Automatic Conversion of Road Networks from OpenDRIVE to Lanelets

Matthias Althoff
Department of Informatics
Technical University of Munich
Munich, Germany
althoff@tum.de

Stefan Urban
Department of Electrical and Computer Engineering
Technical University of Munich
Munich, Germany
stefan.urban@tum.de

Markus Koschi
Department of Informatics
Technical University of Munich
Munich, Germany
markus.koschi@tum.de

***Abstract*—Detailed road maps are an important building block for autonomous driving. They accelerate creating a semantic environment model within the vehicle and serve as a backup solution when sensors are occluded or otherwise impaired. Due to the required detail of maps for autonomous driving and virtual test drives, creating such maps is quite labor-intensive. While some detailed maps for fairly large regions already exist, they are often in different formats and thus cannot be exchanged between companies and research institutions. To address this problem, we present the first publicly available converter from the OpenDRIVE format to lanelets—both representations are among the most popular map formats. We demonstrate the capabilities of the converter by using publicly available maps.**

I. Introduction

While it is the dream of many developers of autonomous systems that a vehicle fully understands its environment from on-board sensors only, it is obvious that maps drastically improve and accelerate building a semantic map of the environment during the operation of the vehicle [1], [2]. Additionally, maps serve as a backup solution if sensors fail and if parts of the road are occluded. For this reason, many larger companies and startups are investing in creating detailed maps for autonomous driving. Besides using maps on-board, maps are also critical for virtual test drives to reduce the cost of testing autonomous vehicles or advanced driver assistance systems; see e.g., [3]–[5].

Maps for navigation already exist and are open-source in some cases, such as OpenStreetMap [6], but creating detailed maps for autonomous driving is costly and reducing the costs of map creation is an ongoing research problem. Although one can automatically create maps using simultaneous localization and mapping (SLAM), those maps are not yet as detailed as the manually created ones [7]. A further way to reduce costs is to convert existing maps into the required format. In this work we present the first openly accessible converter from OpenDRIVE to lanelets. Both representations are among the most popular map formats, where OpenDRIVE [8] is more used in industry and lanelets [9] are currently more used in academia.

We first describe the main advantages of OpenDRIVE followed by addressing the benefits of using lanelets. Several tools support OpenDRIVE, and there are several tools that support creating maps in an OpenDRIVE format, such as the Trian3D Builder[1]. One of the main benefits of OpenDRIVE is the assurance of seamless exchange of models between different simulators. Other works, such as [10], have extended the OpenDRIVE format with further semantic information and partially automate the creation of meaningful maps.

Lanelets are becoming increasingly popular since their definition is more lightweight compared to e.g., OpenDRIVE, yet powerful enough to fulfill all major needs in driving simulators and automated driving. For instance, the autonomous drive along the Bertha Benz memorial route has been conducted using lanelets [11]. Lanelets are used for many aspects of driving simulators and automated driving, such as the composable benchmarks for motion planning on roads (CommonRoad) [12], lane-level map-matching [13], deep learning [14], formalization of traffic rules [15], set-based prediction of traffic participants [16], classification of driver intentions [17], and determination of location compliance [18], among others.

Besides lanelets and OpenDRIVE, other road description formats have been developed. RoadXML [19] is conceptually close to OpenDRIVE and consists of topological, logical, physical, and visual layers. There exist further open road network formats like LandXML [20] and OpenStreetMap [6], but they are designed primarily for geographical purposes and not for driving simulators or automated driving.

To the best of our knowledge, we present the first openly accessible converter of maps suitable for autonomous driving. Our converter is available for download from `commonroad.in.tum.de`. We believe that our converter is useful for many academic groups and people in industry, since maps often only exist in one format.

The paper is organized as follows: In Sec. II we present the principle for converting a road network described by OpenDRIVE to one described by lanelets. The concrete implementation is presented in Sec. III. Numerical examples demonstrating the quality of the conversion are shown in Sec. IV. The paper closes with final conclusions in Sec. V.

We gratefully acknowledge partial financial support by the BMW Group within the CAR@TUM project.

[1]www.triangraphics.de

II. Converting from OpenDRIVE to Lanelets

This section concisely presents the conversion of maps from OpenDRIVE to lanelets. To this end, we briefly introduce both formats and outline the relevant differences between them, followed by a detailed description of the conversion principle.

A. OpenDRIVE Format

In OpenDRIVE, roads are specified based on a reference path. Individual lanes are created by specifying a lateral distance from a reference path as visualized in Fig. 2. Reference paths are constructed by concatenating clothoids (aka Euler spirals) or polynomials. Please note that arc segments and straight lines are special cases of clothoids. The advantage of using clothoids is that the curvature along a reference path changes linearly with the path length, which is why most roads are constructed by clothoids [21]. Fig. 1 shows an example of a reference path, which represents a transition from a straight road into a bend. We call the to-be-concatenated elements *partial reference paths.*

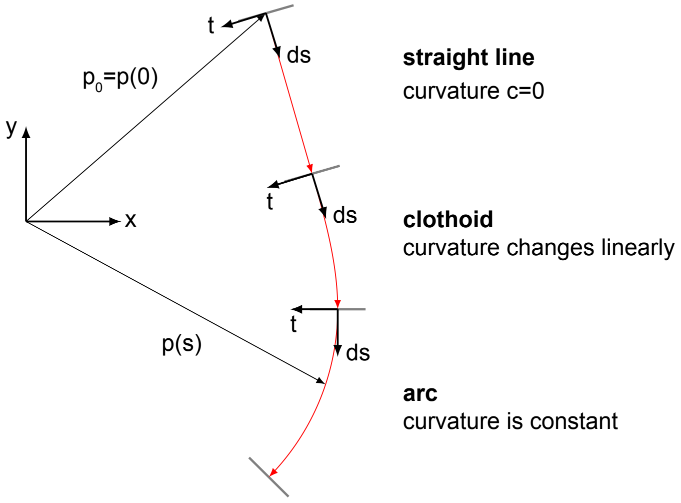


Fig. 1. Example of a reference path describing the transition from a straight road into a bend.

Please note that only the starting point p_0 is absolutely positioned on the map as shown in Fig. 1. Each segment of the reference path has a local coordinate system with axes t and ds as shown in Fig. 1.

The reference path is divided into multiple sections, which are chosen independently of the beginning and end of partial reference paths, i.e., the fully composed reference path is divided anew. Each section has a constant number of lanes, but properties such as the width can change within a section (see Fig. 2). Lanes are mainly specified by their type and width. The type clearly distinguishes between lanes on which a car can or may drive and other areas like sidewalks or parking spaces. The precise definition of the width of lanes with respect to the reference path is rather complex and can be found in the OpenDRIVE format specification[2]. Lanes in OpenDRIVE have no empty space between them; to introduce gaps, one has to create an additional lane of a special *non-road* type. Lanes with a negative lane number (ID) have the same direction as the reference path and positive IDs indicate that the direction is opposite as shown in Fig. 2.

[2]http://www.opendrive.org/docs/OpenDRIVEFormatSpecRev1.4H.pdf

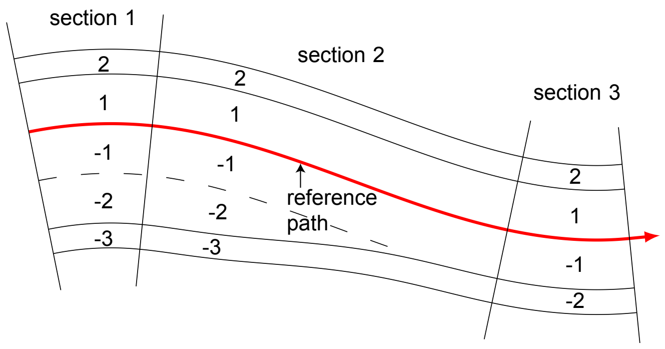


Fig. 2. Example of lane description along a reference path (red) in OpenDRIVE. The example contains three sections, each having a different number of lanes. One can see that the outer driving lane (left -2) merges into neighboring lane (-1). In the last section the merged lane completely vanishes and the IDs are re-adjusted.

Partial reference paths can have none, one, or multiple successors, where the latter is used to model junctions. At junctions with a high density of linked roads, a separate format is available. Multiple junctions can be grouped into a junction group. Connections to lanes within a road section are referred to as neighbors and determine to which lanes one can change.

It should be noted that the variety of possibilities to describe a real-world road with OpenDRIVE can be challenging, e.g., when lanes merge into other lanes as addressed in Sec. II-D. Next, we introduce lanelets, which do not use any reference paths and have a more lightweight representation.

B. Lanelet Format

Lanelets are atomic, interconnected, and drivable road segments [9]. A lanelet is defined by its left and right bound, where each bound is represented by an array of points (a polyline), as shown in Fig. 3. We define *start points* and *end points* of a lanelet as the first and the final points of the left and right border in driving direction, as shown in Fig. 3. The connection of lanelets to form a road network is defined implicitly: Two lanelets are called *longitudinally adjacent*, if the left and right start points of one lanelet are identical with the corresponding final points of the next lanelet in driving direction. We say that lanelet_2 is *left-adjacent* to lanelet_1 if the points of the left border of lanelet_1 are identical to the ones of the right border of lanelet_2. This is analogously defined for *right-adjacent* lanes. For implementation reasons, one might accept small deviations of connection points of lanelets rather than demanding that the values are identical.

The longitudinal, left, right, and empty adjacencies form a road network that can be modeled as a directed graph. It is a good practice to choose laterally adjacent lanelets such

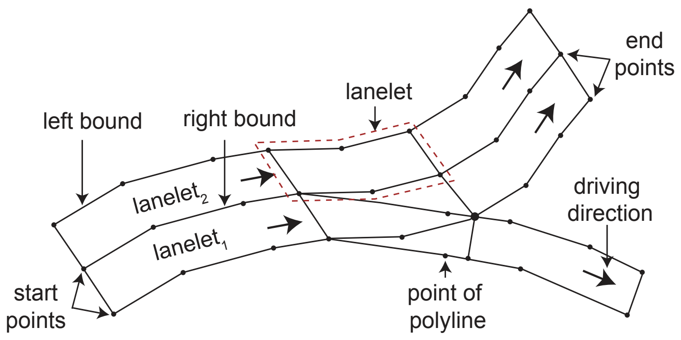
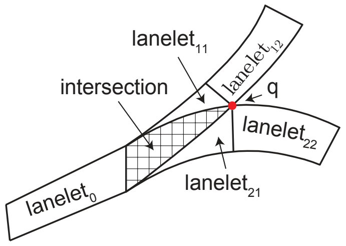
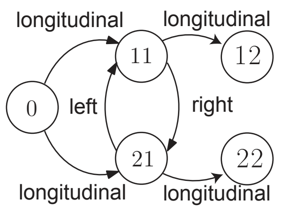


Fig. 3. Lanelets.

that their common bound has equal length, which can be done without loss of generality. This practice reduces the number of lateral adjacencies for multi-lane roads. Similarly, for road forks, it is a good practice to construct them as shown in Fig. 3 to ensure that adjacencies hold for the entire lanelet. This is realized by considering possible lane changes as long as there exists an intersection of lanes as shown in Fig. 4(a). Therefore, we introduce the point q as the intersection of the corresponding lane bounds of the bifurcating lanes, see Fig. 4(a). If the final points of the outer bounds of $\mathrm{lanelet}_{11}$ and $\mathrm{lanelet}_{21}$ correspond with the point q, and $\mathrm{lanelet}_{21}$ and $\mathrm{lanelet}_{22}$ continue the corresponding lanes as shown in Fig. 4(a), all lanelets are either adjacent along their full length or not at all. The resulting directed graph is presented in Fig. 4(b).



(a) Lanelets.



(b) Directed graph.

Fig. 4. Road fork modeled by lanelets.

C. Lane Bounds of OpenDRIVE Roads

Since lanelets are simply defined by their left and right bound, a major task for the conversion is to compute polylines of lane bounds of OpenDRIVE roads as illustrated in Fig. 5. The bounds are computed for each section (using its local coordinate system) so that the length of the sections and the lanelets are identical. First, we compute points s_i along the reference path (gray circles in Fig. 5) whose partial paths consist of lines, arcs, clothoids, and polynomials. We use [22] and [23, Eq. (3)] to obtain the x-and y- coordinates along the various partial path types. For each point s_i a corresponding inner point $I_i^{(j)}$ of the j^{th} lane and a corresponding outer point $O_i^{(j)}$ are computed. Those points are obtained by laterally shifting the points s_i by the lane widths $w_i^{(j)}$, where i refers to the i^{th} point s_i and j to the j^{th} lane. The outer points for one lane are identical to the inner point of the next lane.

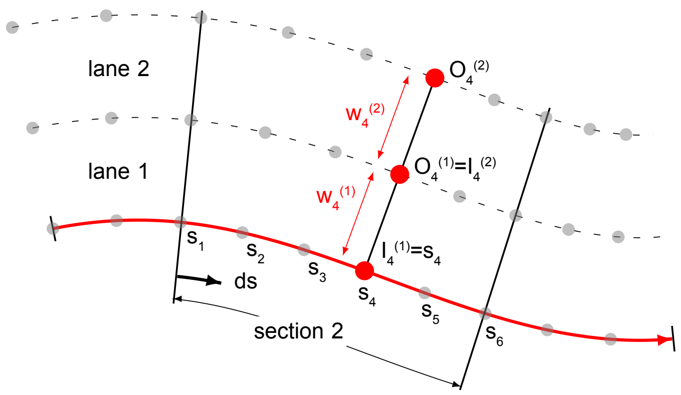


Fig. 5. Computation of lane border points $I_i^{(j)}$ and $O_i^{(j)}$.

Obviously, the obtained polylines are an approximation of the OpenDRIVE lane borders, but clothoids as used in OpenDRIVE do not have an analytical solution so that any driving simulator or any autonomous vehicle has to approximate clothoids by polylines or similar representations suitable for efficient computation. Also, the approximation error can be made arbitrarily small using small distances between the points s_i. For straight lines, however, the conversion is exact and only requires the start and end points. For arcs with curvature c we use the formula in (1) to determine the step size $ds_{\max}$ when the approximation error should be less than $e_{\max}$, which follows from fundamental geometry of arc segments.

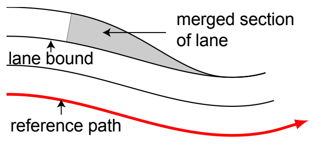
$$ds_{\max}\left(c, e_{\max}\right) = \frac{2}{c}\arccos\left(1 - c\, e_{\max}\right). \qquad (1)$$

Please note that we use the curvature c of the innermost lane bound with the highest curvature and not the one of the reference path. For clothoids we use the largest curvature at the beginning or end since their curvature changes linearly so that the maximum value is to be found at the beginning or end.

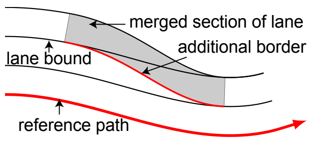
D. Lane Merges and Splits

A major difference in the road network description that remains to be addressed is the merging and splitting of lanes. In OpenDRIVE, lanes are merged by gradually reducing their width to zero or split by gradually increasing the width from zero. Even if the width is zero and the lane practically disappeared, the same lane identifier is reused in another section, as one can see in Fig. 2 at the end of lane -2 in section 2.

In a lanelet network, the end points have to coincide with starting points of another lanelet so that splitting and merging is realized as presented in Fig. 6(b). As a consequence, a lanelet realizing the merging or splitting of a lane overlaps with the lanelets of the neighboring lane.

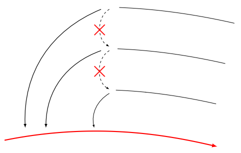


(a) OpenDRIVE.

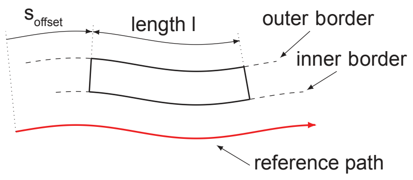


(b) Lanelets.

Fig. 6. Schematic representation of a merging lane using OpenDRIVE and lanelets.



(a) Dependencies of the borders to the inner neighbors are cut.



(b) Parametric lanes are specified by an offset s_{offset}, a length l, and an inner and outer border.

Fig. 7. Parametric lanes are specified with respect to the reference path so that the dependencies between lanes are removed.

To conveniently perform the conversion of merging and splitting, we have developed the concept of *parametric lanes*. These eliminate the dependency of each lane on its inner neighbor as indicated in Fig. 7(a) by specifying the borders with respect to the reference path. The following parameters visualized in Fig. 7(b) are required for parametric lanes:

- offset s_{offset}: Distance from the beginning of the reference path of the considered section.
- length l: Path length of the parametric lane.
- *borders*: Inner and outer borders specified as distances to the reference path varying along the reference path.

Thus, parametric lanes can model a lane as depicted in Fig. 6(b). Afterwards, the conversion of a parametric lane to a lanelet is done analogously to regular lanes.

III. Implementation

The overall implementation of our OpenDRIVE to lanelet converter is presented in Alg. 1. First, in line 1 we go through all roads modeled and obtain the reference path in line 2 based on the OpenDRIVE plan view and an additional lane offset. For each section of the road we generate lanelets in line 4 by calling Alg. 2, which is explained later. After creating all lanelets, we create a directed graph to represent the relationship between them. This step is not required since the road network of lanelets is defined implicitly, but it is convenient for other algorithms to already have the directed graph available. The connections are established for predecessors, successors, left neighbors, and right neighbors, in lines 6-9.

The creation of lanelets within one road section is shown in Alg. 2. We first convert all lanes into parametric lanes, which are converted in a second step into lanelets. Although this is only required for merging and splitting, using a unique pipeline simplifies our code and makes subsequent code updates easier since parametric lanes have no dependencies with other lanes. In line 1 the points of the reference path are assigned to the most inner border, where the step size is chosen according to (1). The points of the next borders are obtained from the points of the previous one by shifting them by the lane width as done in line 3 according to Sec. II-C. Since the next lane shares a border with the inner one, the next inner border equals the current outer border (see line 5). Finally, in lines 7-12 the parametric lanes are converted into lanelets.

IV. Numerical Experiments

To demonstrate the performance of our converter, we are converting openly accessible examples from `www.opendrive.org/download` so that the results can be independently checked. All conversions have been performed on a Dual Core Intel 2.60 GHz processor with 12 GB memory. All computation times are presented in Tab. I.

TABLE I
COMPUTATIONAL TIME OF MAPS DOWNLOADABLE OR LINKED FROM WWW.OPENDRIVE.ORG/DOWNLOAD.

name	**length of lanelets [m]**	**computation time [s]**
CrossingComplex8Course	17620.56	1.87
Crossing8Course	9264.06	1.04
KA-Suedtangente-Vires	37425.85	21.00
Roundabout8Course	9571.42	0.88
CulDeSac	318.75	0.07
sample1.1	23491.14	5.61

First, in Fig. 8 we present an original OpenDRIVE scenario and the result of the conversion. One can clearly see the lanelets created, which explicitly show the splitting of a lane before the roundabout. It can also be seen that the roundabout has two lanes, between which one can perform lane changes. Arrows indicate the driving direction of each lane. Further results of conversions are presented in Fig. 9. In order to recognize the details of the converted maps, we only present small sections of the converted roads; however, the computation times in Tab. I are measured for the complete map.

V. Conclusions

We have presented the first openly accessible converter from OpenDRIVE to lanelets. While OpenDRIVE is popular among manufacturers and suppliers, lanelets are increasingly popular in academia since their format is more lightweight. The main difference between OpenDRIVE and lanelets is that OpenDRIVE requires a reference path and defines lanes laterally to it. Some aspects, like pedestrian islands, can be a little tedious to model in OpenDRIVE. Lanelets, on the other hand, are simply defined by a left and right polyline. To obtain the polylines, we sample clothoid curves such that a maximum error is not exceeded. It should be noted that clothoids cannot be used for direct computations since they have no analytical solution and would have to be converted to polylines or a similar formalism in any case. Our converter

Algorithm 1 convert_opendrive_to_lanelets($opendrive$)

```
 1: for $road$ in $opendrive$ do                                  ▷ Convert each section of each road into lanelets
 2:     $referencePath \leftarrow road$.planView + $road$.laneOffset    ▷ planView contains the reference path without offset
 3:     for $section$ in $road$ do
 4:         $lanelets$.add(convert_to_lanelets($section$, $referencePath$))
 5: for $lanelet$ in $lanelets$ do                                ▷ Create connectivity graph
 6:     $lanelet$.predecessor ← find_predecessor_for_lane($lanelet$.lane, opendrive)
 7:     $lanelet$.successor ← find_successor_for_lane($lanelet$.lane, opendrive)
 8:     $lanelet$.left_neighbor ← find_left_neighbor_for_lane($lanelet$.lane)
 9:     $lanelet$.right_neighbor ← find_right_neighbor_for_lane($lanelet$.lane)
10: return $lanelets$
```

Algorithm 2 convert_to_lanelets($section$, $referencePath$)

```
 1: $inner\_border \leftarrow reference\_path$
 2: for $lane$ in $section$ do                                    ▷ Iterate from inner to outer lanes
 3:     $outer\_border \leftarrow inner\_border$.add_distance(lane.width)
 4:     $parametric\_lanes$.add(new ParametricLane($inner\_border$, $outer\_border$, $lane$))
 5:     $inner\_border \leftarrow outer\_border$                     ▷ Outer border will be inner border of next lane
 6: transform_lane_merges($parametric\_lanes$)                    ▷ Convert parametric lanes to lanelets as in Fig. 6
 7: for all $parametric\_lanes$ do
 8:     for s = 0, $ds_{\max}$, $2\,ds_{\max}$, …, length do           ▷ $ds_{\max}$ is obtained according to (1)
 9:         $left\_vertices$(s) ← $parametric\_lanes$.left_border(s)
10:         $right\_vertices$(s) ← $parametric\_lanes$.right_border(s)
11:     $lanelets$.add(new Lanelet($left\_vertices$, $right\_vertices$))
12: return $lanelets$
```

works flawlessly on all tested scenarios and can be downloaded from `commonroad.in.tum.de`. Even for larger maps, the computation times are within a few seconds.

References

[1] S. Kammel, J. Ziegler, B. Pitzer, M. Werling, T. Gindele, D. Jagzent, J. Schröder, M. Thuy, M. Goebl, F. von Hundelshausen, O. Pink, C. Frese, and C. Stiller, "Team AnnieWAY's autonomous system for the DARPA Urban Challenge 2007," *Journal of Field Robotics*, vol. 25, pp. 615 – 639, 2008.

[2] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. N. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, M. Gittleman, S. Harbaugh, M. Hebert, T. M. Howard, S. Kolski, A. Kelly, M. Likhachev, M. McNaughton, N. Miller, K. Peterson, B. Pilnick, R. Rajkumar, P. Rybski, B. Salesky, Y.-W. Seo, S. Singh, J. Snider, A. Stentz, W. Whittaker, Z. Wolkowicki, J. Ziglar, H. Bae, T. Brown, D. Demitrish, B. Litkouhi, J. Nickolaou, V. Sadekar, W. Zhang, J. Struble, M. Taylor, M. Darms, and D. Ferguson, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of Field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[3] B. Kim, Y. Kashiba, S. Dai, and S. Shiraishi, "Testing autonomous vehicle software in the virtual prototyping environment," *IEEE Embedded Systems Letters*, vol. 9, no. 1, pp. 5–8, 2017.

[4] M. R. Zofka, S. Klemm, F. Kuhnt, T. Schamm, and J. M. Zöllner, "Testing and validating high level components for automated driving: Simulation framework for traffic scenarios," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2016, pp. 144–150.

[5] R. Math, A. Mahr, M. M. Moniri, and C. Müller, "OpenDS: A new open-source driving simulator for research," in *Proc. of Automotive meets Electronics*, 2013.

[6] M. Haklay and P. Weber, "OpenStreetMap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, 2008.

[7] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, "Towards fully autonomous driving: Systems and algorithms," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2011, pp. 163–168.

[8] M. Dupuis, M. Strobl, and H. Grezlikowski, "OpenDRIVE 2010 and beyond – status and future of the de facto standard for the description of road networks," in *Proc. of the Driving Simulation Conference Europe*, 2010, pp. 231–242.

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

[10] T. Haubrich, S. Seele, R. Herpers, M. E. Müller, and P. Becker, "A semantic road network model for traffic simulations in virtual environments: Generation and integration," in *Proc.of the 7th IEEE Workshop on Software Engineering and Architectures for Realtime Interactive Systems*, 2014, pp. 43–50.

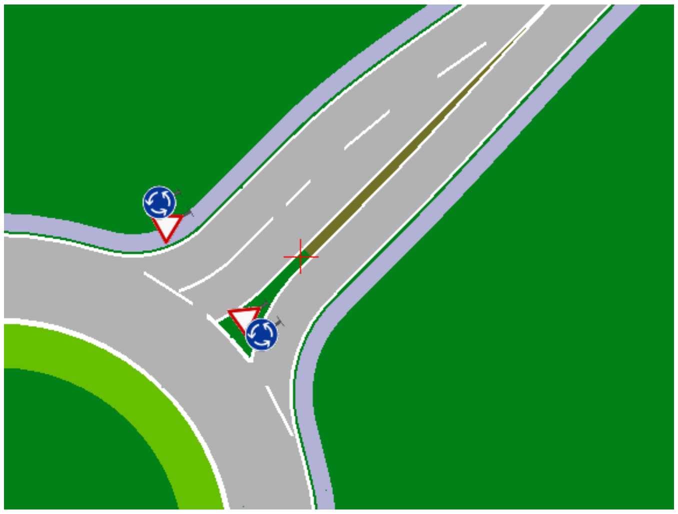
[11] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, T. Dang, U. Franke, N. Appenrodt, C. G. Keller, E. Kaus, R. G. Herrtwich, C. Rabe, D. Pfeiffer, F. Lindner, F. Stein, F. Erbs, M. Enzweiler, C. Knöppel, J. Hipp, M. Haueis, M. Trepte, C. Brenk, A. Tamke, M. Ghanaat, M. Braun, A. Joos, H. Fritz, H. Mock, M. Hein, and E. Zeeb, "Making Bertha drive – an autonomous journey on a historic route," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014.

[12] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 719–726.

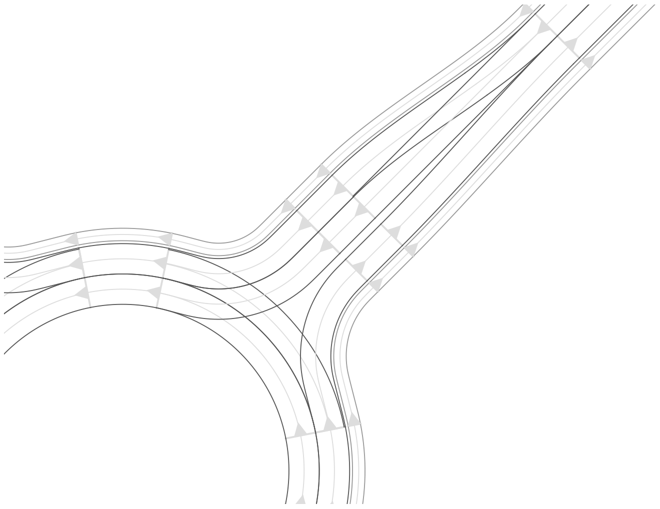
[13] J. Rabe, M. Meinke, M. Necker, and C. Stiller, "Lane-level map-matching based on optimization," in *Proc. of the 19th IEEE International Conference on Intelligent Transportation Systems*, 2016, pp. 1155–1160.

[14] P. Wolf, C. Hubschneider, M. Weber, A. Bauer, J. Härtl, F. Dürr, and J. M. Zöllner, "Learning how to drive in a real world simulation with deep Q-networks," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 244–250.

[15] A. Rizaldi, J. Keinholz, M. Huber, J. Feldle, F. Immler, M. Althoff, E. Hilgendorf, and T. Nipkow, "Formalising traffic rules for autonomous vehicles involving multiple lanes in Isabelle/HOL," in *Proc. of the 13th International Conference on integrated Formal Methods*, 2017, pp. 50–66.

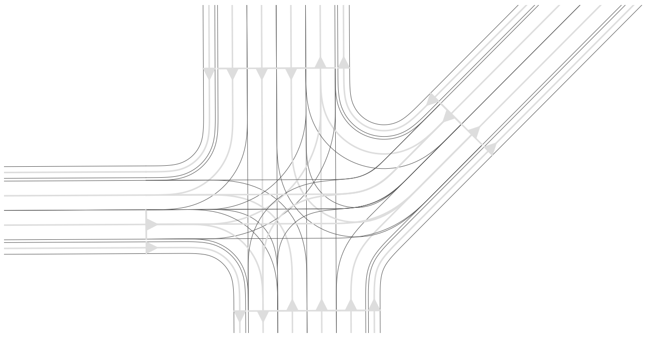


(a) OpenDRIVE example.

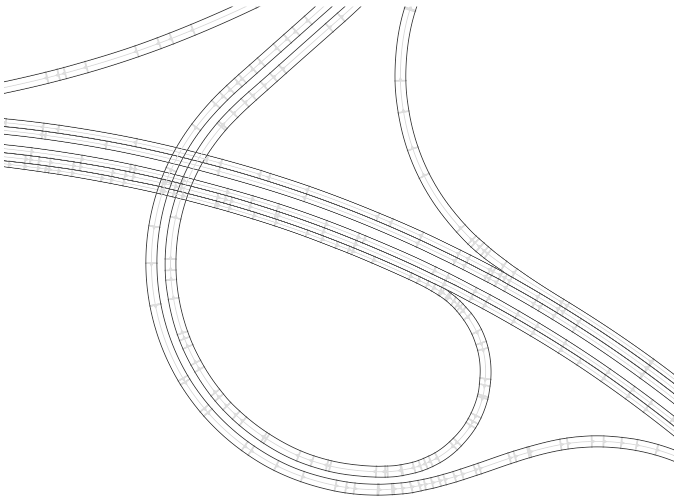


(b) Resulting lanelets.

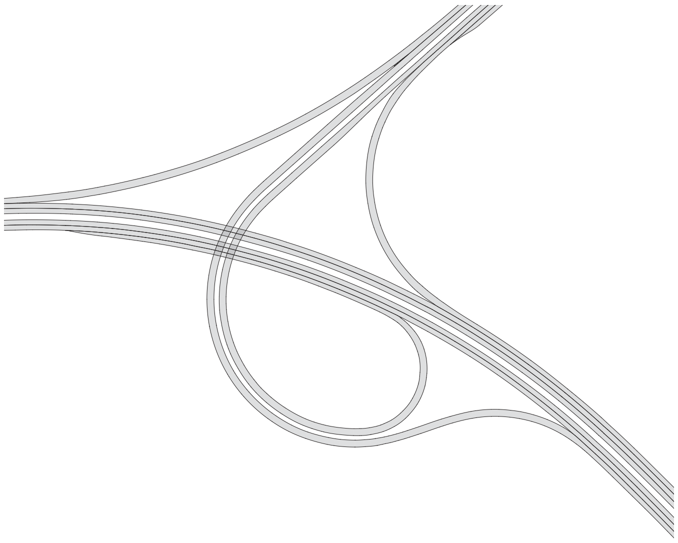
Fig. 8. OpenDRIVE example before and after the conversion to lanelets.



(a) Complex intersection with sidewalk.



(b) Detail of highway scenario in Fig. 9(c).



(c) Highway scenario.

Fig. 9. Examples of converted OpenDRIVE maps.

[16] M. Koschi and M. Althoff, “SPOT: A tool for set-based prediction of traffic participants,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2017, pp. 1679–1686.

[17] J. R. Ward, G. Agamennoni, S. Worrall, A. Bender, and E. Nebot, “Extending time to collision for probabilistic reasoning in general traffic scenarios,” *Transportation Research Part C: Emerging Technologies*, vol. 51, pp. 66 – 82, 2015.

[18] A. Zhu, S. Manzinger, and M. Althoff, “Evaluating location compliance approaches for automated road vehicles,” in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2018.

[19] J. Chaplier, T. N. That, M. Hewatt, and G. Gallée, “Toward a standard: RoadXML, the road network database format,” in *Proc. of Driving Simulation Conference Europe*, 2010, pp. 211–220.

[20] A. Hobson and A. Hobson, “SVG and LandXML for cadastral data on the web,” *Journal of Spatial Science*, vol. 50, no. 1, pp. 13–33, 2005.

[21] K. Baass, “The use of clothoid templates in highway design,” in *Transportation Forum*, vol. 1, 1984, pp. 47–52.

[22] J. Henrie and D. Wilde, “Planning continuous curvature paths using constructive polylines,” *Journal of Aerospace Computing, Information, and Communication*, vol. 4, no. 12, pp. 1143–1157, 2007.

[23] D. K. Wilde, “Computing clothoid segments for trajectory generation,” in *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 2440–2445.