An Initial Study of Bird's-Eye View Generation for Autonomous Vehicles using Cross-View Transformers

Felipe Carlos dos Santos[1], **Eric Aislan Antonelo**[1], **Gustavo Claudio Karl Couto**[1]

[1] Federal University of Santa Catarina (UFSC), Florianópolis, Brazil
Department of Automation and Systems Engineering

`felipe.carlos.santos@posgrad.ufsc.br,`

`eric.antonelo@ufsc.br, gustavo.karl.couto@posgrad.ufsc.br`

Abstract. *Bird's-Eye View (BEV) maps provide a structured, top-down abstraction that is crucial for autonomous-driving perception. In this work, we employ Cross-View Transformers (CVT) for learning to map camera images to three BEV's channels - road, lane markings, and planned trajectory - using a realistic simulator for urban driving. Our study examines generalization to unseen towns, the effect of different camera layouts, and two loss formulations (focal and L1). Using training data from only a town, a four-camera CVT trained with the L1 loss delivers the most robust test performance, evaluated in a new town. Overall, our results underscore CVT's promise for mapping camera inputs to reasonably accurate BEV maps.*

1. Introduction

Bird's-eye-view (BEV) representations have become a cornerstone of modern autonomous driving stacks, because they remove perspective distortions of ego-centric cameras and expose the geometry of the scene in a planner-friendly grid. Early convolutional pipelines such as *Lift-Splat-Shoot* [Philion and Fidler 2020] showed that dense BEV features learned directly from multi-camera images boost both map segmentation and motion forecasting, triggering a surge of BEV-centric perception research.

Rather than rely on hand-crafted inverse-perspective mapping or depth supervision, recent methods employ Transformers to learn the view transformation implicitly. *Cross-View Transformers (CVT)* [Park et al. 2023] introduced camera-aware attention that projects multi-view features into a unified map at real-time speed. Subsequent architectures—*BEVFormer* with spatio-temporal queries [Zhou et al. 2022], and the highly efficient *WidthFormer* [Liang et al. 2024] - have steadily advanced accuracy, latency and robustness on large-scale datasets such as nuScenes.

Dedicated lane- and drivable-area models now exploit decomposed or bidirectional cross-attention—e.g. *Efficient Lane Transformer* [Müller et al. 2023] and *BAE-Former* [Wang et al. 2023]—to overcome the geometric errors of naïve *Inverse Perspective Mapping (IPM)* and to capture long-range context.

Semantic-level BEV generation is also pivotal for closed-loop control. Imitation-learning frameworks such as *Learning by Cheating* [Codevilla et al. 2019] demonstrate that mid-level BEV inputs can substantially stabilize end-to-end driving in CARLA, but they consume oracle or high-fidelity maps; the impact of a *predicted*

BEV on downstream control is somewhat unexplored, with early attempts made in [Couto and Antonelo 2023]. Works in BEV generation can be broadly divided into two categories based on agent communication and sensor modality. The first category includes single-agent approaches, such as CVT, *SinBEV* [Xu et al. 2022], *Fiery* [Hu et al. 2021], and *VPN* [Pan et al. 2020]. The second category comprises multi-agent techniques that leverage vehicle-to-vehicle (V2V) communication to share information among agents, as seen in *CoBEVT* [Xu et al. 2022], *V2VNet* [Wang et al. 2020], and *CoBEVFusion* [Qiao et al. 2024]. In terms of sensor modality, BEV generation methods can be classified into those using only RGB cameras—such as CVT, Fiery and VPN—and those that rely on LiDAR or multi-modal inputs that combine various sensors, such as [Qiao et al. 2024] and [Wang et al. 2020].

A BEV route channel is used by an imitation learning agent in [Antonelo et al. 2024] to enable route-directed navigation. In [Couto and Antonelo 2023], BEV generation via a GAN is conditioned on the intended route, allowing a downstream controller to use this information for navigation in the city. Similarly, we include a sparse trajectory as an additional input channel, enabling the network to predict a BEV route channel, which can be used by navigation agents in future work.

This paper proposes the use of CVT in a single-vehicle approach that is based on camera input only, with training data from only one town. The exclusive use of RGB cameras is motivated by the potential for lower hardware costs. Moreover, single-agent methods are particularly relevant in transitional scenarios where autonomous vehicles are not yet widespread, thus contributing to the practical viability and scalability of autonomous driving technologies.

1.1. Contributions.

This paper fills these gaps by:

1. Adapting Cross-View Transformers to scenarios using the CARLA simulator, where it needs to map three frontal cameras and an optional backward view to three semantic channels (road, lanes and planned route). The generation of the planned route channel is particular to the current work, which was not covered in the literature as far as the authors know;
2. And benchmarking CVT against a strong UNet baseline under equivalent training protocols, analysing data efficiency, view-count scalability (how gracefully a model's performance changes when you add or remove camera views) and generalization to unseen towns.

2. Methods

2.1. Unet

Unet is a pure convolutional neural network, first introduced in [Ronneberger et al. 2015] for medical image segmentation. Unet has become one of the most widely used architectures. It consists of an encoder that reduces spatial resolution through successive convolutional and pooling layers, capturing high-level semantic features, followed by a decoder that recovers spatial details via transposed convolutions, enabling precise pixel-wise predictions. In addition, Unet is known for using skip connections, a technique that has substantially improved the training of very deep neural networks [Orhan and Pitkow 2017] by mitigating the vanishing gradient problem.

While this architecture is originally designed for single-image input, our approach extends its use by concatenating multiple camera views alongside a trajectory representation. Although the network lacks an explicit cross-view fusion mechanism, this input configuration enables the CNN to jointly process information from all views, allowing it to generate a coherent semantic segmentation in the BEV view.

2.2. Cross-view transformer (CVT)

Following the success of the Transformer architecture introduced by [Vaswani et al. 2017] in natural language processing, [Dosovitskiy et al. 2020] demonstrated how the attention mechanism could be effectively adapted to process visual data, laying the groundwork for Vision Transformers. Building upon these concepts, [Park et al. 2023] introduced a method that assigns a positional encoding to each individual camera based on its intrinsic and extrinsic parameters, enabling the fusion of multi-camera inputs into a unified map-view representation.

This model processes a collection of n monocular views $\{(I_k, K_k, R_k, t_k)\}_{k=1}^{n}$, which include the input image $I_k \in R^{H \times W \times 3}$, camera intrinsic parameters $K_k \in R^{3 \times 3}$, along with the extrinsic rotation matrix $R_k \in R^{3 \times 3}$ and translation vector $t_k \in R^3$, all referenced to the ego-vehicle's center.

2.2.1. Cross-view attention

Given a real-world coordinate $\mathbf{x}^{(W)} \in R^3$, the perspective transformation defines its corresponding image-plane coordinate $\mathbf{x}^{(I)} \in R^3$:

$$\mathbf{x}^{(I)} \sim K_k R_k (\mathbf{x}^{(W)} - \mathbf{t}_k). \tag{1}$$

Using the image coordinates from Equation 1, it is possible to rephrase the geometric relationship between real world and image coordinates as a cosine similarity, suitable for use in an attention mechanism, as follows:

$$\text{sim}_k(\mathbf{x}^{(I)}, \mathbf{x}^{(W)}) = \frac{(R_k^{-1} K_k^{-1} \mathbf{x}^{(I)}) \cdot (\mathbf{x}^{(W)} - \mathbf{t}_k)}{\|R_k^{-1} K_k^{-1} \mathbf{x}^{(I)}\| \cdot \|\mathbf{x}^{(W)} - \mathbf{t}_k\|}. \tag{2}$$

This similarity still depends on the exact world coordinate $\mathbf{x}^{(W)}$. To address this, the geometric components is replaced with positional encoding capable of capturing both geometric and appearance features.

2.2.2. Camera-aware positional encoding

Starting from the unprojected coordinate in the image world $\mathbf{d}_{k,i} = R_k^{-1} K_k^{-1} \mathbf{x}_i^{(I)}$ for each image coordinate $\mathbf{x}_i^{(I)}$.

The vector $\mathbf{d}_{k,i}$ is encoded through a linear layer using a multilayer perceptron (MLP), shared across all k views, to obtain a D-dimensional positional embedding.

The CVT architecture combines this embedding with the image features $\phi_{k,i}$ in the keys of the cross-view attention mechanism. This enables the attention module to perceive both appearance and geometric cues to establish correspondences across multiple views.

2.2.3. Overall architecture

The Figure 1 shows the complete CVT architecture, each image I_i is passed through a feature extractor (EfficientNet-B4 [Tan and Le 2019]) to produce a multi-resolution patch embedding $\{\phi_i^1, \phi_i^2, \ldots, \phi_i^R\}$, where R is the number of resolutions considered, following [Park et al. 2023] number of R used is 2. Then, the map-view representation is progressively refined and the process is repeated at higher resolutions. Finally, the decoder upsamples the representation to produce the final prediction.

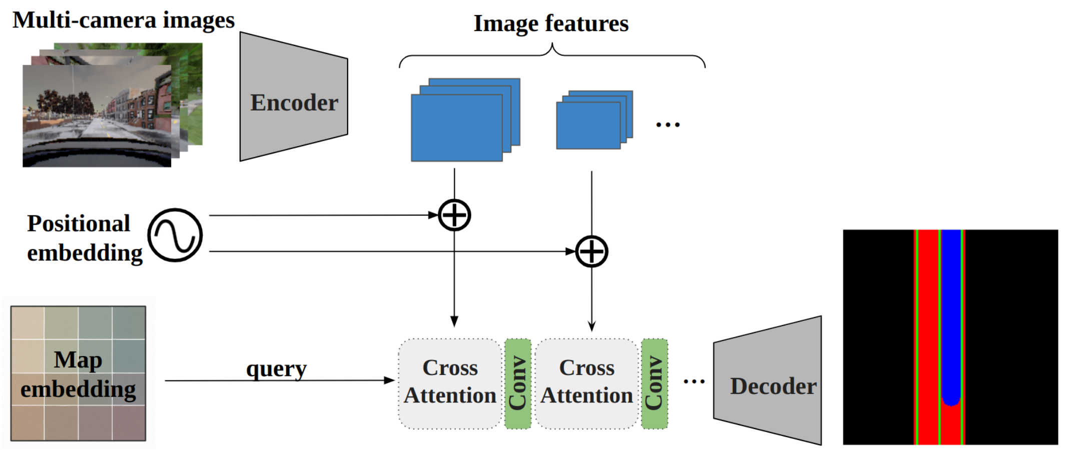


Figure 1. Architecture of the Cross-View Transformer (CVT). Multi-view images are processed through EfficientNet-B4 to extract multi-scale image features. Camera-aware positional encodings and cross-view attention fuse geometric and visual information, enabling iterative refinement of the BEV representation in the latent space.

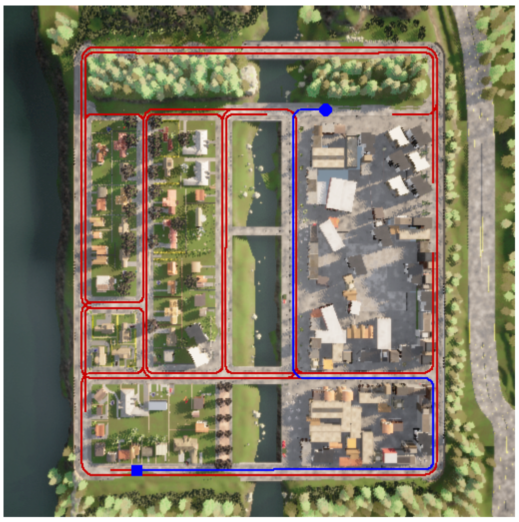
3. Experiments

In this section, we compare the Cross-View Transformer architecture with the Unet model, and evaluate the performance of two loss functions: L1 loss and Focal Loss [Lin et al. 2017]. Additionally, for the CVT, we compare configurations using four cameras and three cameras.

3.1. Dataset Generation

The environment and trajectories are sourced from the CARLA Leaderboard evaluation platform [lea 2020]. Specifically, from *Town01* we build the training and validation sets, while from *Town02* we create the test set. Each town provides ten different routes.

The test *Town02* is shown in Figure 2(b), while Figure 3 shows the frames captured in that city at a particular instant by all the four cameras of the vehicle along with a sparse trajectory, totaling five input channels that feed the learning model. This input was later used to produce the inferences shown in Figure 5. For the three-camera experiment, the rear camera is removed to analyze the model's behavior under limited information from the surroundings.


(a)

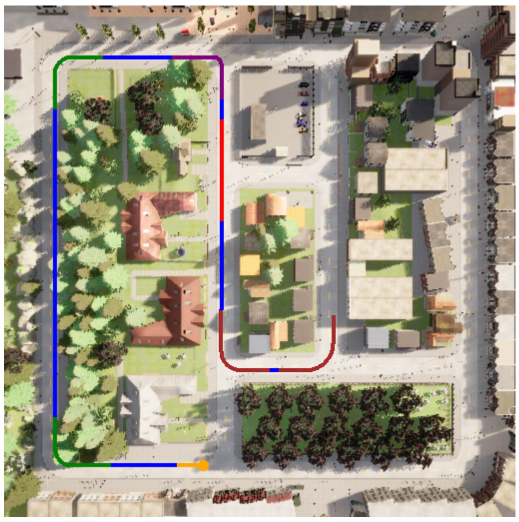

(b)

Figure 2. (a) CARLA Leaderboard *Town01*, showing training routes in red and the validation route in blue. The starting point is marked with a circle, and the endpoint is indicated with a square. (b) Test environment *Town02*, with colored lines indicating different segments of *route00*.

Figure 2(a) shows the *Town01* environment, from which the nine training routes (red) and the validation route (blue) were obtained. On the other hand, Figure 2(b) presents *Town02*, used as the test environment, where the colored lines indicate segments of *route00*.

CVT requires the use of both intrinsic and extrinsic camera matrices, as discussed in Section 2.2, but CARLA does not provide a straightforward way to retrieve them directly. Therefore, based on the camera configuration parameters, the intrinsic matrix used in this work is defined as follows. The focal lengths f_x and f_y are computed from the camera's field of view (FOV)—the angle through which the camera is capable of seeing the world—using the following equations. The image center coordinates c_x and c_y are approximated as $c_x = \frac{\text{width}}{2}$ and $c_y = \frac{\text{height}}{2}$:

$$f_x = \frac{\text{width}}{2 \cdot \tan\left(\frac{\text{FOV}_{\text{rad}}}{2}\right)}, \quad f_y = \frac{\text{height}}{2 \cdot \tan\left(\frac{\text{FOV}_{\text{rad}}}{2}\right)} \tag{3}$$

Given these values, the intrinsic matrix K from Section 2.2 is defined as:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{4}$$

The extrinsic matrix is shown in Equation 5, where $R \in \mathcal{R}^{3\times3}$ is the rotation matrix calculated from the *pitch*, *yaw*, and *roll* parameters used to define the camera, and

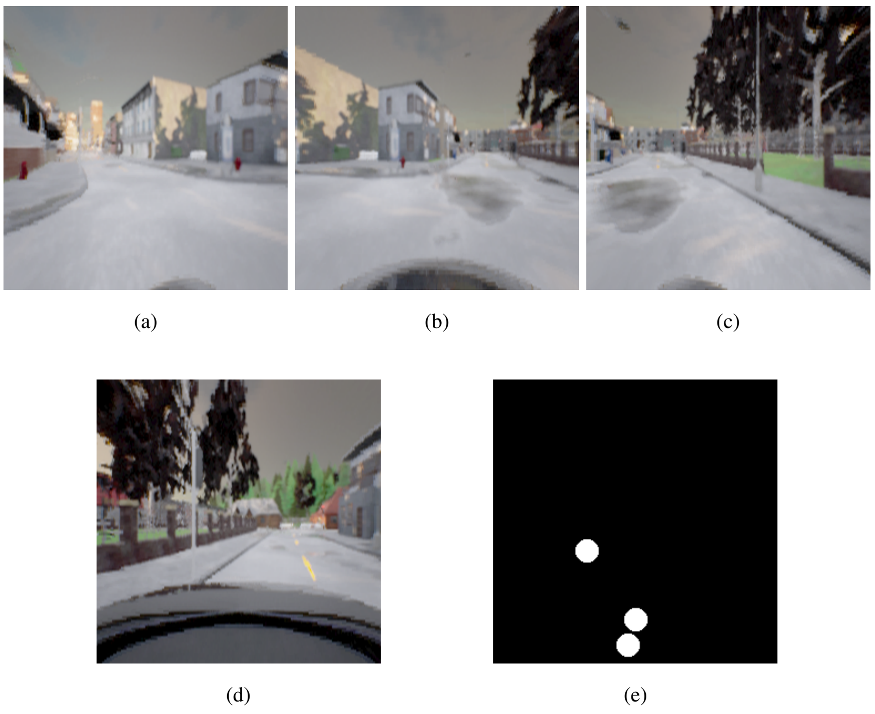


Figure 3. Example of model input captured in *Town02*: (a) Left camera, (b) Central camera, (c) Right camera, (d) Rear camera, (e) Sparse trajectory.

$t \in R^{3\times1}$ is the position x, y, z, which also defines the camera in the simulator. Both R and t are the same matrices discussed in Section 2.2.

$$\mathbf{E} = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \tag{5}$$

In total, the dataset comprises 233,315 images and 46,663 sample points, resulting in approximately 9.3 GB of uncompressed data. The distribution across the training, validation, and test sets is as follows:

- **Training:** 96,435 images and 19,287 sample points, totaling approximately 3.8 GB.
- **Validation:** 10,000 images and 2,000 sample points, totaling approximately 0.3 GB.
- **Test:** 126,880 images and 25,376 sample points, totaling approximately 5.1 GB.

3.2. Training

Both Unet and CVT architectures were trained using 19,287 data points from *routes 01* to *09* in *Town01*, while *route00* was reserved for validation during training. All networks presented in this paper were trained for 50 epochs[1] using the same learning rate configuration. For the sake of simplicity, only the training losses for four cameras using focal loss are presented. Figure 4 shows the training metrics for CVT and UNet, both

[1]The setting of 50 training epochs seamed reasonable enough to compare all the models, given the limited computational resources. Longer training periods with early stopping will be used in future works.

trained with focal loss and using four input cameras. Although the training loss for CVT is higher than that of UNet, the validation loss on *route00* is 156.15% higher for UNet. This result suggests that the transformer-based architecture has a stronger generalization capability, providing initial evidence of its superior performance. performance on unseen input frames.

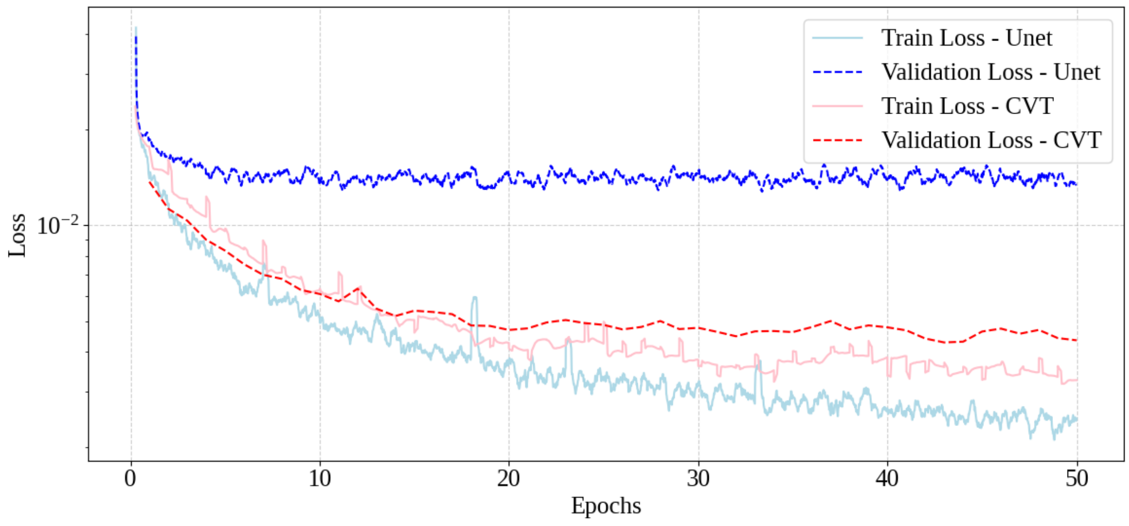


Figure 4. Training and validation losses per epoch (*Town01*) for both CVT and Unet using 4 cameras and focal loss.

In this work, six different models were trained: four based on the CVT architecture, using permutations of three (no rear view) and four input cameras combined with either focal loss or L1 loss; and two based on Unet, which serve as baselines, both using four cameras—one trained with focal loss and the other with L1 loss.

3.3. Inference visualization

Figure 5 presents the inference results at a T-intersection in *Town02* for each of the models. Note that the models were trained only in *Town01*, and that *Town02* is used as an test town.

From Figure 5, it can be observed that, as expected, the CVT architecture demonstrates superior BEV generation compared to Unet. However, the addition of the rear camera appears to introduce additional complexity into the model: without training it longer (all networks were trained for 50 epochs) neither with more data points, the performance of the models with 4 cameras decays compared to the its counterpart of 3 cameras. The longer training of these models, and with more data, is left as future work.

The combination of CVT with L1 loss shows promising results, particularly in the road channel, suggesting that this configuration may better capture the structural features of the environment. In the next subsection, we quantitatively analyze the model inferences in both *Town01* and *Town02*, as well as investigate how the models perform across different route segments.

From Figure 5 its possible to infer CVT outperforms UNet. The combination of CVT with L1 loss yields the best qualitative overall result. This figure also allows

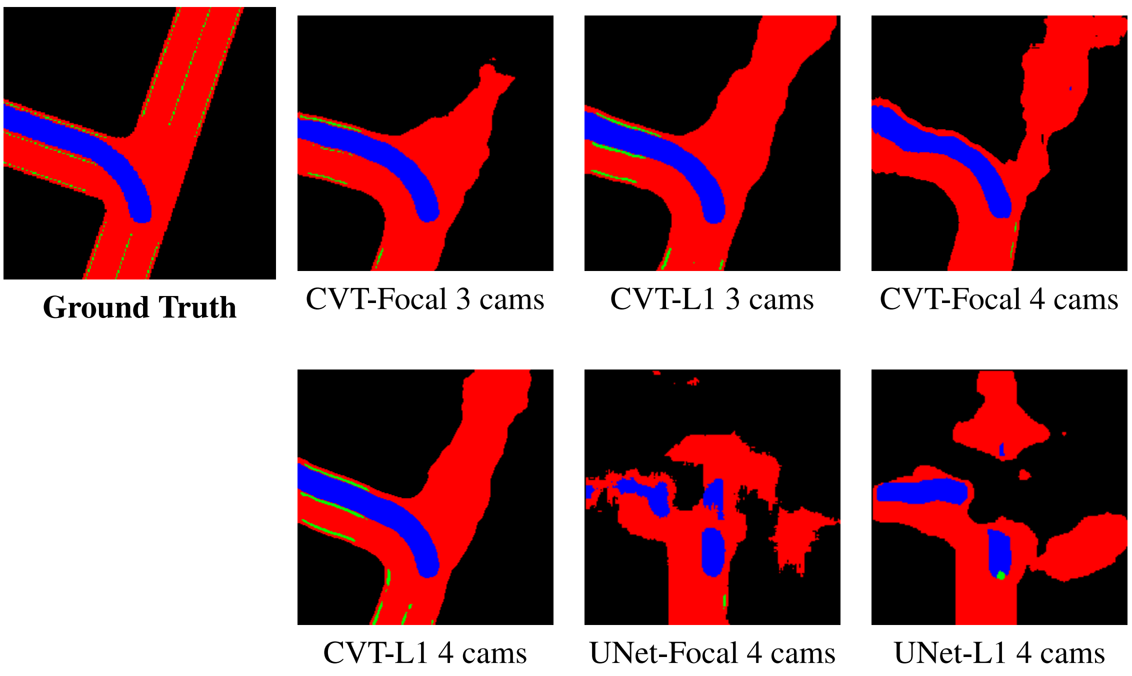


Figure 5. Model inference results across all evaluated methods. The associated input images that served as input to these models are provided in Figure 3.

for a comparison between configurations with and without the rear camera. While the addition of the rear camera improves lane segmentation for the L1 loss configuration, this improvement is not observed with focal loss.

Given the similar inferences observed in Figure 5, Table 1 presents the individual channel performances for these inferences. The table also shows that adding a rear camera in the focal loss configuration significantly improved the trajectory channel performance for this sample point.

Figure 6 shows different trajectories for the CVT model with 4 cameras, trained using L1 loss. Figure 6(a) displays a straight segment—the most common type—and the model performs very well in this case. Figures 6(b), (c), and (d) present less frequent segments, where the model achieves relative success but struggles to segment the lane channel and blind spot regions.

Table 1. mIoU per channel for the CVTs models from Figure 5 inferences

Model	Road	Trajectory	Lane
CVT-L1 4 cams	0.8188	0.6536	**0.0355**
CVT-L1 3 cams	**0.8346**	0.6890	0.0224
CVT-Focal 4 cams	0.7580	**0.8051**	0.0067
CVT-Focal 3 cams	0.6950	0.6709	0.0174

3.4. Evaluation results

To evaluate the trained models, two tables were generated, each comparing the mean IoU for every output channel across the different configurations. Table 2 presents the results

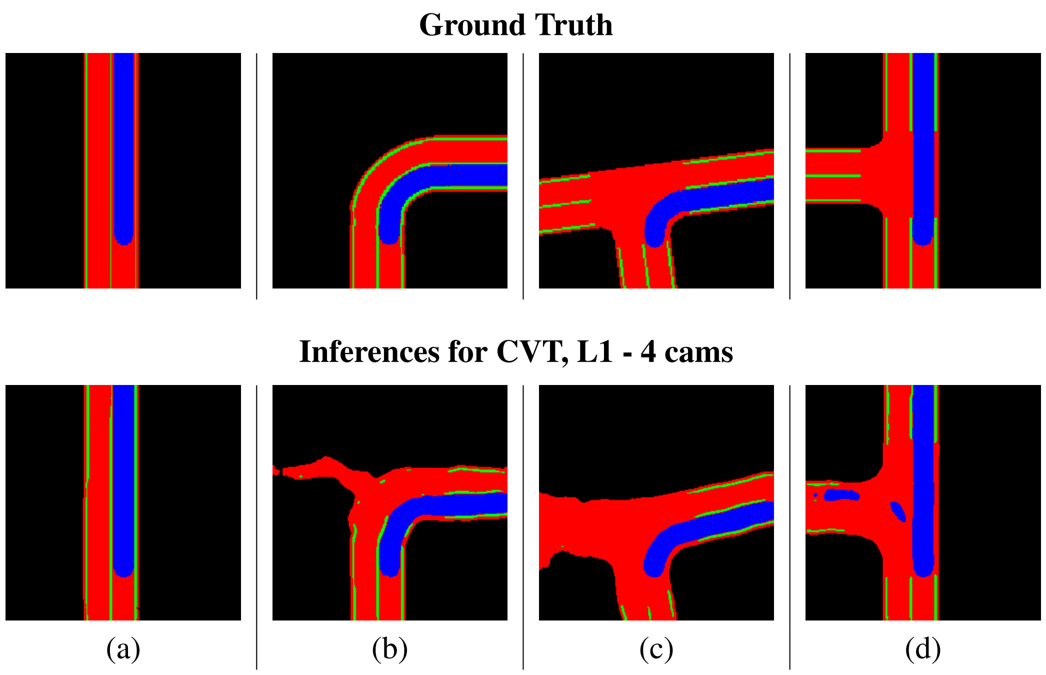


Figure 6. Model inferences for CVT 4 cameras trained using L1 loss, results across different locations in the test dataset - *Town02*.

on 2,000 data points from the validation route, *route00*, in *Town01*, while Table 3 shows the performance across all routes in *Town02*, encompassing 25,376 data points. This comparison provides insights on how well the models generalize to environments and scenarios not seen during training.

Table 2. Mean IoU per channel for different models from *Town01* - Validation route

Model	Road	Trajectory	Lane
CVT, Focal loss - 4 cams	0.9659	0.8650	0.4013
Unet, Focal loss - 4 cams	0.9072	0.7823	**0.6972**
CVT, L1 - 4 cams	**0.9731**	0.8563	0.4265
Unet, L1 - 4 cams	0.8863	0.7759	0.6591
CVT, Focal loss - 3 cams	0.9676	**0.9028**	0.4321
CVT, L1 - 3 cams	0.9593	0.8959	0.4317

Table 2 shows that the CVT models demonstrate superior performance in representing both the road and trajectory channels when compared to the Unet baseline. On the other hand, the Unet model trained with focal loss achieves significantly better results in the lane channel, with an IoU around 60% higher than the best-performing CVT model in that category. For the road channel, the best model was the CVT with four cameras and L1 loss, but the three cameras model was slightly behind. However, for the trajectory channel, the addition of the rear camera seems to have negatively impacted performance, with the three-camera CVT variants outperforming their four-camera counterparts.

Table 3 presents the results for *Town02*, where the trends observed in *Town01* are largely maintained. The addition of the rear camera still slightly improves the perfor-

Table 3. Per-channel mIoU scores for all evaluated models, tested on all *Town02* ten routes.

Model	Road	Trajectory	Lane
CVT, Focal loss - 4 cams	0.9079	0.7528	0.2906
Unet, Focal loss - 4 cams	0.6978	0.5915	**0.3959**
CVT, L1 - 4 cams	**0.9144**	0.7808	0.3158
Unet, L1 4 - cams	0.6804	0.5642	0.3809
CVT, Focal loss - 3 cams	0.8981	0.7787	0.2970
CVT, L1 - 3 cams	0.8823	**0.7935**	0.3099

mance for the road channel. For the trajectory channel, the top result was achieved by the CVT model with three cameras and L1 loss, closely followed by its four-camera counterpart. This suggests a slight advantage of L1 loss over focal loss in terms of generalization to unseen data. Notably, the same configuration also achieved the best CVT performance in the lane channel.

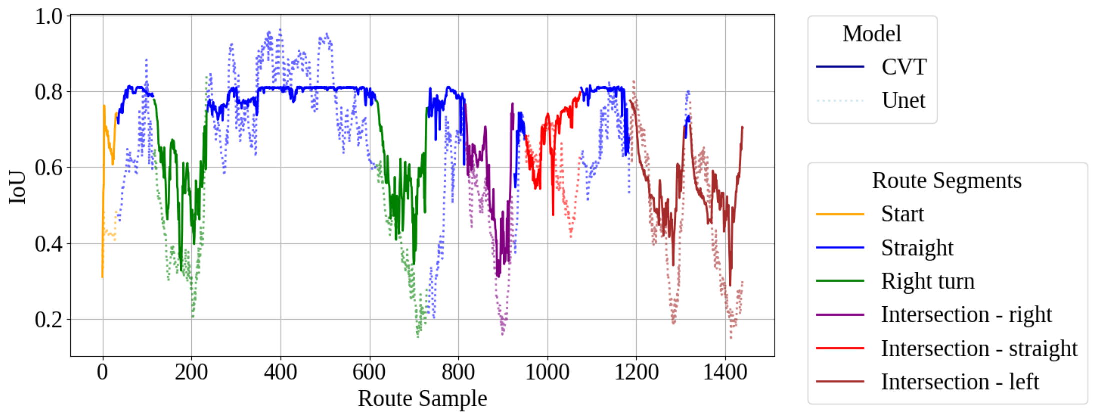


Figure 7. IoU for CVT and Unet predictions during part of the *route00* from *Town02*. For the colored segments of the Figure 2(b).

As for the overall performance in the lane channel, the advantage still lies with the Unet models. One possible explanation is that these models perform better on straight-line segments. Considering that most of the routes are composed of straight trajectories, the mean IoU for Unet in this channel remains higher than that of the CVT models, which appear to struggle more with lane segmentation. This behavior is better illustrated in Figure 7.

Figure 7 presents the mean IoU for each data point along the segments, using colors that match those in Figure 2(b), enabling a comparison between CVT and UNet across different road sections. The transformer-based architecture demonstrates superior performance in turns and intersections. On the other hand, it appears to saturate around 0.8 on straight paths, while Unet occasionally surpasses this threshold. This suggests that Unet may struggle to infer from rarely seen situations but is more effective at reproducing familiar patterns. Otherwise, CVT shows a stronger ability to generalize and extract meaningful representations in more complex or uncommon scenarios, characteristic more aligned with the requirements of the BEV generation task.

4. Conclusion

This study investigates Bird's-Eye View (BEV) generation for autonomous vehicles in the CARLA simulator using Cross-View Transformers (CVT). We included the trajectory channel to be generated, which is not covered in other works from the literature as far as the authors know. With this additional channel, an end-to-end imitation learning agent can be trained conditioned on the route to follow [Antonelo et al. 2024]. Furthermore, the results demonstrate that the CVT architecture significantly outperforms the UNet baseline in terms of generalization, particularly in complex scenarios such as intersections and turns, especially for road and trajectory channels. A comparative analysis of loss functions revealed that CVT trained with L1 loss offers greater robustness in unseen environments, while UNet, though less adaptable, showed an advantage in lane segmentation and performance on straight road segments.

The three-camera frontal configuration proved sufficient for most tasks in this work, reducing reliance on rear sensors without compromising performance. This suggests that autonomous systems in early deployment stages can benefit from data- and hardware-efficient solutions. However, lane segmentation remains a challenge for CVT models in the current setup.

This work advances BEV perception research using the CARLA simulator by validating the performance of transformer-based models. However, further research should be done in training the models longer, with more data, and using resampling techniques, specially the ones with four cameras. For instance, data resampling could improve the performance of CVT models by weighting more the less frequent observations. Future work should also explore the direct integration of generated BEV representations into autonomous control pipelines, as well as the inclusion of more complex semantic channels such as pedestrians, other vehicles and traffic lights.

References

(2020). Carla autonomous driving leaderboard. https://leaderboard.carla.org/.

Antonelo, E. A., Couto, G. C. K., Möller, C., and Fernandes, P. H. (2024). Investigating behavior cloning from few demonstrations for autonomous driving based on bird's-eye view in simulated cities. In *BRACIS*, pages 155–168. Springer.

Codevilla, F., Müller, M., Dosovitskiy, A., López, A., and Koltun, V. (2019). End-to-end driving via conditional imitation learning. In *ICRA*.

Couto, G. C. K. and Antonelo, E. A. (2023). Hierarchical generative adversarial imitation learning with mid-level input generation for autonomous driving on urban environments. *arXiv preprint arXiv:2302.04823*.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Hu, A., Murez, Z., Mohan, N., Dudas, S., Hawke, J., Badrinarayanan, V., Cipolla, R., and Kendall, A. (2021). Fiery: Future instance prediction in bird's-eye view from surround monocular cameras. In *Proceedings of the IEEE/CVF ICCV*, pages 15273–15282.

Liang, C., Xiao, M., Wang, Y., et al. (2024). Widthformer: Fast wide-field bev perception. In *IEEE/CVF CVPR*.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE ICCV*, pages 2980–2988.

Müller, J., Schneider, L., and Dengel, A. (2023). Efficient lane transformer for bird's-eye-view lane detection. In *IEEE Intelligent Vehicles Symposium (IV)*.

Orhan, A. E. and Pitkow, X. (2017). Skip connections eliminate singularities. *arXiv preprint arXiv:1701.09175*.

Pan, B., Sun, J., Leung, H. Y. T., Andonian, A., and Zhou, B. (2020). Cross-view semantic segmentation for sensing surroundings. *IEEE RA-L*, 5(3):4867–4873.

Park, S., Hong, S., Kim, S., and Lee, K. (2023). Cross-view transformers for real-time map-view semantic segmentation. In *IEEE/CVF CVPR*.

Philion, J. and Fidler, S. (2020). Lift, splat, shoot: A multicamera bird's-eye view for 3d object detection. In *European Conference on Computer Vision (ECCV)*.

Qiao, D., Zulkernine, F., and Anand, A. (2024). Cobevfusion cooperative perception with lidar-camera bird's eye view fusion. In *2024 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pages 389–396. IEEE.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *MICCAI 2015, proceedings, part III 18*, pages 234–241. Springer.

Tan, M. and Le, Q. (2019). Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114. PMLR.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.

Wang, K., Luo, J., Hu, X., et al. (2023). Baeformer: Bidirectional aggregation of cross-view features for bev lane segmentation. In *IEEE ITSC*.

Wang, T.-H., Manivasagam, S., Liang, M., Yang, B., Zeng, W., and Urtasun, R. (2020). V2vnet: Vehicle-to-vehicle communication for joint perception and prediction. In *ECCV 2020 proceedings, part II 16*, pages 605–621. Springer.

Xu, R., Tu, Z., Xiang, H., Shao, W., Zhou, B., and Ma, J. (2022). Cobevt: Cooperative bird's eye view semantic segmentation with sparse transformers. *arXiv preprint arXiv:2207.02202*.

Zhou, T., Fang, L., Chen, Z., et al. (2022). Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV*.