# Towards Automatic Inference of Layouts of Traffic Intersections for Smart Cities

Julien A. Vijverberg
*Cyclomedia Technology B.V.*
Zaltbommel, The Netherlands
email: jvijverberg@cyclomedia.com

Bart J. Beers
*Cyclomedia Technology B.V.*
Zaltbommel, The Netherlands
email: bbeers@cyclomedia.com

Peter H. N. de With
*Dept. of Electrical Engineering*
*Eindhoven University of Technology*
Eindhoven, The Netherlands
email: p.h.n.de.with@tue.nl

***Abstract*—Intersection-topology descriptions can help to improve traffic flow, but currently require significant manual creation effort. This paper describes ongoing work on a combination of algorithms to extract the drive lines across intersections from images and point clouds, captured by a ground-based vehicle. The extraction is based on paint striping and edge-of-road line features, which are clustered in subsequent stages into lane separators for all legs of the intersection. The GEO recall with paint striping is 0.29, and significantly contributes to successfully inferring the junction. However, this score decreases when edge-of-road features are added which is counter-intuitive. We indicate ways to combat this problem and further improve our results.**

***Keywords*—machine vision; terrain mapping; road transportation.**

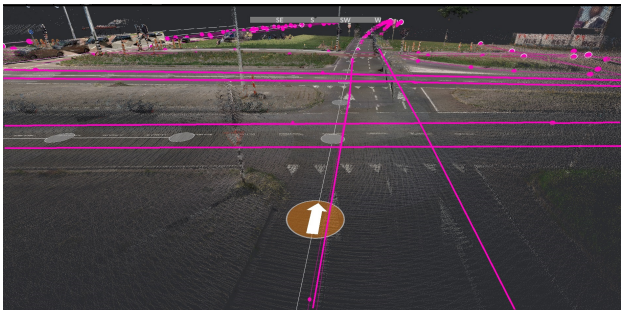


Figure 1: Colored point cloud data with drive lines (pink) and image recordings (gray circles).

## I. Introduction

Analysis of traffic and its infrastructure has gained significant attention in recent years, due to the growth in research on Smart City projects and corresponding themes. For these areas, there has been an continuously increasing focus on a better design of traffic intersections, to improve road safety on those intersections and its users, decrease of travel time and reduction of fuel consumption and related fuel emissions. To this end, new intersection designs have been compared to the current intersection layout using simulations [1]. At the same time, the advent of self-driving cars enforces many manufacturers to increasingly exploit knowledge of the environment to increase the reliability of their algorithms, *i.e.* , sensor-as-a-map [2]. The map data for these applications should include the location of drive lines, to identify signalling that indicates whether changing lanes is acceptable, capture indicated speed limits, etc.

This paper concentrates on generating the drive lines from street-view imagery and point clouds, in order to make accurate descriptions of traffic intersections. We aim at the intended drive lines of the intersection as we are interested in the topology. An example of the input data and drive lines is depicted in Figure 1. The use of drive lines has not been covered yet in most of existing literature, because many publications study modeling of highways, which are mostly straight and often have paint striping (*i.e.* , lane markings) separating all lanes and the normal lanes from the emergency lane (*i.e.* , the shoulder). In contrast, we are generally interested in intersections and their plurality of appearances, up to junctions where paint striping may be absent.

For street-view imagery, some research work [3][4] describes methods to generate intersection layouts, which exploit vehicle detections and their trajectories. For autonomous vehicles, it is considered that such data is available in any case. However, capturing systems specialized for mapping typically have low frame rates and high resolutions, and which do not capture data during non-driving situations to reduce the amount of storage required. The absence of capturing in such situations makes tracking objects and trajectory generation much more difficult and the intersection-layout inference more error-prone.

Other methods detect the paint striping, etc. [5] or the driving lines directly. LaneNet [5] and Gen-LaneNet [6] are proposed networks specialized to detect lanes from images and project them into 3D. These methods do not suggest how to combine the lines across images. Similarly, LaneGraphNet [7] estimates drive lines and directions from the birds-eye view, but does not show yet how to combine the tiles or paint striping. Although Máttyus *et al.* [8] use aerial images and OpenStreetMap as the primary sources for extracting lanes, they propose to construct a Markov Random Field but agree that special considerations are needed for intersections. Zhou *et al.* [9] propose to construct a semantic map and simulate driver behavior using a particle filter to extract drive lines. Initially, this appeared to be overly complex, since it would seem drive lines would follow logical rules based on the geometry of the lane separators, etc.

This paper and its focus on drive lines reports on ongoing research towards an algorithm to generate the intersection layout automatically with intended drive lines. For the algorithm development, data is analyzed from street-view recordings. The proposed research bases the derivation of the drive lines on paint striping and edge-of-road features and investigates their

 ISBN: 978-1-61208-983-6

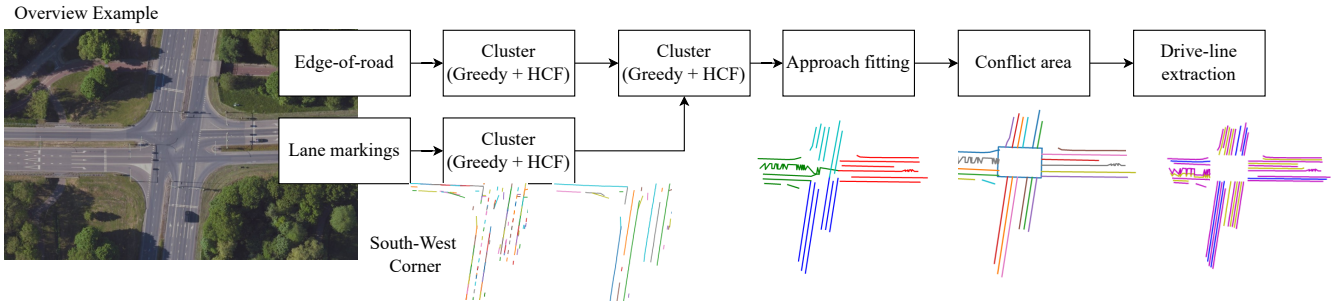



Figure 2: Pipeline flow and intermediate results at several steps in the proposed pipeline.

fusion by manual tuning of the clustering algorithms. This approach should lead to the automated generation of junction topologies, and make those available for traffic guidance and safety control. Thereby, this paper contributes in two ways: (1) an algorithm for automated generation of junction topologies; (2) analysis of the topology detection performance and ways for improving the algorithm in a follow-up development.

This paper is divided as follows. Section II is split into Section II-A, describing the properties of the data and Section II-B proposing a novel drive-line estimation algorithm. The experiments with these methods are described in Section III and visual and numerical results are provided. Section IV presents conclusions.

## II. Method

### A. Input Data

For the automated generation of junctions, we consider the input system, *i.e.* , the capturing, the image processing and the initial vision processing, as a given.

*1) Recordings:* The data are captured using a vehicle which is equipped with an omni-directional camera-system using five video cameras, a LiDAR scanner for point clouds and GPS receiver. The capture occurs during the daytime and with relatively good weather. The sensors have been calibrated with respect to each other, such that we can compute the world position for most pixels in the image with high accuracy. The input data comprises vehicle poses, RGB images and point clouds acquired with the LiDAR scanner. An example of a colored point cloud is shown in Figure 1.

*2) Front-end Vision:* The segmentation is performed by a standard U-Net [10] on the RGB images. Each paint-striping line is projected into the 3D world using the point clouds, *e.g.* , where the horizontal world coordinates are Universal Transverse Mercator coordinates and the third dimension is ellipsoidal height. The result of this step is a list of small 3D line objects. Note that the paint striping intentionally outputs short lines, instead of doing a best guess per image which may yield longer lines and be more consistent. In this way, false positives and true positives are less likely to be merged in a single line and later stages can make decisions per line segment without breaking up these paint-striping lines back into smaller segments. In contrast to paint striping, edge-of-road lines are extracted from polygons with road-surface types [11]. In contrast to [11], we have used an image-segmentation pipeline based on Mask R-CNN [12] with a similar pipeline architecture as the paint-striping pipeline.

### B. Proposed Intersection-Inference Algorithm

The proposed intersection-inference algorithm uses the 3D (line) detections of the paint striping and the edge-of-road. The flow of the data is outlined in Figure 2. Note that we did not yet include estimating drive lines across the intersection, since the most common approach using splines [13] is considered of low importance. The following sub-sections will describe the listed algorithmic steps.

*1) Per-feature lane-separator clustering:* We start with greedy line clustering with the purpose to reduce the amount of lines for the next step. The only features are the distance between the start- and end-points and the angle between the lines.

Highest-Confidence First (HCF) clustering is the main clustering step to both classify all (combined) lines as false or true positive and to determine which lines belong together. This has been applied earlier in multiple object tracking [14]. In this case, given a set of lines $\mathcal{T} = \{T_0, T_1, T_2, ...\}$, we aim at iteratively merging the pair of lines $T_i, T_j$ into the line $T_i \cup T_j$, which reduces an energy function $E$, specified by

$$\Delta E_{(i,j)} = E\left(T_i, T_j | \mathcal{T}\right) - E\left(T_i \cup T_j | \mathcal{T}\right), \qquad (1)$$

where $\Delta E_{(i,j)}$ can be interpreted as a negative log-likelihood and $\Delta E_{(i,j)} < 0$ means the merged line is be more likely than two individual lines. Similar to the tracking application [14], we define separate energy functions for individual features, including the distance, the difference in orientation between two lines, distance between the extrapolated lines, etc.

Another similarity is the energy difference, which can be combined with an energy function, describing the likelihood whether a line is a false positive. Cues to determine the validity of a line include the length, the variance in orientations and the existence of almost parallel lines at approximately the lane width (about 3 meters).

To speed-up the execution of the algorithm, we sub-divide the problem in a spatial hierarchical fashion, cache results $\Delta E_{(i,j)}$, merge batches of lines with similar confidence, and use an R-tree to search for lines.

 ISBN: 978-1-61208-983-6 

**Input:** $\mathcal{I} = \{p_1, p_2, p_3, ...\}$
**Result:** $\mathcal{I}'$
$\mathcal{I}'' \leftarrow \mathcal{I}' \leftarrow \mathcal{I}$;
**do**
  /* Select point to remove */
  $p' = \arg\min_{p \in \mathcal{I}'} A(\mathcal{I}' - \{p\}))$;
  $\mathcal{I}' \leftarrow \mathcal{I}''$ ;
  $\mathcal{I}'' \leftarrow \mathcal{I}' - \{p'\}$ ;
  /* Hull decreases significantly? */
**while** $A(\mathcal{I}') < \lambda A(\mathcal{I}'')$;

Figure 3: Algorithm to determine the conflict area

*2) Combined clustering:* During this step, the clustering as described in Section II-B1 is executed on the combined lane separators, albeit with different parameter settings.

*3) Approach fitting:* At this point, we have a large set of lane separators which are slightly clustered. The goal of this step is to find which approaching roads towards the intersection exist (*approaches*). In addition, we need to determine which lane separators belong to the same approach.

For this fitting, we opt for a RanSaC algorithm [15]. For each iteration, the algorithm randomly selects a (pivot) line. With the selection of all lines on one initial side of the intersection with approximately the same orientation as the pivot line, the approach is fit by projecting all lane separators onto the normal of the pivot line. These lane separators are clustered based on the 1D distance over the line. The clusters that do not lead to sufficiently long lines, are considered false positives for the purpose of the approach fitting. The same method is reused for to find new approaches with the remaining lines until we cannot fit anymore approaches.

To assess the fitting quality between iterations of the algorithm, we compare the length of the lines classified as true and false positives. Moreover, we involve the number of approaches in the comparison, where the a-priori most likely number is 4 and a penalty is given to diverting numbers.

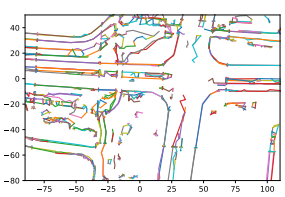
*4) Approach-based clustering:* Since an estimate of the approach is now available, we can more reliably cluster lane separators, that are distant and could not be merged earlier. This merging is achieved by executing the HCF clustering once again, albeit with different parameter settings.

*5) Conflict-area estimation:* The conflict area is defined as the area in which the connecting lanes intersect. An initial estimate of the conflict area can be found as the convex hull around the intersection points of the lane separators from different approaches. This initial estimate of the conflict area leads to the creation of outliers. Using the algorithm in Figure 3, this estimate is improved by comparing the convex-hull surface area $A(\mathcal{I})$ of a set of intersection points $\mathcal{I}$ over consecutive iterations.

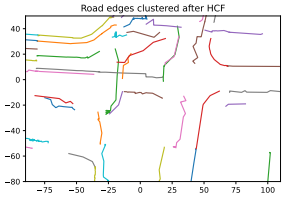
*6) Drive-line extraction:* The drive lines are lines fitted between the lane separators, as illustrated in the right-hand part of Figure 2.

TABLE I: QUANTITATIVE COMPARISON TO REPORTED RESULTS OF OTHER METHODS.

| | GEO | | Topology Accuracy [3] |
|---|---|---|---|
| | recall | precision | |
| Overall w/o edge-of-road | 0.29 | 0.69 | n.a. |
| Overall with edge-of-road | 0.11 | 0.34 | n.a. |
| He and Balakrishnan [13] | 0.821 | 0.835 | n.a. |
| Zhou *et al.* [9] | n.a. | n.a. | 0.8 |
| Geiger *et al.* [3] | n.a. | n.a. | 0.92 |



(a) Road edges after greedy




(b) Road edges after HCF

Figure 4: Edge-of-road example for the same intersection as depicted in Figure 2, showing high amount of incorrect edges.

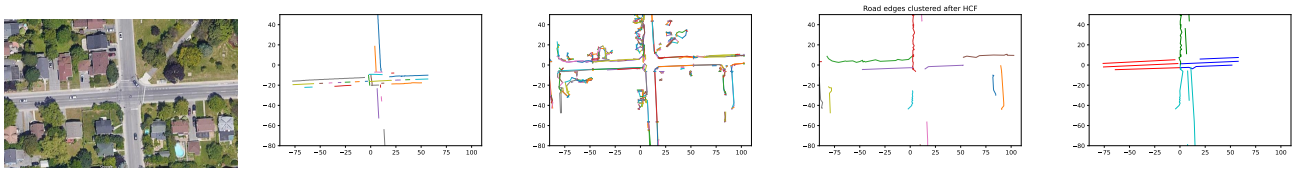
## III. EXPERIMENTS

As already mentioned in Section II-B, Figure 2 visualizes some qualitative results for an intersection where paint striping is abundantly present. Figure 4 shows the road edges after greedy clustering (Figure 4a and HCF clustering (Figure 4b). As Figure 4a shows, an important source of noise is caused by the bike lanes surrounding the intersection and passing under the roads for the vehicles. This makes the HCF of the road edges clustering significantly less successful.

Another visual example is shown in Figure 5 in which the sides of the lanes have not been marked by paint striping and the middle of the north-south lane has only been marked for the first 25 meters. The final result is facilitated by false positive paint striping, *e.g.* , the vertical blue line in the top middle lane in Figure 5b. Comparing greedily clustered road edges (Figure 5c to the result of HCF clustering in Figure 5d, it can be observed that many road edges cannot be clustered to significant line features, so that they remain of minor importance and are consequently removed by HCF.

To quantitatively evaluate the proposed algorithm, we use the small dataset with 4 intersections including the intersections depicted in Figure 2 and Figure 5a, but exclude the drive lines across the intersection. These intersections are located in Almelo, the Netherlands and Kingston, Canada. The input data follows the description in Section II-A. The GEO metric [13] is used for comparison, yielding detections scores for the actual drive lines. To compute the GEO metric, we split each line in segments (of say 0.25 m) and match the points of the ground-truth lines and predicted lines. From these matches, one can derive recall and precision. Table I lists the results for all intersections in the above dataset. It is clear that adding edge-of-road yet leads to worse performance compared to using paint striping only, due to its noisy character. Interestingly, He

 ISBN: 978-1-61208-983-6




(a) Aerial overview [16] (b) Paint striping (HCF) (c) Edge-of-road (greedy) (d) Edge-of-road (HCF) (e) Approach fit

Figure 5: Intermediate results for paint striping and road edges for an exemplary intersection in Kingston, Ontario, Canada.

and Balakrishnan [13] perform better by directly estimating drive lines from aerial images using the same metrics but a different dataset. Furthermore, Table I also compares our results with other papers albeit with both different datasets and metrics, and hence, only give a global indication.

## IV. Conclusion

In this paper, we have presented a combination of algorithms which successfully infers the intersection topology in some cases. Paint striping is evidently an informative cue with which we can successfully infer drive lines. However, our research hypothesis that edge-of-road features would contribute has encountered issues, due to the ability of humans to smoothly integrate a-priori knowledge. This mainly applies to the fine-tuning of the energy function defined in Section II-B1, which is not yet exploited in our algorithmic setting.

We have started our research with a feature exploration on junction topology, to create knowledge on essential features and follow-up directions for finding a good method. After our research, we have found that it is interesting to compare our system, but it is difficult to initially select the preferred method: The algorithm of Zhou *et al.* [9] simulates driver behavior to obtain drive lines, while Zürn *et al.* [7] directly create drive lines without further comparison. Hence, a more detailed comparison between these methods is desired.

Besides this, a significant improvement should be possible by enhancements in the edge-of-road input lines and clustering. The segmentation can become more accurate by improving the annotation quality of the road dataset which has currently noise problems due to projection shifts. Furthermore, the dataset for junctions needs to be significantly enlarged. These last two steps are integrated into our current research.

## Acknowledgment

This work has been funded by the Dutch government and supported by ITEA as part of the ITEA3 18036 SMART Mobility Project.

## References

[1] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, *et al.*, "Microscopic traffic simulation using sumo," *21st Int. Conf. on Intelligent Transportation Systems (ITSC)*, IEEE, pp. 2575–2582, 2018.

[2] J. K. Suhr, J. Jang, D. Min, and H. G. Jung, "Sensor fusion-based low-cost vehicle localization system for complex urban environments," *IEEE Trans. on Intelligent Transportation Systems* vol. 18 no. 5, pp. 1078–1086, 2017, DOI: 10.1109/TITS.2016.2595618.

[3] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3D traffic scene understanding from movable platforms," *IEEE Trans. on Pattern Analysis and Machine Intelligence* vol. 36 no. 5, pp. 1012–1025, 2013,

[4] A. Meyer, J. Walter, M. Lauer, and C. Stiller, "Anytime lane-level intersection estimation based on trajectories of other traffic participants," *IEEE Intelligent Transportation Systems Conf. (ITSC)*, pp. 3122–3129, 2019.

[5] Z. Wang, W. Ren, and Q. Qiu, *Lanenet: Real-time lane detection networks for autonomous driving*. Available from: https://arxiv.org/pdf/1807.01726.pdf, 2022.05.05.

[6] Y. Guo, G. Chen, P. Zhao, W. Zhang, J. Miao, *et al.*, "Gen-lanenet: A generalized and scalable approach for 3d lane detection," *European Conf. on Computer Vision*, pp. 666–681, 2020.

[7] J. Zürn, J. Vertens, and W. Burgard, "Lane graph estimation for scene understanding in urban driving," *IEEE Robotics and Automation Letters* vol. 6 no. 4, pp. 8615–8622, 2021, DOI: 10.1109/LRA.2021.3111433.

[8] G. Máttyus, S. Wang, S. Fidler, and R. Urtasun, "HD maps: Fine-grained road segmentation by parsing ground and aerial images," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3611–3619, 2016.

[9] Y. Zhou, Y. Takeda, M. Tomizuka, and W. Zhan, "Automatic construction of lane-level hd maps for urban scenes," *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, IEEE, pp. 6649–6656, 2021.

[10] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Int. Conf. on Medical Image Computing and Computer-Assisted Intervention*, Springer, pp. 234–241, 2015.

[11] Q. Bai, R. C. Lindenbergh, J. A. Vijverberg, and J. A. P. Guelen, "Road type classification of MLS point clouds using deep learning," *The Int. Arch. of Photogrammetry, Remote Sensing and Spatial Information Sciences* vol. 43, pp. 115–122, 2021,

[12] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," *IEEE Int. Conf. on Computer Vision (ICCV)*, pp. 2980–2988, 2017.

[13] S. He and H. Balakrishnan, "Lane-level street map extraction from aerial imagery," *IEEE/CVF Winter Conf. on Applications of Computer Vision (WACV)*, pp. 1496–1505, 2022. DOI: 10.1109/WACV51458.2022.00156.

[14] J. A. Vijverberg, C. J. Koeleman, and P. H. N. With, de, "Towards real-time and low-latency video object tracking by linking tracklets of incomplete detections," *Proc. of the 10th IEEE Int. Conf. on Advanced Video and Signal Based Surveillance*, pp. 300–305, 2013. DOI: 10.1109/AVSS.2013.6636656.

[15] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Comm. of the ACM* vol. 24 no. 6, pp. 381–395, 1981,

[16] Google, *Google maps*. Available from: https://www.google.com/maps, 2022.05.05.

 ISBN: 978-1-61208-983-6 