge from being misinterpreted as an intersection. Therefore, a height coordinate is also required.

For storing map data, the most important requirement is correctness. This means not only immutability when only reading the map, but also robustness against influences like continental drift. Thus, a locally fixed reference system like ETRS89 in Europe is used with lossless geographic coordinates (latitude/longitude). When loading the map, geographic coordinates are transformed into a local, metric coordinate system in order to be able to perform efficient calculations. For this working representation, the UTM (Universal Transverse Mercator) coordinate system offers itself.

A. Lanelet Primitives

A *Lanelet2* map consists of five elements: *Points* and *line strings* belonging to the physical layer and *lanelets*, *areas* and *regulatory elements* belonging to the relational layer. All elements have in common that they are identified by a unique ID (this is useful for an efficient construction of the topological layer) and that attributes in the form of key-value pairs can be assigned to them. Some of these attributes are fixed, but additional attributes can be used to enhance the map. With the exception of areas, all primitives were also already part of *Liblanelet.* However, in particular, the definition of lanelets has been modified and generalized to meet the aforementioned requirements.

Points are the basic element of the map. Single points represent, for example, vertical structures such as poles. Usually, however, they are part of line strings. Each point is described by its three-dimensional position in metric coordinates. Points are the only primitives that actually have position information. All other primitives are directly or indirectly composed of points.

Linestrings are an ordered array of two or more points between which linear interpolation takes place. It is used to represent the shape of elements in the map. Examples are road markings, curbs, facades, fences, etc. Linestrings can also be *virtual*, for example, if they form an implicit border of a lane. Linestrings were chosen as a form of representation, because they can be used to describe any one-dimensional form, if necessary through high discretizations. Compared to splines, they can be calculated efficiently and can be used to represent sharp corners. Viable solutions exist for the problem of non-linear differentiability (see [15]).

Lanelets define an atomic section of the map in which directed motion takes place. Examples of this are normal lanes, but also pedestrian crossings and rails. Atomic means that currently valid traffic rules do not change within a lanelet, but also that the topological relationships with other lanelets does not change. A lanelet is defined by exactly one line string as the left and exactly one as the right border. In addition, it may have several *regulatory elements* expressing traffic rules applying to this lanelet. Lanelets can also overlap or intersect. The type of the left or right border expresses whether, for example, lane changes to an adjacent lanelet are possible. Successive lanelets share the end points of the left and right border. Within a lanelet, movement in the opposite direction can be allowed, so that the left border becomes the right (and vice versa).

Areas are sections of the map in which undirected or no movement is possible. Examples include parking areas, squares, green spaces or buildings. They are defined by one or more linestrings, which together form a closed outer border, and can have one or more line strings, which together define several inner borders and thus holes within the area. Similar to lanelets, these can also have regulatory elements.

Regulatory elements (short: regElem) define traffic rules, such as speed limits, priority rules or traffic lights. A regulatory element is always referenced by one or more lanelets or areas for which they apply. Because there are very different types of traffic rules, the exact structure of a regulatory element can be very different. They usually reference an element that defines the rule (e.g. a traffic sign) and, if necessary, an element that cancels the rule (e.g. a sign at the end of a speed zone). In addition, stop lines can be referenced, for example. The attributes of a rule element specify what kind of rule it is. RegElems can also be dynamic, which means that a rule is only valid based on a condition. Examples include digital speed indicators or traffic rules that are only valid at certain times of the day.

B. Modules

As explained in Section III-D, it is important to separate the representation of the elements of a map from their interpretation. *Lanelet2* is therefore based on a consistent modularization of the individual tasks. This also ensures the extensibility of the framework. Representation is solved by the *core* module, which contains the above-mentioned primitives. There is the *traffic rules* module for interpretation, with which the traffic rules can be queried depending on the road user and country. For example, it decides which maximum speed may be driven on a lanelet and whether a lane change between two lanelets is possible. Based on these modules, there are further modules specialized for the individual applications, such as for routing, matching and access to the individual physical elements. In addition, there are other modules that provide interfaces for common software frameworks, such as Python, ROS [16] or for reading and writing to certain file formats.

In summary, *Lanelet2* is separated into the following modules:

Core: The core module contains the basic primitives and layers described above. In addition, geometry functionalities are included to calculate center lines, distances and overlaps, for example.

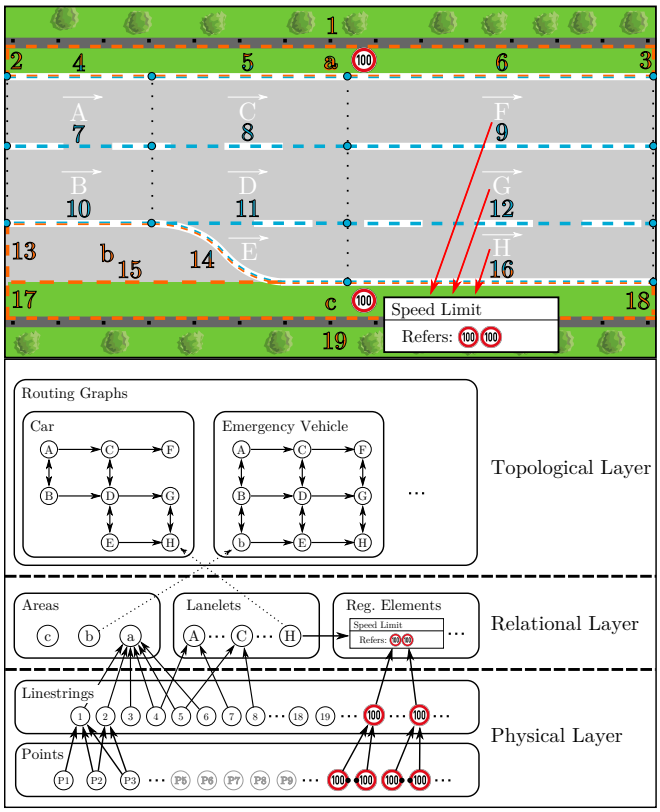

Fig. 2: Map example for a highway road (enclosed by guardrails) and the resulting map structure. Lanelets have capital letters, areas lowercase letters and linestrings have numbers.

Traffic Rules: This serves to interpret the rules contained in the map depending on the type of road user and country. It also determines whether lane changes are possible or whether a certain road user is permitted to enter a lanelet.

Physical: This module allows direct, comfortable access to the elements of the physical layer.

Routing: With the help of traffic rules, routing graphs can be set up to determine the exact route to be driven, including possible lane changes, or to predict possible routes and points of conflict for other road users. It is also possible to construct maneuverable zones by combining adjacent areas and lanelets.

Matching: This module is used to assign lanelets to road users or to determine possible positions on the map based on specific observations of the sensors.

Projection: Contains functionality to convert global latitude/longitude coordinates to local, metric coordinates.

IO: The input/output module contains functions for reading and writing maps from various map formats, in particular the OSM format.

Validity: Module that searches and reports typical mapping errors in a map.

ROS: A connection to the Robot Operating System (ROS) which is often used in demonstrators.

Python: For using the above modules in Python.

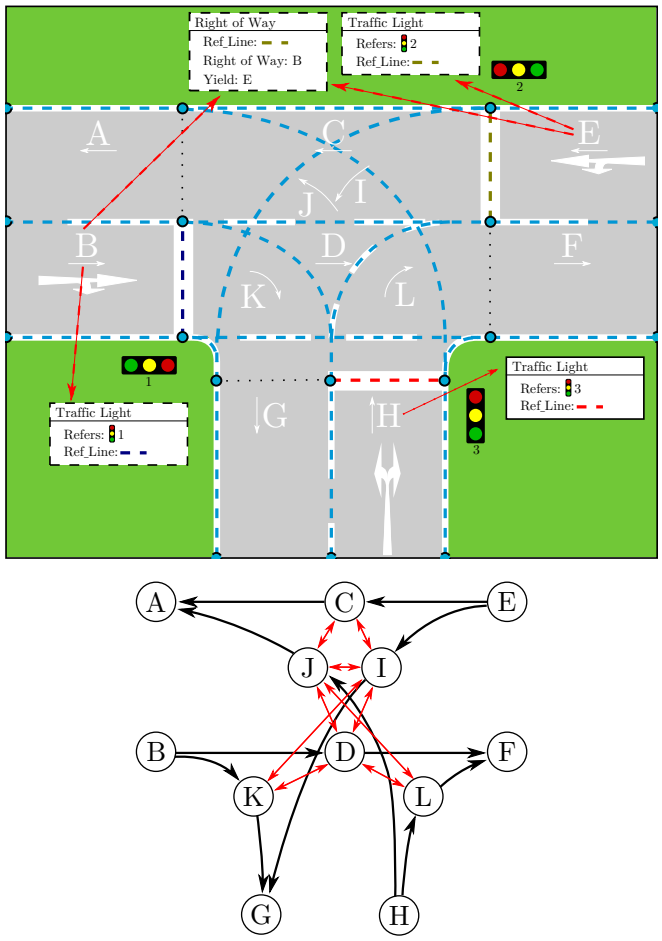

Fig. 3: Map example for an intersection and resulting routing graph for normal vehicles. Conflicting lanelets are shown in red in the graph.

V. CASE STUDIES

In this section we would like to illustrate with some examples how a lanelet map can be constructed. For this purpose we first show the basic concept by means of some typical scenarios and then present more complex real examples to demonstrate the applicability on them.

A. Scenarios

Fig. 2 shows a *Lanelet2* map example for one direction of a motorway, which is extended by one lane. Since this lane is added to the side, it can only be reached by changing lanes. The map also contains several areas: one for the emergency lane, and two for the green strip between the road border and the guardrail. The latter may seem somewhat pedantic, but are intended to serve as an example for areas that can be included in emergency maneuvers, for example.

Two routing graphs a shown that might result from this scenario: One for a normal vehicle and one for an emergency vehicle. Because emergency vehicles are allowed to use the emergency lane b and might do a lane switch from F to G, the routing graph contains more edges.

Note also the division of the lanelets E and H. This is necessary, as with the addition of the continuous line no lane

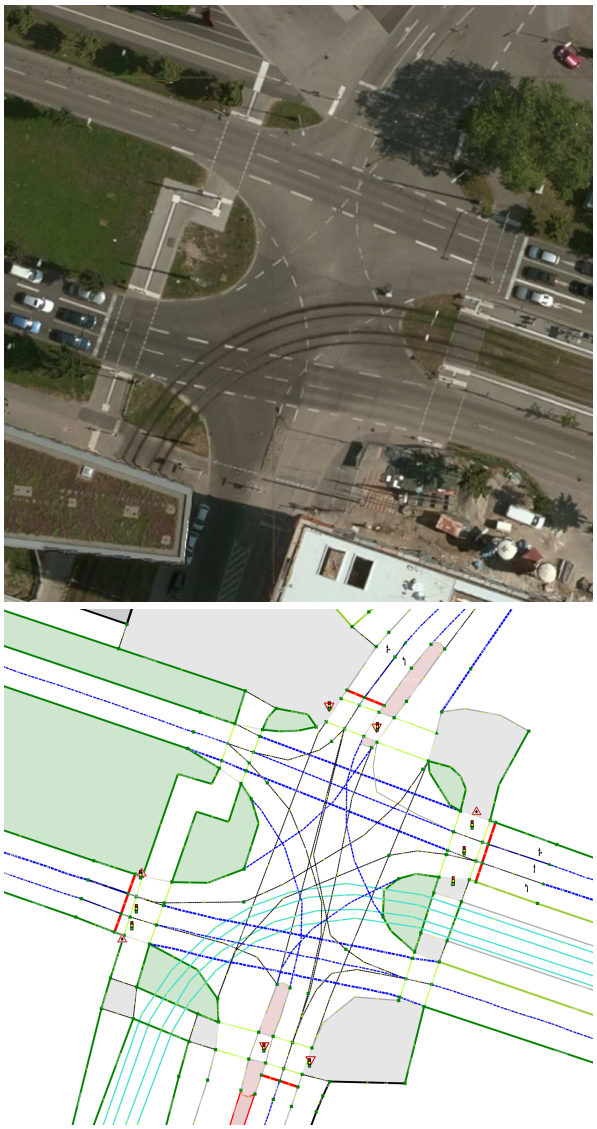

Fig. 4: Complex intersection in the east of Karlsruhe, Germany and corresponding map. Lane markings are visualized in blue, curbs in gray, road borders in green and virtual lines in black. Areas for pedestrians are shown in gray, green stripes in green.
Aerial image: © CNES Distribution Airbus DS, HERE.

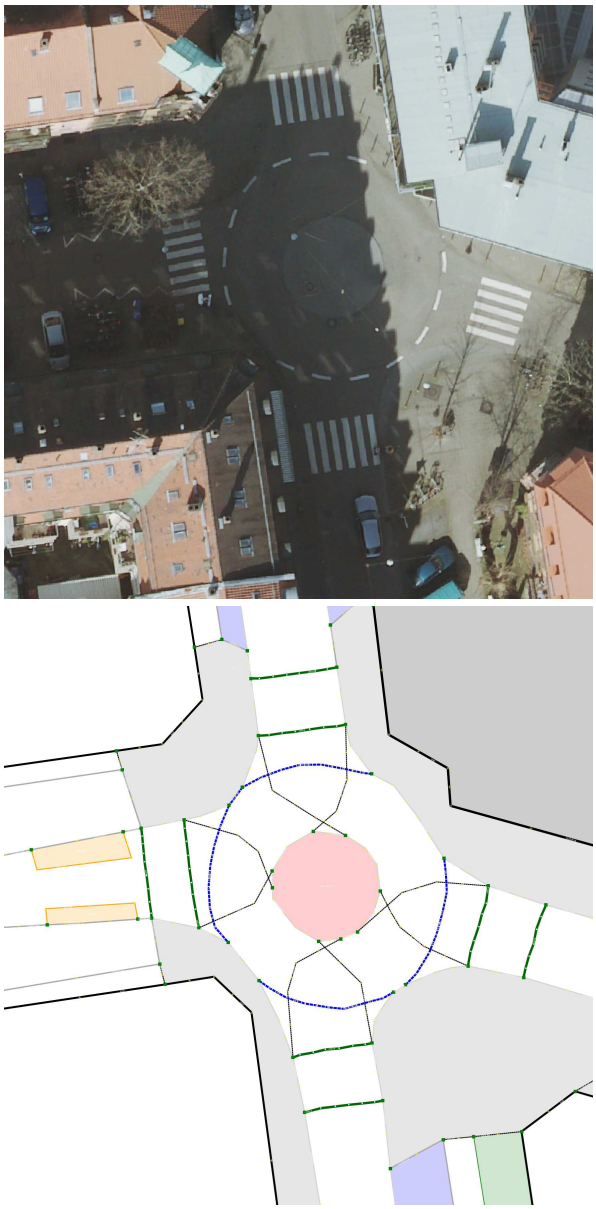

Fig. 5: Small roundabout in an urban area in Karlsruhe and corresponding map. In addition to the above, walls are shown in black, traffic islands as red areas and parking spaces as blue areas.
Aerial image: © Stadt Karlsruhe | Liegenschaftsamt

change from G to F is possible. If E and H were one single lanelet, this would allow the conclusion that a lane change from the former H to C would be possible, since these are now connected via D. The restriction that lanelets may only have one right edge forces that such a situation cannot arise, since the splitting of the lanelets D and G forces that also the right edge is separated, which propagates further to the lanelets E and H.

Fig. 3 is an example of an intersection. Since the intersection does not contain road markings for some of the turning lanes, the lanelets G, H and I partly have one virtual line as edge. Furthermore, the routing graph contains relationships between conflicting lanelets, so that potentially critical situations can be determined.

Because the intersection is controlled by traffic lights, the lanelets B, E and H each have a regElem that references the corresponding traffic light and the corresponding stop line. Because vehicles turning left from E have to yield to the vehicles going straight from B, the lanelet E and B both have a *right of way* regElem that references B as the lanelet with right of way and E as lanelet that has to yield. The *right of way* regElems always reference lanelets right at the entrance of the intersection where vehicles can be assigned unambiguously to one single lanelet. If lanelets intersect without a right of way regElem, both lanelets have equal priority and the first of two vehicles has the right of way.

B. Real World Example

A more complex real-world example for an intersection in Karlsruhe, Germany is shown in Fig. 4. The map also includes footpaths and tram tracks that cross the intersection from right to bottom. The routing graph for this section is too complex and is therefore not displayed.

Fig. 5 shows the map of a roundabout in the east of Karlsruhe. It lies in the middle of roads that can be driven on in both directions and have no dividing middle line. These roads are represented by a single lanelet each, but the routing graph contains two nodes, one for each drivable direction.

VI. CONCLUSION AND FUTURE WORK

In the past pages we have shown what a framework for HAD can look like and demonstrated that it is applicable to practical situations. Maps for HAD have so far received little attention in scientific publications. However, we believe that this topic is too important for the future of HAD to be left entirely to commercial map suppliers.

An important element that maps for highly automated driving have so far lacked is the traceability of the information stored in them. This principle requires a fundamentally different structure, so that even very abstract components of maps, such as the road network, can be traced back to their basic origins, such as the border of a traffic lane. This is the only way to ensure the validity and updatability of a map.

We also propagate the change from a map structure that focuses solely on the representation of information to an application-oriented structure that provides the respective functionalities in HAD with exactly the information they need through clear interfaces. This makes it possible to store significantly more complex information and relations on a map without making its simple use impossible. A good map for HAD therefore does not only consist of a certain format, but it is a comprehensive software framework.

Since the applications that depend on a highly accurate map are still the object of research, we need a discourse on how such a framework should be structured and which functionalities it must support. For this the map framework *Lanelet2* should represent a starting point and support these developments through its expandability and flexibility.

We will make the map framework *Lanelet2* available under the BSD license on Github[1] and will continue to add increasingly more functionality. We hope for feedback and contributions.

Lanelet2-based maps for large parts of Karlsruhe, Germany, are currently in preparation and will also be published.

[1] `https://github.com/fzi-forschungszentrum-informatik/lanelet2`

ACKNOWLEDGEMENT

This publication was written in the framework of the Profilregion Mobilitätssysteme Karlsruhe, which is funded by the Ministry of Science, Research and the Arts and the Ministry of Economic Affairs, Labour and Housing in Baden-Württemberg and as a national High Performance Center by the Fraunhofer-Gesellschaft.

REFERENCES

[1] J. Ziegler, P. Bender, and M. Schreiber, et al., "Making bertha drive – an autonomous journey on a historic route," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 2, pp. 8–20, 2014.

[2] J. M. Wille, F. Saust, and M. Maurer, "Stadtpilot: Driving autonomously on braunschweig's inner ring road," in *2010 IEEE Intelligent Vehicles Symposium*, pp. 506–511, June 2010.

[3] Ö. Ş. Taş, N. O. Salscheider, and F. Poggenhans, et. al, "Making bertha cooperate – team annieway's entry to the 2016 grand cooperative driving challenge," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, pp. 1262–1276, April 2018.

[4] M. Aeberhard, S. Rauch, and M. Bahram, et al., "Experience, results and lessons learned from automated driving on germany's highways," *IEEE Intelligent Transportation Systems Magazine*, vol. 7, pp. 42–57, Spring 2015.

[5] E. Rehder and H. Kloeden, "Goal-directed pedestrian prediction," in *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, pp. 139–147, Dec 2015.

[6] "Here hd live map - on the road towards autonomous driving." `https://www.here.com/en/products-services/here-automotive-suite/highly-automated-driving/here-hd-live-map`, 2018. Accessed: 2018-08-20.

[7] "Tomtom hd map with roaddna." `automotive.tomtom.com/automotive-solutions/automated-driving/hd-map-roaddna`, 2018. Accessed: 2018-08-20.

[8] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, pp. 12–18, Oct 2008.

[9] P. Neis, D. Zielstra, and A. Zipf, "The street network evolution of crowdsourced maps: Openstreetmap in germany 20072011," *Future Internet*, vol. 4, no. 1, pp. 1–21, 2012.

[10] "Opendrive." `www.opendrive.org/`, 2018. Accessed: 2018-04-11.

[11] I. Shim, J. Choi, and S. Shin, et al., "An autonomous driving system for unknown environments using a unified map," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, pp. 1999–2013, Aug 2015.

[12] F. Dierkes, M. Raaijmakers, and M. T. Schmidt, et al., "Towards a multi-hypothesis road representation for automated driving," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pp. 2497–2504, Sept 2015.

[13] A. Schindler, *Vehicle Self-Localization Using High-Precision Digital Maps*. PhD thesis, University of Passau, Faculty of Computer Science and Mathematics, 2013.

[14] P. Bender, J. Ziegler, and C. Stiller, "Lanelets: Efficient map representation for autonomous driving," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pp. 420–425, June 2014.

[15] E. Héry, S. Masi, P. Xu, and P. Bonnifait, "Map-based curvilinear coordinates for autonomous vehicles," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1–7, Oct 2017.

[16] M. Quigley, K. Conley, and B. P. Gerkey, et al., "Ros: an open-source robot operating system," 2009.