2014 IEEE Intelligent Vehicles Symposium (IV)
June 8-11, 2014. Dearborn, Michigan, USA

Lanelets: Efficient Map Representation for Autonomous Driving

Philipp Bender, Julius Ziegler and Christoph Stiller*

***Abstract*— In this paper we propose a highly detailed map for the field of autonomous driving. We introduce the notion of *lanelets* to represent the drivable environment under both geometrical and topological aspects. Lanelets are atomic, interconnected drivable road segments which may carry additional data to describe the static environment. We describe the map specification, an example creation process as well as the access library `libLanelet` which is available for download. Based on the map, we briefly describe our behavioural layer (which we call behaviour generation) which is heavily exploiting the proposed map structure. Both contributions have been used throughout the autonomous journey of the MERCEDES BENZ S 500 INTELLIGENT DRIVE following the Bertha Benz Memorial Route in summer 2013.**

I. INTRODUCTION

In the last years, intelligent transportation systems drew the attention of governments and private companies and promising improvements in security, efficiency and comforts showed up. Recently, companies like Daimler or Google tried to make the technologies for autonomous driving available for the broader audience. These technologies include perception, planning and decision making systems. While all subsystems impose different requirements, the underlying model shows interesting similarities. The perception system tries to match the sensor input with this model. The planning and decision making systems deduce actions and plans from the sensor input in combination with the model. A good model eases the communication between the different subsystems, it is understandable and examinable by humans, and is based on reliable and available tools.

On public roads, the most obvious model is easy to identify: it is the map, the set of lanes, their interconnection as well as the according road traffic regulations. The map has to be complete in both topological and geometrical terms. A precise description of elements itself as well as their relation has to be supported. However, there exists no such unifying model up to date which suffices all mentioned requirements.

Herein, we propose *lanelets*, a novel concept for map representation, to fill this gap. We model the relevant parts of the environments by means of *lanelet* elements, which are atomic, interconnected drivable road segments, geometrically represented by a left and right bound. The bound is approximated by a list of points, yielding a polygonal line or a *polyline*. We identified a set of elements to describe traffic regulations and attribute those elements to the lanelets.

The BERTHA BENZ project is a collaboration between Daimler AG and FZI Research Center for Information Technology.

*Philipp Bender, Julius Ziegler and Christoph Stiller are with FZI Research Center for Information Technology, Mobile Perception Systems, 76131 Karlsruhe, Germany {pbender,ziegler,stiller}@fzi.de

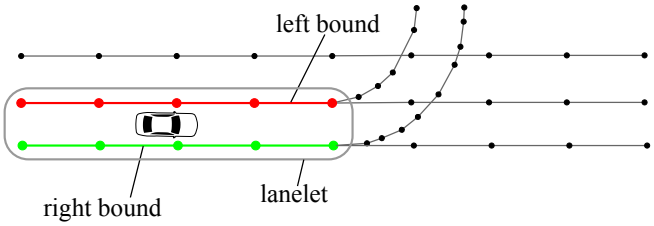

Fig. 1: Map composition by lanelets. A singe lanelet is highlighted. The roles of the bounds specify the driving direction, here from left to right.

The lanelets compose the road network with lanes, roads and intersections. The resulting map, which we call *lanelet map*, is then used to infer situations, to predict their evolutions as well as to find a route from the current position to the journey's destination.

Our lanelets were used to complete the Bertha-Benz-Memorial-Route (BBMR) to verify our approach. This journey in September 2013 was an ambitious project aiming to autonomously complete the route chosen for the very first long distance drive by Bertha Benz[1] 125 years ago. Figure 2 gives an overview over the route which passes the cities of Mannheim, Heidelberg and Pforzheim as well as 23 smaller villages. The route evinces a very high variability ranging from federal highways to small and narrow streets downtown. Additional challenges were 18 roundabouts, more than 200 traffic light driven intersections, unknown static and dynamic obstacles, speed limits and merges. The map proved to be reliable and complete, representing both topology and geometry in an accessible, human readable and modifiable way. Lanelets were an important cornerstone of the full system.

The remainder of this paper is laid out as follows: In Section II, we present some of the related work. In Section III, we introduce *lanelets*. A formal specification of lanelets and their properties is given in Section IV, followed by Section V, which briefly describes lanelets in the context of the BBMR and how our behaviour generation module profits from the lanelet map. We conclude this paper with Section VI.

II. RELATED WORK

Road representation is important in several domains of autonomous driving. Each domain has particular demands. We discriminate two aspects: the representation of geometries and the representation of topologies. However, most

[1]Bertha Benz was the wife of the Karlsruhe University alumnus Karl Benz, the inventor of the automobile.

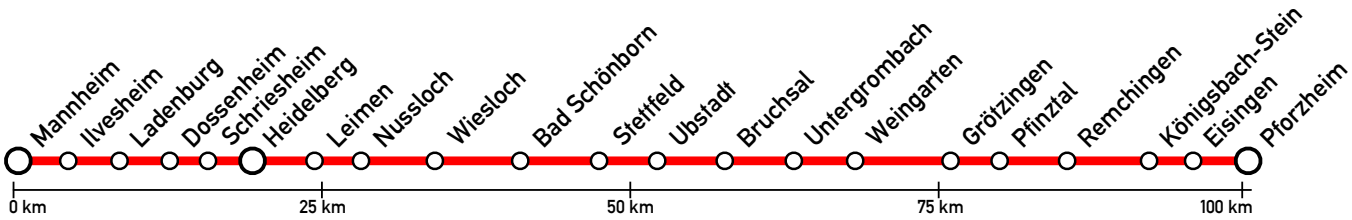

Fig. 2: Starting from Mannheim, the figure depicts the course of the BBMR with the passed cities as well as the approximate travel distance.

topological representations incorporate – at least coarse – geometrical information as well. We will start by first showing purely geometrical representations and then switch to mixed geometrical and topological representations.

A. Geometrical approaches

Wurm *et al.* represent the 3-D environment using *octrees* which allow for arbitrary resolutions where needed [1]. In their work, they present a probabilistic mapping approach as well as a memory efficient implementation. Due to the probabilistic occupancy representation, sensor measurements can be integrated seamlessly.

For the 2011 Grand Cooperative Driving Challenge, the winning team AnnieWay represented a lane as a polyline with a vertex distance of 0.5 m with vertices stored and accessed efficiently by a 2-D kd-tree [2].

Another source of inspiration is the domain of lane detection. Dickmanns *et al.* used clotoids to model road boundary geometry [3], which had a big impact on the topic of lane estimation. Wang *et al.* introduce the usage of Catmull-Rom splines to model road boundaries and lane markings [4]. The main reason was local controlability (moving a single control point only affects nearby points), smoothness and continuity, and the property that the spline will go through the control points. An approach with different models, a linear one for the near field, and a parabolic one for the far field, was presented by Jung *et al.* [5]. Baer *et al.* present an approach to model and track lane markings in a 4-D manner [6]. Other approaches utilize so-called *line-snakes* [7] or a pair of parabolas with constant curvature [8].

B. Mixed geometrical and topological approaches

For the 2007 Darpa Urban Challenge, the ROUTE NETWORK DEFINITION FILE (RNDF) format has been published [9]. Within this format, roads are split into segments comprising one or more *lanes*. The lanes are approximated by a set of waypoints as well as an optional width parameter. Some of the waypoints are labelled as *entry* or exit waypoints. Thereby, connections between lanes are represented. In addition to lanes, *zones* are the second important building block. Zones represent a freely driveable area approximated by a bounding polygon. It is possible to assign speed limits to road segments and zones. In general, the representation seems to be very complete when it comes to topological representation with some drawbacks when it comes to geometrical representation.

Betaille *et al.* propose *Emaps*, *extended maps*, which model the world in terms of interconnected clothoids, line and circle segments [10].

Hasberg and Hensel use global splines and a Bayesian approach to reconstruct and represent geometry and topology in the context of rail vehicle localization [11].

Contributions focusing on localization by means of an arc length relative to a given road segment content with coarser geometrical representations. However, they depend on reliable topological representations [12]. Brubaker *et al.* [13] use data stemming from OpenStreetMap (OSM). OSM [14] is a project collecting geographic data allover the world, depending on the work of volunteers and data put into the public domain by companies or governmental organizations. The core of OSM is a database which models the world with three primitives, namely *nodes*, *ways* and *relations*. Nodes represent geographic points, ways represent lists of nodes (hence representing polylines or polygons) and relations consist of any number of members. A member may be of any of the three types and has a specified *role*. Role names do not have to be unique. Similar to RNDF, OSM data is very good at representing topological information as well as positions, but has drawbacks in representing the geometry itself: the course of roads is represented reasonably well, but the local geometry is not since roads are only represented as lines. Other road properties, like the driving direction, the number of lanes et cetera are given as properties of the lines.

III. LANELETS

The main contribution of this work is the introduction of maps and traffic rules based on *lanelets* and their application to maneuver planning. A lanelet describes an atomic lane segment which is characterized by its *left* and *right bound*. The bounds themselves are polylines, and therefore allow for arbitrary precise approximation of *lane geometries*. The *role* of a bound (left or right) specifies the driving direction. See Figure 1 for an example for a simple lanelet.

A. A network of lanelets

To enable routing through the map, we build a graph of *adjacent* lanelets. Two lanelets A, B are called adjacent if the begin and end points are identical and the curvature of the polylines determined by the left and right borders is below a certain threshold. This indicates, that the transition is smooth. An example illustrating adjacency is shown in Figure 3. The graph is defined by the set of vertices, represented by the

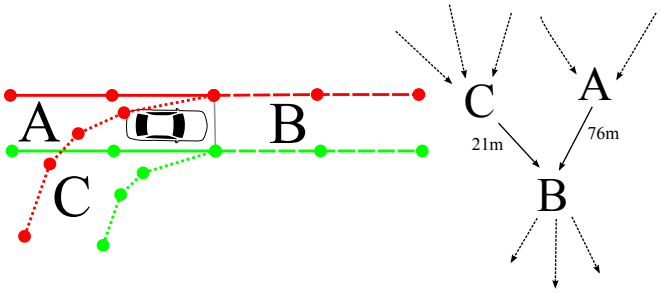

Fig. 3: Adjacency of lanelets: (A, B) and (C, B) are adjacent. On the left side, the lanelets are shown, the right side shows the obtained graph.

lanelets, and there are edges from each lanelet to the adjacent ones. By convention, we assign the length of a lanelet as weight to its outgoing edges which enables us to perform a routing on based on this graph using Dijkstra's algorithm [15]. By assigning properties to the lanelets such as road type or discount factors it is easy to implement other routing strategies.

B. Regulatory elements

A journey on public roads is guided by regulatory elements like traffic signs, rules at intersections and traffic lights. The general idea is to conceive those elements as own relations which are linked to the lanelets. The relations store all information needed to describe the situation. The vehicle must obey all the regulatory elements which are linked to the lanelets visited. In the following, we describe the important relation types, Section IV provides the missing details.

Regulatory elements provide two types of information. On one hand, they provide a *rule* or a *maneuver name*. On the other hand, they provide all the static information or parameters to obey this rule. At a traffic light for example the rule may be *stop at the line if one of the lights shows red*, and the parameters are the stop line and the positions of the relevant lights.

Another important regulatory elements are traffic rules at intersections without lights. Driving such situations has to be planned in a way that obstructs other traffic participants as little as possible if they have the right of way. Looking at the lanelet topology, we identified two types of such situations which suffice to describe intersections and roundabouts: the paths of two vehicles can either *cross* or *merge* (Figure 4). Each type requires additional information. Which are the lanelets crossed, where do the paths merge? Again, this information is given in the corresponding relation.

C. Composition of lanelets

The length of a lanelet may vary from very short, like a few meters, up to hundreds of meters or even a few kilometers. Therefore we concatenate lanelets to *driving corridors*. A driving corridor is a set of adjacent lanelets, representing a section of the path leading to a destination.

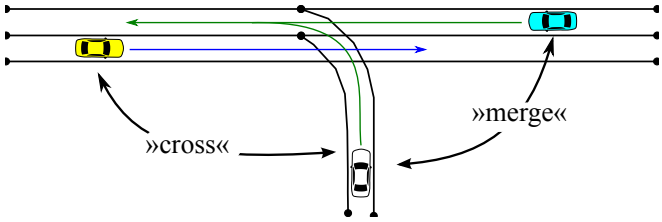

Fig. 4: *Merge* and *cross*: from the perspective of the white car, the yellow one crosses the path while the paths of white and cyan merge.

D. Calculation and measures

When working with lanelets, it is important to measure the distance of a pose to the bounds of a lanelet. Since the bounds are given as polylines, this could be the projection of the point on the polygonal line, with the gradient being perpendicular to the line segments. However, neither the distance nor the gradient is continuous in the surrounding of the support points. Therefore we show a method to obtain such a continuous distance metric, as well as the gradient. We call this distance measure the *pseudo distance*. First, we consider single segments. Such a segment is defined by the ensemble $G = (\overline{\mathbf{p}_b, \mathbf{p}_t}, \mathbf{t}_b, \mathbf{t}_t)$, with $\mathbf{p}_b, \mathbf{p}_t$ being the corner points (*base* and *tip*) of the line segment and the corresponding tangent vectors $\mathbf{t}_b, \mathbf{t}_t$. Now we interpolate the tangent vectors linearly along the line segment with a parameter $\lambda \in [0, 1]$. $\mathbf{t}_\lambda$ is the *pseudo tangent vector* in point $\mathbf{p}_\lambda$, and $\mathbf{n}_\lambda$ is the corresponding normal vector with

$$\mathbf{t}_\lambda = \lambda \mathbf{t}_t + (1-\lambda)\mathbf{t}_b \quad (1)$$

$$\mathbf{p}_\lambda = \lambda \mathbf{p}_t + (1-\lambda)\mathbf{p}_b \quad (2)$$

The *pseudo distance* of a point $\mathbf{x} = (x, y)^\top$ to the line segment is now defined as the length of the vector $\mathbf{n}_\lambda$ (orthogonal to $\mathbf{t}_\lambda$) connecting the base point $\mathbf{p}_\lambda$ and $\mathbf{x}$:

$$\mathbf{n}_\lambda = \mathbf{x} - \mathbf{p}_\lambda \quad (3)$$

under the orthogonality constraint

$$\mathbf{n}_\lambda^\top \mathbf{t}_\lambda = 0. \quad (4)$$

Please note that $||\mathbf{n}_\lambda||$ is not necessarily 1. The distance is illustrated in Figure 5. To obtain the value of λ, we transform our problem without loss of generality so that $\mathbf{p}_b$ lies in the origin $(0, 0)^\top$ and $\mathbf{p}_t$ lies on the x axis $(l, 0)^\top$. In this case, the tangent vectors can be represented only by their slopes $\mathbf{t}_b = (1, m_b)^\top$ or $\mathbf{t}_t = (1, m_t)^\top$. Now, Equation 4 can be solved for λ:

$$\lambda = \frac{x + y\, m_b}{l - y\,(m_t - m_b)} \quad (5)$$

The norm $||\mathbf{n}_\lambda||$ is the *pseudo distance* $d(\mathbf{x}, G)$ while $\frac{\mathbf{n}_\lambda}{||\mathbf{n}_\lambda||}$ is the *pseudo gradient* for the respective line segment. The y coordinate determines the sign of the distance measure,

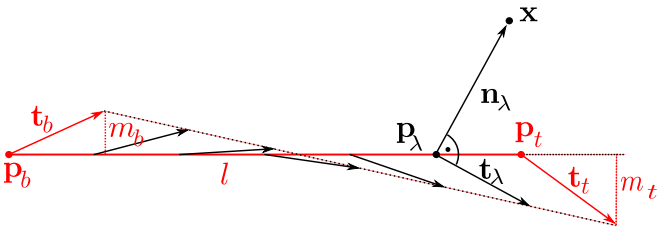

Fig. 5: Interpolation of normals along the segment with given tangent vectors at the terminal points.

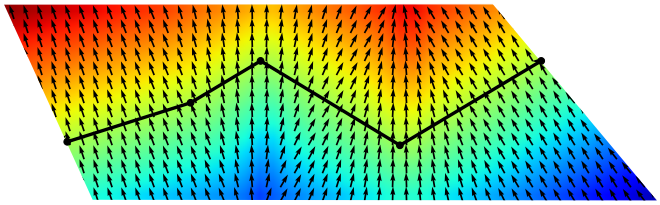

Fig. 6: Pseudo distances and gradients for the black line. The color map represents the (signed) distances (from negative distances, blue, to positive ones, red), the arrows show gradients.

that is, if $\mathbf{x}$ is on the left or right side of the line. This process needs to be repeated for each segment the polygonal line $\mathbf{p}_{1,\dots,N}$ consists of. In our case, the tangents $\mathbf{t}_{1,\dots,N}$ are not stored in the map, they are calculated as $\mathbf{t}_i = \mathbf{p}_{i+1} - \mathbf{p}_{i-1}$ for inner points, and $\mathbf{t}_1 = \mathbf{p}_2 - \mathbf{p}_1, \mathbf{t}_N = \mathbf{p}_N - \mathbf{p}_{N-1}$ for the corner points, respectively. The technique of interpolating normals this way is known as *Phong shading* [16]. Figure 6 depicts the signed distance measure for an example polyline. The trapezoidal shape of the plot comes from the fact that this distance function is only defined for points which lie on one of the pseudo normals of the line strip.

Once the value for λ is found for a given pose with position $\mathbf{x}$ and orientation φ, the pseudo tangent can be obtained. To calculate a distance to the implicitly given centerline (the line, where each point has the same distance to the left and right lanelet bound), the mean values of both distances and orientation differences are used.

IV. Specification of Lanelets

A. Storage format and editing tool chain

We represent lanelets and all other mentioned data structures resorting to the OSM formalism which uses the three primitives *nodes*, *ways* and *relations*. OSM defines an XML based file format for storage and exchange of map data as well as a server architecture to centrally store and validate map data. This enables parallel editing of the map as the server is able to deal with and merge so-called *change sets*. OSM saves positional data with a resolution of up to 10^{-7} deg which is about one centimeter at the equator. For comparison, the localization map Levinson *et al.* used for autonomous driving has about 5 cm precision [17].

One particular feature of the map format is the ability to use an arbitrary number of ways to determine each bound and to use the same way in any number of relations. This reduces the redundancy and enables the possibility to share bounds between lanelets.

JOSM, an OSM editor written in Java, drew our attention to create and edit the map. Figure 8 depicts a screenshot of the user interface. We developed our own style sheets for JOSM as well as a set of tools to validate the map structure.

Due to the utilization of OSM, all coordinates are given in the WGS84 coordinate frame.

In the remainder of this section, we will specify the elements used in the map. Nodes will be typeset in *italics*, ways will be typeset **bold** and relations will be underlined. When elements are expected to have certain *tags* or *properties*, they will be typeset `key=value`.

B. Modelling geometry and topology: Lanelets.

A lanelet is given as a OSM relation and consists of at least one **left** margin and one **right** margin. This means, way elements are given as members with the names left or right. If the roles are assigned multiple times, the ways are expected to be mergeable into one way which means, that by reversing and resorting the members can be made adjacent. It is possible to assign a speelimit with the `speedlimit=real number` tag. Zero or more regulatory elements, like traffic lights (see below), may be passed as regulatory_element members.

Bidirectional lanes need to be modelled as two lanelets, with the members **left** and **right** simply swapped. Shared bounds are represented by just one way element, which is present in all the neighbouring lanelets with the appropriate role. This keeps the redundancy low.

C. Obeying Rules: Regulatory elements.

The relations introduced here are characterized by the tag `type=regulatory_element`. They can be distinguished by the `maneuver` tag. We limit the description to elements which encode traffic rules at intersections. We skip information which depends on the implementation.

a) merge and cross: Both relations have a **stop_line** which tells the system where to stop if necessary. For the maneuver itself, a case discrimination is needed, as explained above: *merge*, identified by `maneuver=merge`, needs a ref lanelet, usually this is the first common lanelet of the merging paths. It is expected that with entering this lanelet, the velocity matches the velocities of the other vehicles as good as possible, and safety distances are maintained. If the type of the situation is *cross* (`maneuver=cross`), the relation needs at least one ref lanelet as well, but in this case in combination with the current driving corridor they define the area which is simply forbidden for the ego vehicle, if occupied. See Figure 7.

b) traffic light: Similar to give way, traffic light needs a **stop_line**. Additionally, for each relevant light, there is one *ref* node which encodes the position. OSM nodes are limited to latitude and longitude: additional data, like the height over ground, can only be given as key/value pairs. The attributes of those nodes can be exploited by a traffic light state recognition module (e. g. [18]). Traffic light maneuvers are identified by `maneuver=traffic_light`.

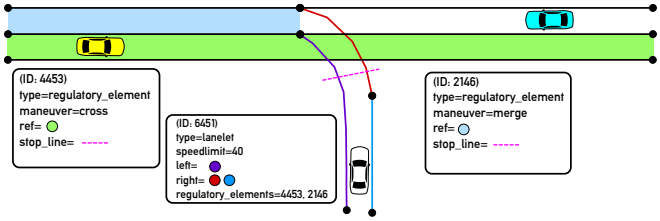

Fig. 7: Illustration of a lanelet as well as a *merge* and *cross* situation. The lanelets referenced in the regulatory elements are highlighted, while the lanelet 6451 is shown with colored bounds.

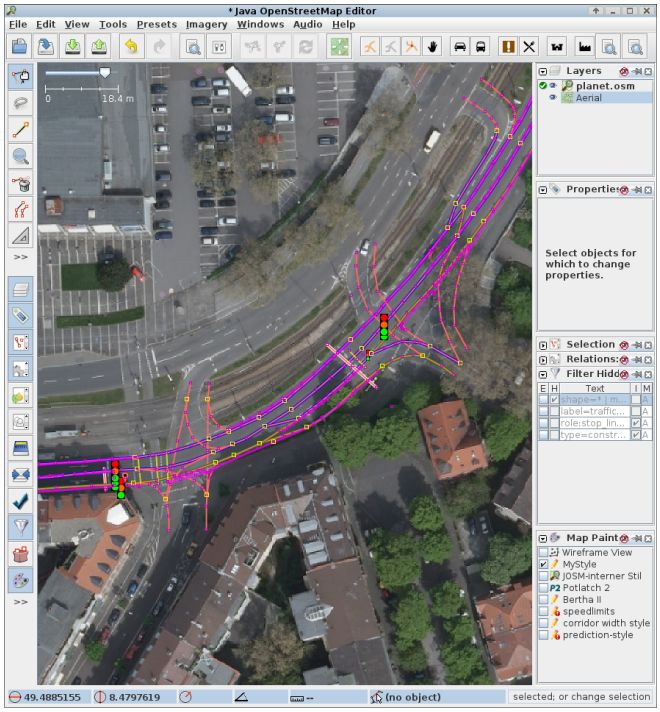

Fig. 8: Screenshot of JOSM, the Java OSM editor.

D. Efficient spatial access

To model the BBMR with all the intersections, oncoming lanes et cetera, we used more than four thousand lanelets. This number rapidly increases with the mapped area. To access this data, we propose the use of *R-Trees* [19] which enables the spatial search for objects inside an arbitrary bounding box in $\mathcal{O}(\log n)$.

E. Accessing the map: `libLanelet`.

With this publication, we also provide *libLanelet* [20], a library to read and parse the XML file and to profit from the fast spatial access as well as the routing routine, implemented as a Dijkstra search [15]. The library is written using the *Boost C++ libraries*, and unit tests have been written using *googletest*, the Google C++ testing framework. The library hides the construction of the lanelets, the R-tree and the routing graph behind a `LaneletMap` class. We provide a tiny map snippet, instructions and other supplementary material for use with *JOSM* and *libLanelet*.

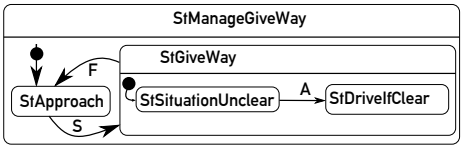

Fig. 9: An outtake of the state machine used.

V. Applying Lanelets: The Bertha-Benz-Memorial-Route

In this section, we describe how lanelet maps were successfully used in the context of the BBMR. During this project, many people contributed to the map or were dependent on the it. Many valuable findings found their way into the map creation process and design of the lanelet concept. In this section, we want to briefly outline the map creation and some example applications.

A. Map creation

To map the BBMR, we used virtual top view images as a foundation for manual annotation. With the help of stereo images, 3-D points from the vicinity of the vehicle are reconstructed and projected onto the ground plane. Those projections are superimposed by using the recorded trajectory of the vehicle [21]. The resulting images were displayed as background images in the editor. Examples are depicted in Figure 10. Based on those images, lanelets were annotated by hand and published on the central database.

B. Behaviour generation

The *behaviour generation* module in our system is upstream to trajectory planning [22] which is modelled as a variational optimization problem, where optimality is determined primarily in terms of driving dynamics and comfort. The behaviour generation sets up some of the constraints for this optimization problem, to make the trajectory planner obeying rules and to make it stay within bounds of the driving corridor. To set up those constraints, the behaviour generation uses the map in various ways and is internally organized as a *state chart* [23]. In the remainder of this section, we will show how the lanelet map is used for behaviour generation.

1) Driving the state machine: Figure 9 shows a detail of our state hierarchy. An important role of the map is to trigger transitions in the state machine. For instance, the event `S` (*start*) is triggered when a corresponding regulatory element is part of the current driving corridor, and the distance drops below a certain threshold. The vehicle is on alert and prepares to stop at the stop line. When the distance drops below a second threshold, event `A` (*approach*) is triggered and the system dynamically builds the constraints for this intersection (which is not part of this publication). After passing the intersection, event `F` (*finished*) resets the system to idle state.

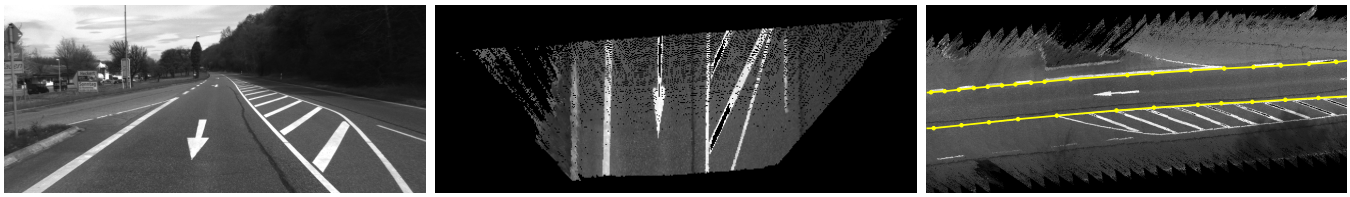

Fig. 10: Virtual topview: Single image, reconstructed top view, superposed top view with annotated lanelets.

2) Prediction of other vehicles: To accomplish this task, we first retrieve all the lanelets in the surrounding of the vehicle. Then, thresholds on distance and orientation difference to the centerline of a lanelet are used to reject unwanted lanelets and to get a measure for the probability of each hypothesis left. For each hypothesis, we find all the paths through the map graph up to a specific length which depends on the prediction horizon, e.g. 10 s, and the current speed of the vehicle. Then, for each of those driving corridors, we use a minimal planning scheme assuming that other traffic participants maintain a constant *distance* toward the right bound of their respective driving corridor and a constant *speed*. The resulting trajectories are used by the trajectory planner to avoid collisions [22].

VI. Conclusion

In this work, we proposed a specification for highly detailed maps for autonomous driving. As the most comprehensive experiment, the proposed behavioural map was successfully employed for automated driving on the 104 km long BBMR in 2013. The lanelets advocated in this contribution not only allow to compose complex road situations, but furthermore incorporate tactical information for maneuver generation, such as rule-compliant traversal of intersections with or without traffic lights. Our approach is extensible and the target of human readability has been met. Except `libLanelet` for parsing the XML files, our approach was fully supported by open source tools. Further extensions of the lanelet concept imply incremental refinement of the map over time, as well as completely automatic deduction of lanelet maps by exploiting trajectories from human drivers.

Acknowledgement

The authors thank Daimler AG for the fruitful collaboration and the opportunity to contribute to the 2013 Bertha Benz project.

References

[1] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: a probabilistic, flexible, and compact 3D map representation for robotic systems," in *Proc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation*, vol. 2, 2010.

[2] A. Geiger, M. Lauer, F. Moosmann, B. Ranft, H. Rapp, C. Stiller, and J. Ziegler, "Team AnnieWAY's entry to the 2011 grand cooperative driving challenge," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 3, pp. 1008–1017, 2012.

[3] E. D. Dickmanns and B. D. Mysliwetz, "Recursive 3-d road and relative ego-state recognition," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 14, no. 2, p. 199213, 1992.

[4] Y. Wang, D. Shen, and E. K. Teoh, "Lane detection using spline model," *Pattern Recognition Letters*, vol. 21, no. 8, pp. 677–689, Jul. 2000.

[5] C. Jung and C. Kelber, "A robust linear-parabolic model for lane following," in *17th Brazilian Symposium on Computer Graphics and Image Processing, 2004. Proceedings*, 2004, pp. 72–79.

[6] M. Baer, U. Hofmann, and S. Gies, "Multiple track 4D-road representation," in *2010 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2010, pp. 319–324.

[7] D. J. Kang, J. W. Choi, and I.-S. Kweon, "Finding and tracking road lanes using line-snakes," in *, Proceedings of the 1996 IEEE Intelligent Vehicles Symposium, 1996*, 1996, pp. 189–194.

[8] Y. Wang, N. Dahnoun, and A. Achim, "A novel system for robust lane detection and tracking," *Signal Processing*, vol. 92, no. 2, pp. 319–334, Feb. 2012.

[9] Darpa, "Urban challenge route network definition file (RNDF) and mission data file (MDF) formats," Mar. 2007. [Online]. Available: http://archive.darpa.mil/grandchallenge/docs/RNDF_MDF_Formats_031407.pdf

[10] D. Betaille and R. Toledo-Moreo, "Creating enhanced maps for lane-level vehicle navigation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 4, pp. 786–798, Dec. 2010.

[11] C. Hasberg, S. Hensel, and C. Stiller, "Simultaneous localization and mapping for path-constrained motion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 541–552, 2012.

[12] M. El Badaoui El Najjar and P. Bonnifait, "Road selection using multicriteria fusion for the road-matching problem," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 2, pp. 279–291, 2007.

[13] M. Brubaker, A. Geiger, and R. Urtasun, "Lost! leveraging the crowd for probabilistic visual self-localization," in *2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[14] F. Ramm, J. Topf, and S. Chilton, *OpenStreetMap*. UIT Cambridge, 2007.

[15] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, p. 269271, 1959. [Online]. Available: http://www.springerlink.com/index/uu8608u0u27k7256.pdf

[16] B. T. Phong, "Illumination for computer generated pictures," *Communications of the ACM*, vol. 18, no. 6, p. 311317, 1975.

[17] J. Levinson, M. Montemerlo, and S. Thrun, "Map-based precision vehicle localization in urban environments." in *Robotics: Science and Systems*, 2007.

[18] U. Franke, D. Pfeiffer, C. Rabe, C. Knoeppel, M. Enzweiler, F. Stein, and R. G. Herrtwich, "Making bertha see," in *IEEE ICCV Workshop Computer Vision for Autounomous Vehicles*, 2013.

[19] A. Guttman, "R-trees: a dynamic index structure for spatial searching," *SIGMOD Rec.*, vol. 14, no. 2, p. 4757, Jun. 1984.

[20] P. Bender, "libLanelet," 2014. [Online]. Available: http://www.mrt.kit.edu/software/

[21] M. Schreiber, C. Knoppel, and U. Franke, "LaneLoc: lane marking based localization using highly accurate maps," in *2013 IEEE Intelligent Vehicles Symposium (IV)*, 2013, pp. 449–454.

[22] J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for bertha - a local continuous method," in *submitted to IEEE Intelligent Vehicles Symposium (IV)*, 2014.

[23] D. Harel, "Statecharts: a visual formalism for complex systems," *Science of Computer Programming*, vol. 8, no. 3, pp. 231–274, Jun. 1987.