Learning Lane Graphs from Aerial Imagery Using Transformers

Martin Büchner[*], Simon Dorer[*], Abhinav Valada

University of Freiburg

***Abstract*—The robust and safe operation of automated vehicles underscores the critical need for detailed and accurate topological maps. At the heart of this requirement is the construction of lane graphs, which provide essential information on lane connectivity, vital for navigating complex urban environments autonomously. While transformer-based models have been effective in creating map topologies from vehicle-mounted sensor data, their potential for generating such graphs from aerial imagery remains untapped. This work introduces a novel approach to generating successor lane graphs from aerial imagery, utilizing the advanced capabilities of transformer models. We frame successor lane graphs as a collection of maximal length paths and predict them using a Detection Transformer (DETR) architecture. We demonstrate the efficacy of our method through extensive experiments on the diverse and large-scale UrbanLaneGraph dataset, illustrating its accuracy in generating successor lane graphs and highlighting its potential for enhancing autonomous vehicle navigation in complex environments.**

I. Introduction

Modern self-driving vehicles rely on accurate topological representations of their immediate surroundings to plan and navigate safely. While onboard sensors such as cameras or LiDARs provide vital observations used in localization [7, 1], scene understanding [2, 16, 6] and downstream decision making [17], they come with significant challenges such as limited availability and occlusions. Nonetheless, robust planning and control require high-fidelity topologies such as HD map data to act in a timely and safe manner. Thus far, both LiDARs and cameras have been successfully used for predicting the road topology [13, 14, 11] in the vicinity of the ego vehicle. However, the input sensor data is starkly affected by occlusions leading to frequent hallucinations, which is common for supervised learning techniques.

Lane graphs are a fundamental element of topological maps as they provide lane-level connectivity information of the road network suitable for planning, trajectory prediction, and control. However, predicting a sparse data structure such as a lane graph is usually hard due to its connectivity [12, 21]. In order to resolve occlusions in the surround view, several methods have been recently proposed that leverage aerial and bird's-eye-view (BEV) data for improving the lane graph prediction accuracy [9, 3, 21]. While these aerial methods provide useful representations, they either lack capabilities in out-of-distribution scenarios [9], suffer from inaccurate connectivity [21], or cannot provide the required level of fidelity of, e.g., the node positions [3].

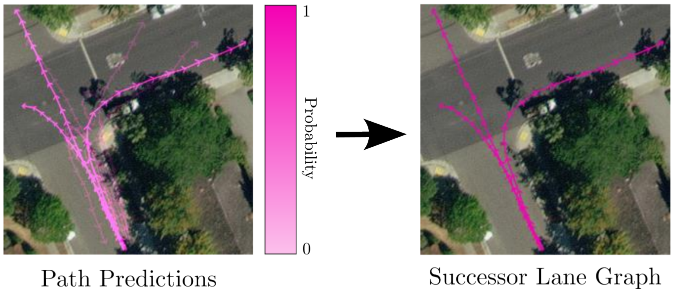

Fig. 1: We present aerial lane graph transformers (ALGT) for learning feasible high-fidelity traversals of successor lane graphs. The left image shows raw predicted traversals while their opacity represents the predicted probability score. The right image shows the thresholded and aggregated traversals forming a successor lane graph.

In this work, we tackle the problem of predicting successor lane graphs, as introduced in the *UrbanLaneGraph* dataset [3]. Successor lane graphs provide actionable representations given an initial starting pose of the ego-vehicle useful for short-term planning as depicted in Fig. 1. For this task, we follow Liao et al. [12] in decomposing the graph-level task into multiple path-level predictions. Leveraging recent developments in set-based prediction (DETR) [4, 15, 20], we introduce the Aerial Lane Graph Transformer (ALGT) model. Our proposed network takes aerial images including context as input to predict successor lane graphs within a region of interest at the center of the input image as shown in Fig. 2. The successor lane graphs themselves are predicted as multiple feasible traversals of a directed-acyclic graph, which are later fused to form final predictions. As part of this work, we demonstrate the benefit of using polyline path representations over Bézier parametrizations for improved graph prediction. In addition, we present insights into the capabilities of various image backbones used for encoding aerial imagery.

To summarize, our main contributions are:

- We present the ALGT framework for predicting successor lane graphs from aerial imagery using transformers.
- We demonstrate the benefit of using polyline representations over Bézier parametrization and ablate over various network architectures.

[*] Equal contribution.

This work was funded by the German Research Foundation (DFG) Emmy Noether Program grant number 468878300 and an academic grant from NVIDIA.

II. Technical Approach

Our approach aims to predict successor lane graphs $\hat{\mathcal{G}}$ in terms of path proposals $\hat{Y}$ given aerial image crops $\mathbf{I}$ in a supervised learning manner. In the following, we outline two lane graph representations (Sec. II-A), introduce the ALGT model in Sec. II-B, detail the training procedure in Sec. II-C, and describe the aggregation of proposal paths to form a single successor graph $\hat{\mathcal{G}}$ in Sec. II-D.

A. Lane Graph Representation

We adopt the path-level representation strategy as proposed by *MapTR* [13] and decompose $\mathcal{G}$ into a set of maximal length paths. These paths represent a collection of traversals from the initial ego pose v_0 to arbitrary terminal nodes v_{end} of $\mathcal{G}$. This path-level decomposition is necessary for predicting set-wise outputs later using the transformer architecture. We can represent these traversal paths either as polylines or as Bézier curves:

- A polyline $\mathcal{P}_n$ consists of n straight line segments linked sequentially to create a piece-wise linear path. It is defined by a series of points $\boldsymbol{p}_0, \boldsymbol{p}_1, \ldots, \boldsymbol{p}_k$ located in a two-dimensional space. Each adjacent pair of points $(\boldsymbol{p}_i, \boldsymbol{p}_{i+1})$ is connected by a straight line.
- A Bézier curve $\mathcal{B}_n$ in n dimensions is specified by $n+1$ control points $\boldsymbol{b}_0, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_n$, with $\boldsymbol{b}_0$ marking the start and $\boldsymbol{b}_n$ the end of the curve. The intermediate control points influence the curve's shape through Bernstein polynomials of the form $B_{i,n} = \binom{n}{i} t^i (1-t)^{n-i}$. A Bézier curve is formulated as

$$\mathcal{B}_n(t) = \sum_{i=0}^{n} \boldsymbol{b}_i B_{i,n}(t) \quad \text{for} \quad 0 \leq t \leq 1. \tag{1}$$

We visualize these decomposed path representations including the Bézier control points in Fig. 2.

B. Aerial Lane Graph Transformer

The input to the proposed transformer model is an RGB image crop $\boldsymbol{I}$, from which the central region represents the actual target image $\boldsymbol{I}_{roi}$, while the outer margin serves as contextual information. We aim to predict a successor lane graph $\hat{\mathcal{G}}$ within the image domain $\boldsymbol{I}_{roi}$, where the initial node v_0 is positioned at the bottom center of $\boldsymbol{I}_{roi}$. This is also visualized in Fig. 2 where the dark region represents the context and the inner region represents $\boldsymbol{I}_{roi}$. Our ALGT model is structured around three primary components. Initially, an image backbone extracts relevant features from the input image $\boldsymbol{I}$ (Sec. II-B1). Following this, a transformer-based path predictor derives a set of path proposals $\hat{Y}$ (Sec. II-B2). These proposals are later aggregated. An overview of our model's architecture is illustrated in Fig. 3.

1) Aerial Image Backbone: To extract features from the context RGB image $\boldsymbol{I}$, we leverage an ego lane regression network, similar to LaneGNN [3]. The network is built upon two stacked PSPNet [19] instances, initially generating a context lane regression mask. In combination with the original input image, the context regression mask is then used to predict an ego lane regression mask. We disregard the PSPNet's upsampling layers and prediction heads and use it as our image backbone, transforming $\boldsymbol{I}$ into feature maps $\boldsymbol{F}_1$. Additionally, we integrate an image classification backbone, such as ResNet [8] or ViT [5] that produces another set of feature maps $\boldsymbol{F}_2$. Lastly, we combine the feature maps $\boldsymbol{F}_1$ and $\boldsymbol{F}_2$ in a learned fashion to forge a unified feature representation $\boldsymbol{F} = \text{CONCAT}(\boldsymbol{F}'_1, \boldsymbol{F}'_2)$.

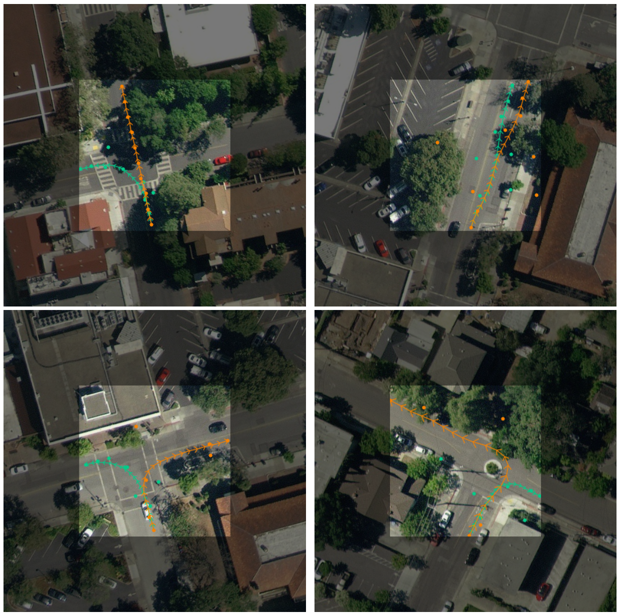

Fig. 2: Visualization of decomposed successor lane graphs of the UrbanLaneGraph dataset [3]. We choose Bézier curves of degree 10 and polylines from 20 sample points on this curve. All paths along with their Bézier control points are depicted in varying colors. The context part of the samples is darkened.

2) Path Prediction: The path predictor is the core component of our architecture and utilizes the DETR framework [4]. The path predictor consists of two stages. In the encoder stage, the image features $\boldsymbol{F}$ are encoded using an iterative self-attention mechanism. In the following decoder stage, this encoded version of the extracted features is then used to transform a predefined number n_Q of lane path queries into proposal vectors using self- and cross-attention. Finally, two prediction heads predict all n_Q path proposals in parallel. In the following, we detail the specific components:

Transformer Encoder: To maintain the positional relationships within the image features before feeding them into the transformer encoder, we apply a fixed two-dimensional sinusoidal positional encoding $\boldsymbol{P}_{enc}$ [18]. We add the positional encoding onto the original features and flatten the feature maps $\boldsymbol{Z} = \text{FLATTEN}(\boldsymbol{F} + \boldsymbol{P}_{enc})$ and feed them to the transformer encoder to yield $\boldsymbol{Z}'$.

Transformer Decoder: The decoder takes the encoded image features $\boldsymbol{Z}'$ to generate a set of path proposals. We employ a set of fixed-size vectors $\boldsymbol{Q} = (\boldsymbol{q}_1, \ldots, \boldsymbol{q}_{n_Q})^T$ as trainable transformer queries, where each query $\boldsymbol{q} \in \boldsymbol{Q}$ is a higher-dimensional representation of a potential lane path. This query design aims to capture various paths without specifying the

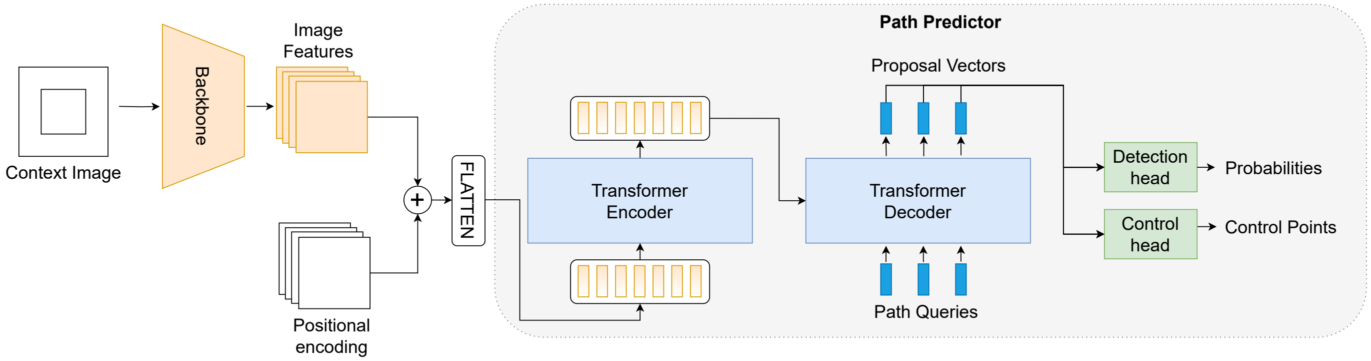

Fig. 3: Overview of our ALGT model for successor lane graph prediction. An image backbone extracts relevant features from the context image that are passed through a transformer-based path predictor to produce successor lane graph proposals.

order of prediction for each path, eliminating the need for positional encoding in the decoder segment of our model. By employing a standard decoder structure with n_{dec} layers, we refine each path query $\boldsymbol{q} \in \boldsymbol{Q}$ into a path proposal vector $\boldsymbol{q}' \in \boldsymbol{Q}'$.

Prediction Heads: For each path proposal vector $\boldsymbol{q}' \in \boldsymbol{Q}'$, we obtain the following outputs using two distinct prediction heads: First, the detection head predicts the likelihood that the proposed path exists using an MLP followed by a Sigmoid activation layer to obtain outputs in $[0, 1]$. Secondly, the control head predicts the configuration of each path as a series of control points $\boldsymbol{A} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_{n_{cp}})^T$. The MLP-based control head projects the points' coordinates onto a normalized $[0, 1]$ scale wrt. $\boldsymbol{I}_{target}$. This process is executed concurrently for all path proposals $\boldsymbol{q}' \in \boldsymbol{Q}'$, producing a collection of path with associated probabilities $\hat{Y} = \{(l_i, \boldsymbol{A}_i)\}_{i=1}^{n_Q}$ as shown in Fig. 1.

C. Training

Optimizing the ALGT model is challenging due to varying set sizes between the ground truth and predicted paths. First, we identify the optimal matching between the predicted paths $\hat{Y}$ and the ground truth paths Y using Hungarian matching [10]:

$$\mathcal{C}_{match}(Y_i, \hat{Y}_j) = \sum_{k=1}^{n_{cp}} \alpha \cdot d(\boldsymbol{y}_i^k, \boldsymbol{a}_j^k) + \beta \cdot (1 - l_j), \quad (2)$$

where $d(\boldsymbol{y}_i^k, \boldsymbol{a}_j^k)$ represents the Manhattan distance between the k-th control points of Y_i and $\hat{Y}_j$, while α and β are constants that balance the contributions of spatial discrepancies and classification confidence, respectively. Following the obtained optimal matching, we minimize a composite loss that penalizes the regression errors of control points and the classification inaccuracies of the predicted paths:

$$\mathcal{L} = \alpha \cdot \sum_{Y_i \in Y} \mathcal{L}_{mse}(Y_i, \hat{Y}_{\sigma^*(i)}) + \beta \cdot \mathcal{L}_{bce}(l_i), \quad (3)$$

where the MSE loss regularizes the errors over all control points and the binary cross-entropy penalizes path likelihoods.

D. Path Filtering and Aggregation

In the aggregation step of our approach, we convert the set of path proposals $\hat{Y}$ into the successor lane graph representation $\hat{G}$. First, we disregard all path proposals that do not meet a certain minimum likelihood threshold p_{min}. Next, we fuse the thresholded paths into a cohesive successor lane graph $\hat{G}$ by adding directedness and merging the individual path graphs iteratively: Given a candidate path graph to merge we identify the nearest nodes v^* in the existing graph using L2 distances. If the distance $d(v, v^*)$ is less than or equal to d_{max}, indicating the nodes represent the same location, we merge v into v^*. Otherwise, v is added as a new node. After completing the iterative merging process, we obtain our final successor lane graph, $\hat{\mathcal{G}}$, as shown in Fig. 1.

III. Experimental Evaluation

In the following, we present our experimental findings on the Palo Alto split of the *UrbanLaneGraph* benchmark dataset [3]. To quantify our experimental results we make use of the same set of metrics utilized in the benchmark: TOPO and GEO measuring geometric and topological similarity, the average path length similarity (APLS), the split detection accuracy (SDA), and the image-based Graph IoU. We present quantitative evaluations including an ablation study and compare our proposed ALGT model with the strong LaneGNN baseline [3]. While the LaneGNN method shows high topological accuracy it suffers from inaccurate node positions, which show significant offsets with respect to the ground truth graphs. We refer to the LaneGNN paper [3] for more insights. Additionally, we also present qualitative results in Fig. 4.

A. Quantitative Results

We conduct an ablation study on the utilized path representation, different image backbones, and different encoder-decoder sizes of the network. As detailed in Tab. I, we demonstrate that the polyline-based prediction greatly outperforms the Bézier parametrization across all metrics, which leads us to conclude its greater suitability for set-based lane graph prediction tasks. Moreover, we observe that the PSP-Net image backbone outperforms variants that solely utilize

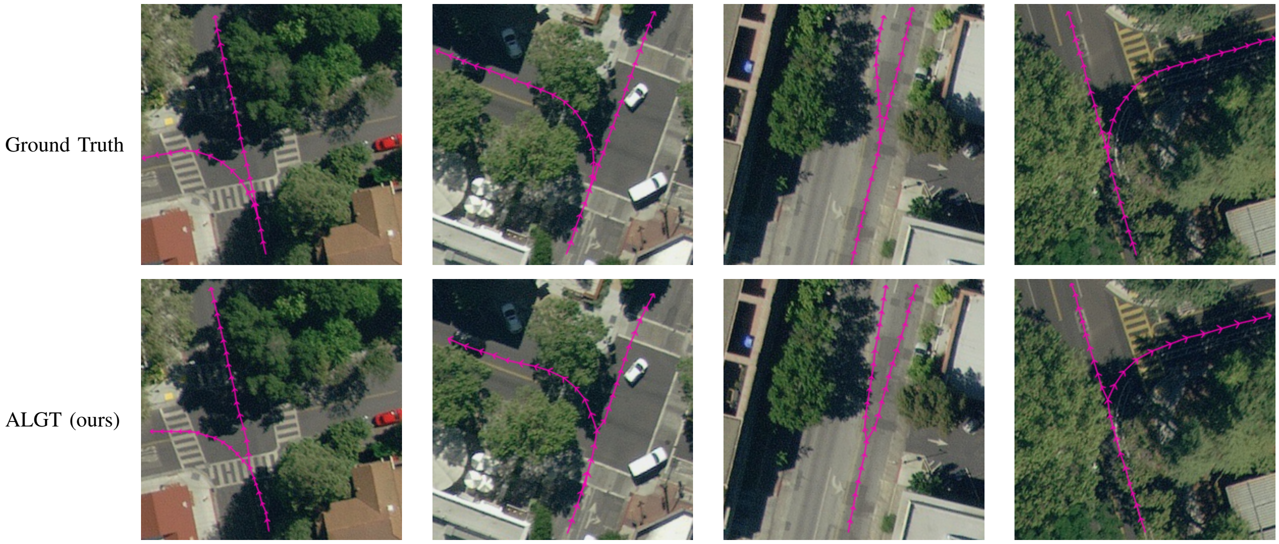

Fig. 4: Qualitative results obtained by our ALGT model in comparison to the ground truth. The top row represents the ground truth, while the bottom row presents our model's predictions. Our proposed architecture predicts highly accurate lane graphs that do not suffer from sampled node positions as LaneGNN and shows high split detection accuracy. In general, the obtained graphs show a smooth characteristic.

Variant	TOPO P/R	GEO P/R	APLS	SDA$_{20}$	SDA$_{50}$	Graph IoU
Path Representation						
Bézier	0.395/0.339	0.567/0.527	0.619	0.191	0.405	0.290
Polyline	**0.479/0.420**	**0.639/0.594**	**0.664**	**0.251**	**0.479**	**0.338**
Backbone						
ResNet-50	0.268/0.223	0.433/0.396	0.509	0.158	0.367	0.200
ViT-B-16	0.253/0.212	0.418/0.383	0.465	0.114	0.348	0.195
PSPNet	0.479/**0.420**	0.639/**0.594**	0.664	**0.251**	**0.479**	0.338
PSPNet + ResNet-50	**0.485**/0.414	**0.644**/0.587	**0.665**	0.237	0.447	**0.345**
Architecture						
$(1, 1, 64, 10)$	0.432/0.353	0.598/0.536	0.643	0.193	0.344	0.304
$(2, 2, 128, 20)$	0.474/0.402	0.631/0.575	0.651	0.223	0.420	0.331
$(4, 4, 128, 10)$	**0.479/0.420**	**0.639/0.594**	**0.664**	**0.251**	**0.479**	**0.338**

TABLE I: Ablation studies conducted on our foundational ALGT model, utilizing the Palo Alto validation set from the UrbanLaneGraph dataset [3]. The underlined variants indicate the selected parameters used for comparison again LaneGNN. The highest-performing values in each category are highlighted in bold. The architecture ablation study compares varying configurations of the number of encoder and decoder layers, the model's hidden dimensionality and the number of queries: $(n_{enc}, n_{dec}, c, n_Q)$. Higher values are better, best results are written bold.

standard image backbones such as ResNet or ViT that are often rather used in detection tasks. When coupling a PSPNet instance with a ResNet-50 as described in Sec. II-B1, we observe similar results compared to only relying on a PSPNet instance. Furthermore, a variation of the transformer encoder and decoder sizes shows that a higher number of encoder and decoder layers improves predictions while keeping the number of path proposals at 10.

As depicted in Tab. II, we observe competitive results on the Palo Alto test data split when comparing our ALGT model with the LaneGNN baseline [3]. While LaneGNN shows greater topological accuracy, the ALGT model vastly outperforms the LaneGNN on the APLS metric. This outlines that the ALGT architecture does not suffer from inaccurate node positions and ultimately resolves this limitation inherent to sampled node manifolds used by LaneGNN.

Method	TOPO P/R	GEO P/R	APLS	SDA$_{20}$	SDA$_{50}$	Graph IoU
LaneGNN	**0.584/0.744**	0.582/**0.739**	0.177	0.220	0.367	**0.378**
ALGT	0.481/0.437	**0.645**/0.606	**0.714**	**0.224**	**0.497**	0.343

TABLE II: Quantitative results of our ALGT model in comparison with the LaneGNN [3] baseline mode for the successor lane graph prediction task. Both models are trained and tested on the subset of Palo Alto from the UrbanLaneGraph dataset [3]. Higher values are better, best values written bold.

B. *Qualitative Results*

We present additional qualitative results of the proposed ALGT model in Fig. 4. As depicted, our proposed method predicts highly accurate lane graphs that do not suffer from sampled node positions (as LaneGNN) and shows high split detection accuracy. In addition, we show a number of failure cases in Fig. 5 in the appendix.

IV. Conclusion

We presented a novel successor lane graph prediction approach that generates highly accurate paths while not suffering from graph initialization errors. This leads to improved path accuracy and allows better split point predictions at intersections. Future work could address the temporal aggregation of the transformer-based predictions and tackle the out-of-distribution problem inherent to large-scale lane graph prediction.

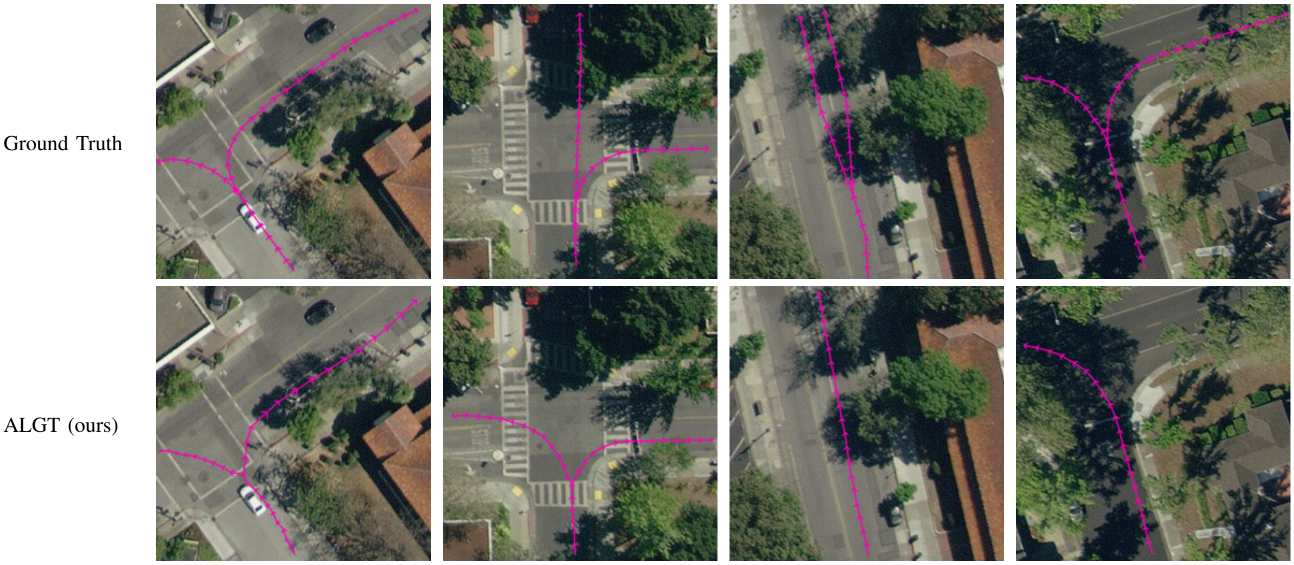

Fig. 5: Failure cases of the ALGT model compared to GT

References

[1] José Arce, Niclas Vödisch, Daniele Cattaneo, Wolfram Burgard, and Abhinav Valada. Padloc: Lidar-based deep loop closure detection and registration using panoptic attention. *IEEE Robotics and Automation Letters*, 8(3):1319–1326, 2023.

[2] Martin Büchner and Abhinav Valada. 3d multi-object tracking using graph neural networks with cross-edge modality attention. *IEEE Robotics and Automation Letters*, 7(4):9707–9714, 2022.

[3] Martin Büchner, Jannik Zürn, Ion-George Todoran, Abhinav Valada, and Wolfram Burgard. Learning and aggregating lane graphs for urban automated driving. In *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 13415–13424, 2023.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proc. of the Europ. Conf. on Computer Vision*, pages 213–229, 2020.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[6] Nikhil Gosala, Kürsat Petek, Paulo LJ Drews-Jr, Wolfram Burgard, and Abhinav Valada. Skyeye: Self-supervised bird's-eye-view semantic mapping using monocular frontal view images. In *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 14901–14910, 2023.

[7] Elias Greve, Martin Büchner, Niclas Vödisch, Wolfram Burgard, and Abhinav Valada. Collaborative dynamic 3d scene graphs for automated driving. *arXiv preprint arXiv:2309.06635*, 2023.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 770–778, 2016.

[9] Songtao He and Hari Balakrishnan. Lane-level street map extraction from aerial imagery. In *Proc. of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2080–2089, 2022.

[10] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[11] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: An online hd map construction and evaluation framework. In *Int. Conf. on Robotics & Automation*, pages 4628–4634, 2022.

[12] Bencheng Liao, Shaoyu Chen, Bo Jiang, Tianheng Cheng, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Lane graph as path: Continuity-preserving path-wise modeling for online lane graph construction. *arXiv preprint arXiv:2303.08815*, 2023.

[13] Bencheng Liao, Shaoyu Chen, Xinggang Wang, Tianheng Cheng, Qian Zhang, Wenyu Liu, and Chang Huang. MapTR: Structured modeling and learning for online vectorized HD map construction. In *Int. Conf. on Learning Representations*, 2023.

[14] Bencheng Liao, Shaoyu Chen, Yunchi Zhang, Bo Jiang, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Maptrv2: An end-to-end framework for online vectorized hd map construction. *arXiv preprint arXiv:2308.05736*, 2023.

[15] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. Dab-detr: Dynamic anchor boxes are better queries for detr. *arXiv preprint arXiv:2201.12329*, 2022.

[16] Rohit Mohan and Abhinav Valada. Perceiving the invisible: Proposal-free amodal panoptic segmentation. *IEEE Robotics and Automation Letters*, 7(4):9302–9309, 2022.

[17] Raphael Trumpp, Martin Büchner, Abhinav Valada, and Marco Caccamo. Efficient learning of urban driving policies using bird'view state representations. In *IEEE Int. Conf. on Intelligent Transportation Systems*, pages 4181–4186, 2023.

[18] Zelun Wang and Jyh-Charn Liu. Translating math formula images to latex sequences using deep neural networks with sequence-level training. *arXiv preprint arXiv:1908.11415*, 2019.

[19] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proc. of the IEEE Conf. on Comp. Vision and Pattern Recognition*, pages 2881–2890, 2017.

[20] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

[21] Jannik Zürn, Johan Vertens, and Wolfram Burgard. Lane graph estimation for scene understanding in urban driving. *IEEE Robotics and Automation Letters*, 6(4):8615–8622, 2021.

V. Appendix

In addition to the qualitative insights shown in the main manuscript, we provide a number of failure cases in Fig. 5.