

Anytime Lane-Level Intersection Estimation Based on Trajectories of Other Traffic Participants

Annika Meyer¹, Jonas Walter¹, Martin Lauer² and Christoph Stiller²

Abstract—Estimating and understanding the current scene is an inevitable capability of automated vehicles. Usually, maps are used as prior for interpreting sensor measurements in order to drive safely and comfortably. Only few approaches take into account that maps might be outdated and lead to wrong assumptions on the environment. This work estimates a lane-level intersection topology without any map prior by observing the trajectories of other traffic participants.

We are able to deliver both a coarse lane-level topology as well as the lane course inside and outside of the intersection using Markov chain Monte Carlo sampling. The model is neither limited to a number of lanes or arms nor to the topology of the intersection.

We present our results on an evaluation set of 1000 simulated intersections and achieve 99.9% accuracy on the topology estimation that takes only 36ms, when utilizing tracked object detections. The precise lane course on these intersections is estimated with an error of 15cm on average after 140ms. Our approach shows a similar level of precision on 14 real-world intersections with 18cm average deviation on simple intersections and 27cm for more complex scenarios. Here the estimation takes only 113ms in total.

I. INTRODUCTION

In recent autonomous driving systems, highly precise maps have been seen as a inevitable base for not only routing, but also for environment perception [1][2].

Accurate maps allow to replace difficult perception tasks, e.g. recognizing the road boundary, by simple and efficient map lookups. However, due to construction sites or traffic accidents, the road layout or at least the routing is prone to changes, which leads to outdated maps and a huge effort in updating those maps for ensuring safe and comfortable autonomous driving. In addition, maps rely on a similarly precise localization, which is still an active topic of research. Recent system architectures therefore did not only rely on map data, but also added a perception system that is able to estimate further cues on the current road layout [3]–[9]. The majority of these systems focused on lane detection either on simpler (e.g. almost straight) roads or highways [6][7][9]. They leveraged probabilistic approaches for reasoning over different straight or curved lane hypotheses and relied mainly on visual cues like markings and curb detections as borders. Similarly, in our previous work we applied a deep learning approach to the problem of lane detection in images [8].

¹Annika Meyer and Jonas Walter are with FZI Research Center for Information Technology, Karlsruhe, Germany, ameyer@fzi.de

²Annika Meyer, Martin Lauer and Christoph Stiller are with the Institute of Measurement and Control Systems, Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany

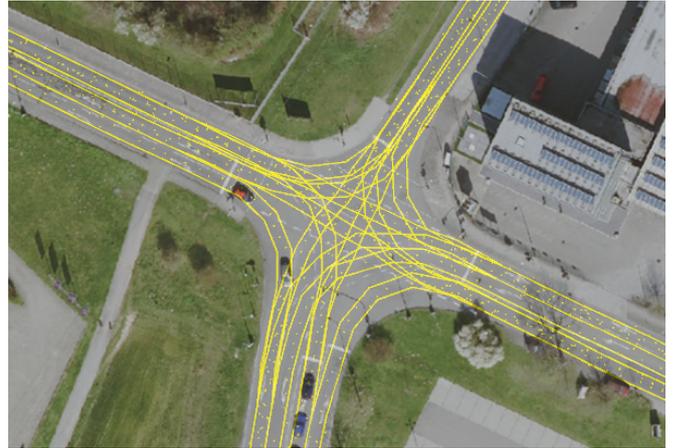


Fig. 1: Intersection in Karlsruhe as an example of urban intersections with our estimation. Aerial image source: City of Karlsruhe, www.karlsruhe.de, dl-de/by-2-0

For estimating intersections, curbs and markings are not sufficient because the latter usually intersect with each other, leading to highly ambiguous hypotheses. Moreover, both features might be occluded in heavy traffic and approaches that utilize the image coordinate system are often imprecise for distant areas due to perspective distortion. This makes it especially hard to detect road areas in huge intersections solely based on camera information, e.g. if a lane is represented by just a single pixels.

Hence, the task of estimating intersections has only been researched in a coarse fashion. Currently, autonomous driving systems would fall back to driving with maps assuming their correctness. Previous approaches dealing with the estimation of intersections are presented in the following.

Beck [5] presented an approach that aims at ego lane estimation, but could be extended to intersection estimation. They applied a graph-based shortest path algorithm to find the ego lane in an intersection, that could be applied to all entries of the intersection in order to estimate the lane courses at large. However, they calculated the costs based on marking detections in the image and a coarse semantic labeling, which limits the approach to the image domain and its viewing angle.

Other approaches [10][11] determined, which incoming and outgoing lanes are connected without any further assumption on the geometry of the connection. Joshi and James [10] based this kind of estimation on trajectories from other vehicles detected with an onboard Lidar system. Likewise,

other approaches [11]–[13] used those trajectories for estimating intersections in offline scenarios. However, those approaches used fleet data and assumed a huge number of trajectory detections per lane. Ruhhammer et al. [11] applied clustering algorithms, which themselves require a multitude of trajectory data, whereas Chen and Krumm [12] calculated a center line for each lane by fitting Gaussian mixture models in order to create a map. Roeth et al. [13] also aimed at map generation, but limited themselves to map graphs instead of lane-level estimation. Their approach, however, might still be applicable to online lane-level estimation using an extended model. They applied a Markov chain Monte Carlo (MCMC) algorithm, to sample intersection models that are evaluated against the measurement data.

Similarly, Geiger et al. [4] estimated the intersection geometry using MCMC. Based on image cues like object detections and tracks, coarse semantic labels and vanishing points. They estimated the intersection structure with a short video sequence approaching the intersection, that took 1 s for estimation. The approach achieved promising results although the viewing angle in the dataset was limited to a single camera and lacked sensors facing to the sides. This might have been sufficient because of the low complexity of the intersections. They published the intersection ground truth for 113 video sequences, but the dataset also has the limitation of a narrow field of view and the lack of lane-level ground truth for intersections with more than a single lane per direction.

Furthermore, the intersection model assumes some limitations that are not consistent with the majority of urban intersections. At first, they limit the approach to 7 topologies ranging from straight roads to four-armed intersections with only a single lane per driving direction. Additionally, the crossing arms are forced to be collinear. Typically, big, urban real-world intersections rather look like that in Figure 1, which is an aerial image with our estimation of an intersection in Karlsruhe. We analyzed urban roads and highways around Karlsruhe and found that intersections have up to 5 arms and an average of 2.5 arms. Of these, at least 20% have more than 3 lanes in total. It should be noted that two structurally separate lanes are considered as two separate arms.

Nevertheless, later works used the dataset of Geiger et al. or their intersection model [14][15], but still could not overcome the drawbacks, that might not hinder simple intersections from being driven autonomously, but still leave more complex ones to be a problem.

Similar to some of the previously mentioned work [3][4][6][11][13], we base our approach on the detection and the tracking of other vehicles passing the intersection. Even uncommon intersection structures and spontaneous deviations due to accidents are represented in the measurements. Additionally, the use of object detections increases the possible viewing range to more than a hundred meters, when working with lidar or radar detections [16].

Like others before [4][13][14], we use MCMC because it is able to work with contradicting hypotheses in a complex

model and can deliver results while approaching an intersection. Using a probabilistic sampling approach like MCMC, we are able to incorporate measurements from different sensors with different measurement uncertainties. In comparison to all mentioned previous approaches, this work can be applied in an incremental fashion, where the estimation is updated with every measurement and can provide an up-to-date estimate of the intersection at any time in a world-fixed coordinate system (like a map). Our model also goes a lot further compared to earlier approaches by not limiting the number of arms or lanes and by providing the exact lane course both within and without the intersection.

In summary, our contributions are the following:

- Estimation of intersection topology, geometry and precise lane course by observing other traffic participants
- Progressive incorporation of new measurements generating results at any time
- Use of an unrestricted model in terms of e.g. number or collinearity of arms

II. PROBABILISTIC GENERATIVE MODELS

For our approach, we rely on the trajectories of other traffic participants as measurements, to estimate the topology and lane course. As depicted in Figure 2, we are able to use measurements that come from either lidar, camera or radar. For a detailed estimation of the intersection (including the lane course) a tracking system has to be executed beforehand.

In order to get the best intersection estimation I based on the detections Z , we need to calculate the posterior probability $P(I|Z)$. The model for such an intersection becomes quite complex when regarding each lane and its individual course across the intersection. Because the posteriors for those models have no simple analytical solution, we use a sampling approach for estimating the intersection. MCMC allows for sampling from such high-dimensional spaces and is especially useful when there is no analytical solution to the problem.

In the manner of MCMC we sample intersections, calculate their posterior probability according to the measurements and decide, whether we would like to retain the solution (or not). During the process we apply simulated annealing [17] in order to converge towards the best intersection model. To decide, whether we want retain a new solution, we calculate only the relation between the posterior probabilities of the two intersections I_a and I_b according to the metropolis algorithm [18]. By applying Bayes' rule, we can simplify our calculations as in (1) by removing the factor $P(Z)$.

$$\begin{aligned} \frac{P(I_a|Z)}{P(I_b|Z)} &= \frac{P(Z|I_a) \cdot P(I_a) \cdot P(Z)}{P(Z|I_b) \cdot P(I_b) \cdot P(Z)} \\ &= \frac{P(Z|I_a) \cdot P(I_a)}{P(Z|I_b) \cdot P(I_b)} \end{aligned} \quad (1)$$

Assuming that our measurements z_i are independent of each other, we reformulate (1) for each intersection I as

$$P(Z|I) \cdot P(I) = P(I) \cdot \prod_{i=1}^k P(z_i|I). \quad (2)$$

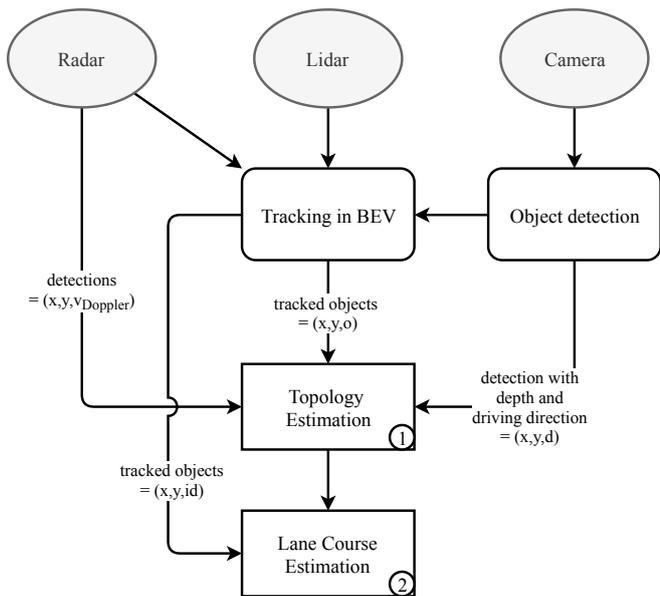


Fig. 2: Overview of the system. Either radar, lidar or camera serve as input. For estimating the lane-level topology ① only point detections and a coarse driving direction are necessary. In order to infer the lane course ②, a tracking system has to be added.

Thus, for our model, we need to calculate the prior $P(I)$ and likelihood $P(z_i|I)$ for every intersection. When having a deeper look at the measurements in our approach, it's obvious that object detections are in reality not statistically independent. They might belong to the same object and detections belonging to the same object follow the same movements. In order to correctly model the dependency of detections on the same vehicle, we would need to implement an extended object tracking algorithm to calculate the probabilities on the association. We therefore decided to analyze both options:

- i) average points of trajectories (= tracked objects), where the statistical independence can be assumed, and
- ii) single object detections without tracking information, willingly modeling the problem inaccurately.

III. INTERSECTION TOPOLOGY ESTIMATION

The approach is divided into two steps, which makes it possible to process a broad range of sensor measurements as depicted in Figure 2. In the first step, we estimate the lane-level intersection topology. Here, we assume the outgoing lanes to be straight as initial approximation for fast convergence. In the second step, this initial result is refined to the exact course of each lane both inside and outside of the intersection. For the latter, we need all the input data to be processed by a tracking system that calculates the temporal association of the detections.

A. Preprocessing

The first step requires point detections of other traffic participants (e.g. cars, trucks) that have at least a coarse information about their driving direction (see Figure 2).

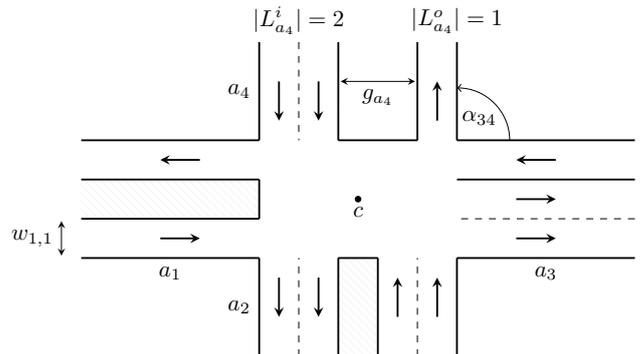


Fig. 3: The model is fixed at a center point and comprised of a set of arms with assigned lanes. Each lane is assumed to be straight for simplification and is represented by its angle.

We are able to use radar measurements, when represented as position $p = (x, y)$ and the Doppler velocity v_{Doppler} [16], since the Doppler velocity of radars can be transformed heuristically into the necessary classification of the driving direction d by

$$d = \begin{cases} v_{\text{Doppler}} > 0 & \text{leaving} \\ v_{\text{Doppler}} \leq 0 & \text{entering.} \end{cases} \quad (3)$$

This classification assumes, that the perceiving ego vehicle is driving on a lane of the intersection. For detections behind the ego vehicle the classification is vice-versa.

Using a camera system we would need to preprocess the images with an object detection system, that is able to provide the position of the object in a world-fixed coordinate system and a classification of the driving direction d as entering or leaving the intersection (e.g. [19]).

When dealing with lidar detections a tracking system is necessary in order to provide the driving direction of the objects (e.g. [20]). Here, we can reduce the input to detections on the vehicles $p = (x, y)$ and the driving direction as orientation vector \vec{o} derived from the velocity. Depending on the data type, a preprocessing for reducing noise and decreasing the number of detections might be useful. E.g. for radar measurements, filtering detections by their compensated Doppler velocity and radar cross-section is necessary in order to avoid static detections and clutter.

Additionally, a point reduction algorithm like voxelization is required, which combines local neighborhoods into a single detection. Because high resolution radars have a considerable high number of detections on a single vehicle, we can reduce computation times.

In case of tracked measurements, we speed up the calculations by preprocessing the associated detections. Trajectories passing the intersection are split into two by cutting the trajectory at the closest point to the center. Additionally, we reduce each trajectory to its mean point and mean driving direction, resulting in only one point per trajectory to process.

B. Intersection Model

The object detections Z are point positions $p = (x, y)$ in a world-fixed coordinate system and their orientation \vec{o} or

classification of the driving direction d .

In our approach, we model the intersection $I = (c, A)$ as depicted in Figure 3. The model is fixed at a center point c and comprised of a set of arms $a \in A$ with $a = (\alpha_{ab}, g_a, L_a)$. We also model structurally separated lanes, e.g. by a verge or guard rails, with a gap of width g_a between lanes of different driving directions. The angle α_{ab} specifies the angle between adjacent arms a and b . Each lane $l \in L_a$ can be classified with $d(l)$ as incoming or leaving lane. Each lane l also has a fixed width w_l assigned.

In our approach, we sample new intersections by modifying a single parameter and evaluating, whether this change should be accepted as valid sample or not. The detailed process for generating a new sample is enlisted in Algorithm 1. We randomly decide which parameter of the intersection is changed using predefined probabilities (sampled by ω). We either modify the number of arms or lanes, the angle of an arm, the center position or the medial strip. When we add an arm, we equally likely add this arm in the largest gap between two others or split one arm into two. For the lanes, we add that lane either on the medial strip or at the border of the arm. The same applies for removing a lane. For all add / remove steps, the general geometric bounds of the intersection are regarded (e.g. minimum number of lanes).

C. Probabilistic Evaluation

As described in Section II, each sampled intersection I is evaluated based on the conditional probability $P(I|Z)$ given the measurements Z . For each detection z_i , we need to calculate the likelihood $P(z_i|I)$ and for each intersection the prior $P(I)$.

1) *Intersection Prior*: The prior probability of the intersection $P(I)$ is based on the number of lanes and the number of arms. An invalid setting of the angles between two arms (e.g. overlapping arms) is already prevented by our sampling procedure (see Algorithm 1).

For the number of arms and lanes, we learned a distribution. The gap g_a , the angle between arms α_{ab} and the center c are modeled with non-informative priors. Thus, the intersection prior is calculated as

$$P(I) = P(|A|) \cdot P(|L|) \quad (4)$$

2) *Likelihoods*: In order to evaluate whether the measurements can be explained given the intersection model, we calculate $P(z_i|I)$. As discussed in Section III-A, we have positions of other vehicles $p = (x, y)$. Each position either has a classification of the moving direction d or an orientation \vec{o} , which leads to the triplet for each measurement $z_i = (x, y, d)$ or $z_i = (x, y, \vec{o})$.

Independent of measurement type, we determine the closest lane of the current intersection model with the same driving direction. Using $d_{\perp}(z_i, M_l)$, we describe the orthogonal distance between the point detection z_i and the center line M_l of that lane. Additionally, we take the angular deviation $d_{\triangleleft}(\vec{o}_i, M_l)$ between the center line M_l and the detected orientation \vec{o}_i , in case of tracked detections. We assume the following distributions

Algorithm 1 Sample new intersections by modifying a single parameter at a step.

```

 $\omega \leftarrow \mathcal{U}[0, 1]$ 
if  $\omega < 0.4$  then
    rotate a random arm by  $\Delta\alpha \leftarrow \mathcal{U}[-6^\circ, 6^\circ]$ 
else if  $\omega < 0.6$  then
    shift center by  $\{\Delta c, \phi\} \leftarrow \mathcal{U}([0 \text{ m}, 6 \text{ m}] \times [0, 2\pi])$ 
else if  $\omega < 0.7$  then
    change gap by  $\Delta g \leftarrow \mathcal{U}[-1.8 \text{ m}, 1.8 \text{ m}]$ 
else if  $\omega < 0.85$  then
     $\theta \leftarrow \mathcal{U}[0, 1]$ 
    if  $\theta < 0.5$  then add arm (details see text)
    else remove arm  $a \leftarrow \mathcal{U}_D(A)$ 
    end if
else
     $\theta \leftarrow \mathcal{U}[0, 1]$ 
    if  $\theta < 0.5$  then add lane (details see text)
    else remove lane  $l \leftarrow \mathcal{U}_D(L)$ 
    end if
end if

```

$$\begin{aligned} d_{\perp}(z_i, M_l) &\sim \mathcal{N}(0, \sigma_{\perp}) \\ d_{\triangleleft}(\vec{o}_i, M_l) &\sim \mathcal{N}(0, \sigma_{\triangleleft}). \end{aligned} \quad (5)$$

We define an association, using the driving direction of both the lane $d(l)$ and the detection $d(z_i)$, as

$$\mathbf{1}(z_i, l) = \begin{cases} 0 & d(z_i) = d(l) \\ 1 & d(z_i) \neq d(l) \end{cases} \quad (6)$$

determining whether z_i is assigned to lane l . When marginalizing over all lanes L in the model, the likelihood $P(z_i|I)$ becomes

$$P(z_i|I) = \sum_l^L \mathbf{1}(z_i, l) \cdot P_{\perp}(z_i|M_l, I) \cdot P_{\triangleleft}(\vec{o}_i|M_l, I). \quad (7)$$

For measurements without an orientation \vec{o}_i , we set $P_{\triangleleft}(\vec{o}_i|l, I) = 1$.

IV. LANE COURSE ESTIMATION

In the second step of this work, the input data needs to be enhanced with an association of object detections of the same vehicle. This means that every point detection gets associated with points from other time steps, yielding trajectories of objects. Using this information, we can connect estimated lanes at two different arms, in order to reconstruct a path that connects a pair of lanes on the intersection. As can be seen in Figure 2, different environment perceiving sensor types can be used for estimating the necessary trajectories when fed to a tracking algorithm.

This step refines the coarse model estimated in the topology estimation (see Section III) by estimating the lane course of the intersection, but it does not change the topology estimated before.

A. Preprocessing

First, the trajectory data has to be processed as depicted in Figure 4. We split each trajectory into its parts as described in Section III-A and depicted in Figure 4b. Then, our algorithm assigns each trajectory part to the closest arm of the estimated intersection model by searching for the lane whose center line is the closest to the mean point of the trajectory part (see Figure 4c). Each pair of trajectory parts can then be used to get possible connections between lanes (see Figure 4d).

Each lane course is represented as a lanelet [21], which is comprised of a pair of two poly lines, the left and the right border. Each lane in the topology model, estimated in the first step, is converted into a lanelet representation with equidistantly distributed support points (see Figure 5). For every lane that is connected with another arm by a trajectory, we also connect the lanelets in order to have a continuous lane model within the intersection as depicted in Figure 4d. In the preprocessing step, the connection is a straight lanelet, but will be refined in the estimation. Adjacent lanes can share all or a subset of the support points of the lanelet border, depending on whether they are parallel over the whole intersection or either of them turns into another arm.

For the generated lanelets we further calculate a center line for easier calculations later on. The center line M_l of each lanelet is defined as the poly line of center points m between opposing points on the lanelet borders $b \in B_l$. We also double the discretization rate of the center line by adding an additional support point between the existing ones.

We finally refine the representation using the trajectories. Each point of the center line is moved to minimize the distance to the assigned trajectories. As metric we calculate the orthogonal distance of the trajectory to the center line.

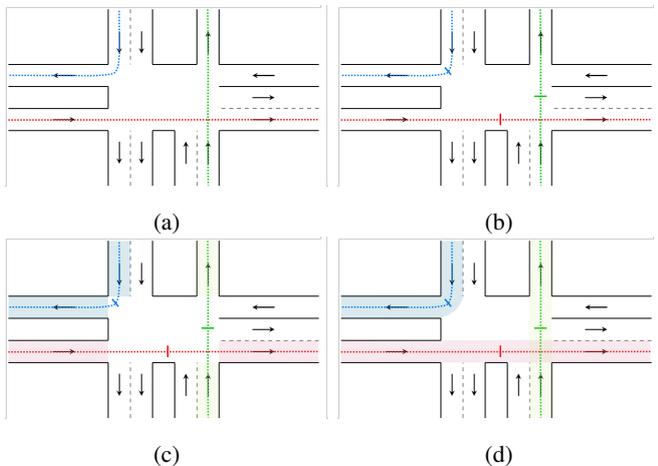


Fig. 4: Initialization of the lanes as lanelets using the topology estimation. (a) Input trajectories. Colors highlight different track ids. (b) Split trajectories into two parts. (c) Assign each trajectory part to a lane. (d) Connect lanes assigned to the same trajectory with a straight interpolation.

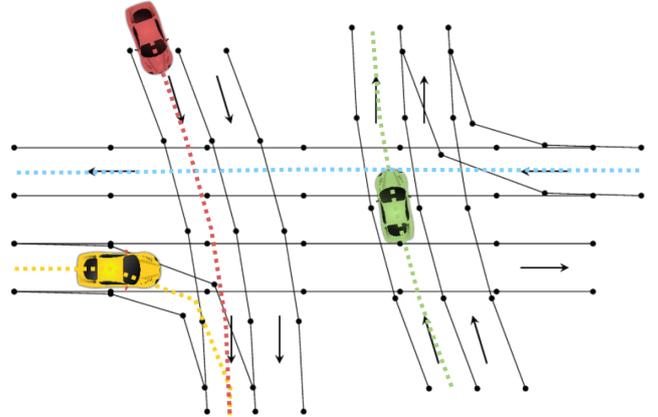


Fig. 5: For the lane course estimation, the intersection is represented as a set of lanelets with equally distributed support points.

With this refinement we finally calculate the neighbors of each border point that are candidates for fusing lane borders during the sampling.

B. Intersection Model

The tracked object detections T are a set of timestamped positions $p = (x, y)$ in a world-fixed coordinate system and an association that assigns multiple positions to a single trajectory t_i . Thus, we can additionally calculate the orientation \vec{o} for each position.

In this step, we model the intersection I_2 as set of lanelets L as depicted in Figure 5. Each lanelet l consists of optionally shared border points $b \in B_l$ and center points $m \in M_l$. With this representation, each sampling step either

- modifies a point m_l of the center line of a lane,
- splits two adjacent lanelets by generating a new border point for one of them, which is sampled around the original position with distance $\Delta b \sim \mathcal{U}[0 \text{ m}, 0.6 \text{ m}]$ or
- merges two neighboring lanelets l_a and l_b by assigning both lanelets the same border point. The new border point is equally likely to be the one taken from l_a or l_b .

C. Probabilistic Evaluation

For evaluating the resulting lane courses of I_2 we calculate the posterior probability $P(I_2|T)$ depending on the detected trajectories T . As described in Section III, this is split into calculating the prior on our model $P(I_2)$ and the likelihood for each trajectory point as $P(t_i|I_2)$.

1) *Lane Course Prior*: The prior is based on the number of shared border points between two adjacent lanes and the smoothness of the lanes $s(I_2)^\tau$. The smoothness term is applied because intersections are machine or man made and both have a tendency to stay continuous when designing intersections. Vehicles are bound to non-holonomic movements that prevent roads from having huge curvature changes. We calculate the smoothness of each lane individually as sum of absolute angular errors. For each pair of three subsequent center points $\{i, j, k\} \in M_l$ we calculate the two

TABLE I: Parameters and their definition range for generating intersections.

Parameter	Definition Range
Number of arms	$ A \in [3, 4, 5]$
Lanes per driving direction	$ L^i , L^o \in [1, 2, 3, 4]$
Angle between arms	$\alpha_{ab} \geq 45^\circ$
Width of the gap	$g_a < 3 \text{ m}$

direction vectors γ_{ij} and γ_{jk} and derive the sum of absolute differences in the angle between the directions as

$$\delta_l = \sum_{i,j,k}^{M_l} |\arccos(\langle \gamma_{ij}, \gamma_{jk} \rangle)|. \quad (8)$$

We assume this difference δ_l to follow a normal distribution. Using the two metrics, the prior is calculated as

$$P(I_2) = s(I_2)^\tau \cdot \prod_{l \in L} P(\delta_l). \quad (9)$$

2) *Likelihood*: For the fitness of the estimated lane course and detected trajectories, the distance between the trajectory points and the center line is calculated. We reuse the orthogonal distance d_\perp from Section IV-A between a point on the center line $m \in M_l$ of lane l and a trajectory point $p \in t_i$ and assume it to follow a normal distribution which leads to

$$P(t_i|I_2) = \sum_{l \in L} \mathbf{1}(t_i, l) \prod_{p \in t_i} P_\perp(p|M_l, I). \quad (10)$$

V. EXPERIMENTAL EVALUATION

Because of the lack of publicly available datasets containing precise, lane-level maps, we created ground truth for 14 real-world intersections manually labeled in a map format. We separated those into small and big intersections. For a more meaningful, quantitative evaluation, we generated 1000 artificial intersections randomly using the parameters from Table I.

For both simulated and real intersections, we simulated vehicles traveling alongside the lanes, which resulted in a maximum of six trajectories per lane. The vehicle routes have been determined by randomly choosing two lanes in the intersection and following the ground truth center lines. For a more realistic simulation, we added Gaussian noise with $\Delta d \leftarrow \mathcal{N}(0 \text{ m}, 1 \text{ m})$ to every detection and some random, false detections around the center with $s \leftarrow \mathcal{U}(0 \text{ m}, 80 \text{ m})$, in order to replicate sensor measurements, which are often prone to noise.

All following experiments were conducted on a Ubuntu system with an Intel Core i7-8750H CPU 2.20 GHz (Turbo Boost: 4.1 GHz). The implementation has been done in the ROS framework in C++, allowing for fast execution time. In order to make this approach feasible for online usage, we limit the evaluation to execution times of approximately 150 ms, which results in limiting the number of samples that each step is allowed to generate, before the result for evaluation is drawn.

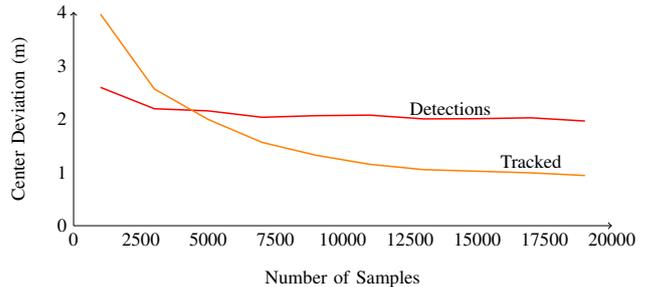


Fig. 6: Topology Estimation: Influence of number of samples on the accuracy of the intersection center.

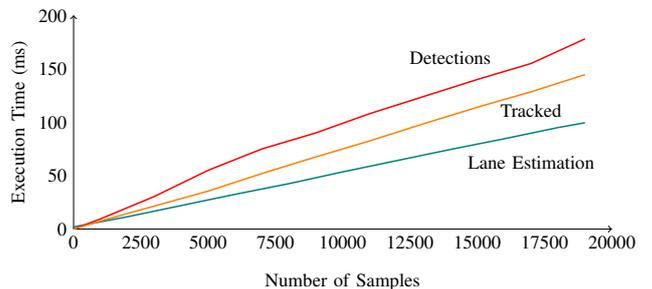


Fig. 7: Influence of the number of samples on the execution time.

A. Topology Estimation

First, we evaluate the topology estimation. We use distance measures on the individual parameters (see Section III-B) to determine the quality of the estimated intersection.

We found that our approach is able to detect the topology, describing the estimated number of arms, for all synthetic intersections correctly although we added noise and simulated false detections. Only on a single intersection (in the raw detections experiment), we could not detect the correct number of arms, which led to an accuracy of 99.90%.

The more complex lane-level topology, where the number of lanes and their direction in each arm is regarded but not the precise course, achieved an accuracy of 92.28% when executed on tracked data. Based on raw detections we could estimate the lane-level topology with an accuracy of 84.90%. The errors on the lane-level occurred because an additional lane at the outer borders of the models was estimated in some cases.

With 5000 samples we achieved an average execution time of 55.38 ms in case of raw detections and 35.72 ms in case of tracked objects. The difference can be explained by the considerable reduction of points in the tracked case. When changing the number of samples for an estimation the accuracy of e.g. the center (see Figure 6) improves exponentially, whereas the execution time changes linearly (see Figure 7).

For the center of the intersections we depicted the error distribution in Figure 8. Here, we can infer that the distribution is a consequence of the noise, that we added to our measurements. Additionally the center is prone to be wrongly

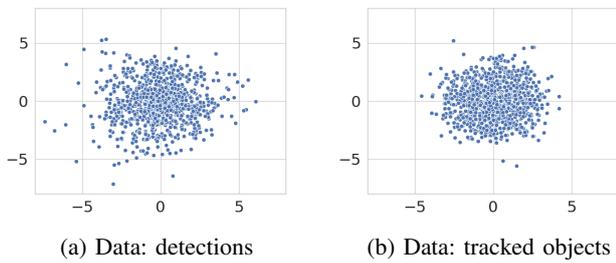


Fig. 8: Topology Estimation: Distribution of the error on the center position after 5000 samples. For the raw detection case, three outliers of $(1, -20)$, $(9, -19)$ and $(1, 21)$ are outside of the plot.

estimated, without leading to wrong intersections in terms of driveability. Since moving the center of the intersection influences the direction of the arms, a wrong center position might compensate errors in the lane angle estimation.

For most cases of tracked data, the road angles are less than 1° with an average error of 0.34° . In the detection case, we estimate the angle with a deviation of 0.97° on average.

In summary, our results show, that the topology estimation benefits from fewer detections and a tracking in beforehand, which increases the accuracy and speed of the approach.

B. Lane Course Estimation

For evaluating the course of the individual lanes, we used a different measure. We calculated the orthogonal distance between the estimated center line and the ground truth center line for all correctly estimated lanes. The overall quality of a lane was formulated as the mean Euclidean distance. The execution times presented here also include the preprocessing described in Section IV-A.

For the synthetic intersections, we achieved an accuracy of 14 cm on average. 20 000 sample points allowed for an execution time of 104.05 ms. With 5000 samples in the topology estimation and 20 000 for the lane course, we achieve an overall execution time of 140.05 ms. When choosing fewer samples the execution time is reduced as depicted in Figure 7, with exponentially worse results.

In the real-world scenarios we also used 20 000 sample points and achieved an accuracy of 18 cm for the small intersections and 27 cm for big ones. For smaller intersections, we get the results after 77.5 ms on average, whereas for big ones after 89 ms. The real scenarios could be solved quite faster, since we usually have fewer lanes per arm, than were simulated. For a qualitative evaluation of the real scenarios, we depicted examples in Figure 9.

VI. CONCLUSIONS

In this work, we showed an approach for estimating both the coarse lane-level intersection topology and the lane course inside and outside the intersection. On simulated data, the topology was estimated correctly in all cases based on trajectories of other traffic participants. For simulated and real-world intersections the lanes borders were detected with

an average error of 14 cm and 23 cm, resp. We are able to calculate the results within 140 ms and 113 ms. However, we chose this approach, because we can extract results at any time during the estimation, risking, of course, a less precise result.

Our model is able to represent a lot of different intersections because it is neither limited in the number nor the geometry of arms and their lanes. To our knowledge, similar approaches either could not achieve comparable results or require considerably more computation time.

Our method of estimating intersection topologies on the fly overcomes critical limitations of state-of-the-art autonomous driving solutions. It allows to check the validity of maps, to update outdated maps and even to drive in unknown environments.

Since our approach is based on trajectory data, it can also be used in dense traffic when traditional features for roadway recognition, e.g. markings or curbstones, fail due to occlusions. However, we plan to extend our framework to those traditional features in order to benefit from multiple independent features which promises to increase robustness even further.

REFERENCES

- [1] J. Ziegler, P. Bender, M. Schreiber, *et al.*, “Making Bertha Drive an Autonomous Journey on a Historic Route,” *IEEE Intell. Transp. Sys. Mag. (ITSM)*, vol. 6, no. 2, pp. 8–20, 2014.
- [2] F. Kunz, D. Nuss, J. Wiest, *et al.*, “Autonomous Driving at Ulm University: A Modular, Robust, and Sensor-Independent Fusion Approach,” in *IEEE Intell. Veh. Symp. (IV)*, pp. 666–673, June 2015.
- [3] A. L. Ballardini, D. Cattaneo, S. Fontana, *et al.*, “An Online Probabilistic Road Intersection Detector,” in *IEEE Int. Conf. on Rob. and Autom. (ICRA)*, pp. 239–246, 2017.
- [4] A. Geiger, M. Lauer, C. Wojek, *et al.*, “3D Traffic Scene Understanding from Movable Platforms,” *IEEE Trans. on Pattern Analysis and Machine Intell. (TPAMI)*, vol. 36, no. 5, pp. 1012–1025, 2014.
- [5] J. Beck and C. Stiller, “Non-Parametric Lane Estimation in Urban Environments,” in *IEEE Intell. Veh. Symp. (IV)*, pp. 43–48, 2014.
- [6] A. Joshi and M. R. James, “Generation of Accurate Lane-Level Maps from Coarse Prior Maps and Lidar,” *IEEE Intell. Transp. Sys. Mag. (ITSM)*, vol. 7, no. 1, pp. 19–29, 2015.
- [7] D. Töpfer, J. Spehr, J. Effertz, *et al.*, “Efficient Road Scene Understanding for Intelligent Vehicles Using Compositional Hierarchical Models,” *IEEE Trans. on Intell. Transp. Sys.*, vol. 16, no. 1, pp. 441–451, 2015.
- [8] A. Meyer, N. O. Salscheider, P. F. Orzechowski, *et al.*, “Deep Semantic Lane Segmentation for Mapless Driving,” in *IEEE/RSJ Int. Conf. on Intell. Robots and Sys. (IROS)*, pp. 869–875, 2018.
- [9] F. Dierkes, K. Siedersberger, and M. Maurer, “Corridor Selection Under Semantic Uncertainty for Autonomous Road Vehicles,” in *IEEE Int. Conf. on Intell. Transp. Sys. (ITSC)*, pp. 505–512, 2018.
- [10] A. Joshi and M. R. James, “Joint Probabilistic Modeling and Inference of Intersection Structure,” in *IEEE Int. Conf. on Intell. Transp. Sys. (ITSC)*, pp. 1072–1078, 2014.
- [11] C. Ruhhammer, N. Hirsenkorn, F. Klanner, *et al.*, “Crowdsourced Intersection Parameters,” in *IEEE Intell. Veh. Symp. (IV)*, pp. 581–587, 2014.
- [12] Y. Chen and J. Krumm, “Probabilistic Modeling of Traffic Lanes from GPS Traces,” in *SIGSPATIAL Int. Conf. on Advances in Geographic Information Sys.*, pp. 81–88, 2010.
- [13] O. Roeth, D. Zaum, and C. Brenner, “Road Network Reconstruction Using Reversible Jump MCMC Simulated Annealing Based on Vehicle Trajectories from Fleet Measurements,” in *IEEE Intell. Veh. Symp. (IV)*, pp. 194–201, 2016.
- [14] J. Wang and J. Kim, “Semantic Segmentation of Urban Scenes with a Location Prior Map Using Lidar Measurements,” in *IEEE/RSJ Int. Conf. on Intell. Robots and Sys. (IROS)*, pp. 661–666, 2017.



Fig. 9: Example results of real-world intersections. Aerial images: City of Karlsruhe, www.karlsruhe.de, dl-de/by-2-0

- [15] A. L. Ballardini, D. Cattaneo, and D. G. Sorrenti, "Visual Localization at Intersections with Digital Maps," in *IEEE Int. Conf. on Rob. and Autom. (ICRA)*, pp. 6651–6657, 2019.
- [16] H. Winner, "Automotive RADAR," in *Handbook of Driver Assistance Sys.: Basic Information, Components and Sys. for Active Safety and Comfort* (H. Winner, S. Hakuli, F. Lotz, and C. Singer, eds.), pp. 325–403, Springer Int. Publishing, 2016.
- [17] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
- [18] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, *et al.*, "Equation of State Calculations by Fast Computing Machines," *The J. of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.
- [19] C. Guindel, D. Martin, and J. M. Armingol, "Joint Object Detection and Viewpoint Estimation Using CNN Features," in *IEEE Int. Conf. on Vehicular Electronics and Safety (ICVES)*, pp. 145–150, 2017.
- [20] A. Dewan, T. Caselitz, G. D. Tipaldi, *et al.*, "Motion-Based Detection and Tracking in 3D Lidar Scans," in *IEEE Int. Conf. on Rob. and Autom. (ICRA)*, pp. 4508–4513, 2016.
- [21] F. Poggenhans, J.-H. Pauls, J. Janosovits, *et al.*, "Lanelet2: A High-Definition Map Framework for the Future of Automated Driving," in *IEEE Int. Conf. on Intell. Transp. Sys. (ITSC)*, pp. 1672–1679, 2018.