Towards Viewpoint Robustness in Bird's Eye View Segmentation

Tzofi Klinghoffer[1,2*] Jonah Philion[2,3,4] Wenzheng Chen[2,3,4] Or Litany[2]
Zan Gojcic[2] Jungseock Joo[2,5] Ramesh Raskar[1] Sanja Fidler[2,3,4] Jose M. Alvarez[2†]
[1]MIT [2]NVIDIA [3]University of Toronto [4]Vector Institute [5]UCLA

Project page: https://nvlabs.github.io/viewpoint-robustness

Abstract

Autonomous vehicles (AV) require that neural networks used for perception be robust to different viewpoints if they are to be deployed across many types of vehicles without the repeated cost of data collection and labeling for each. AV companies typically focus on collecting data from diverse scenarios and locations, but not camera rig configurations, due to cost. As a result, only a small number of rig variations exist across most fleets. In this paper, we study how AV perception models are affected by changes in camera viewpoint and propose a way to scale them across vehicle types without repeated data collection and labeling. Using bird's eye view (BEV) segmentation as a motivating task, we find through extensive experiments that existing perception models are surprisingly sensitive to changes in camera viewpoint. When trained with data from one camera rig, small changes to pitch, yaw, depth, or height of the camera at inference time lead to large drops in performance. We introduce a technique for novel view synthesis and use it to transform collected data to the viewpoint of target rigs, allowing us to train BEV segmentation models for diverse target rigs without any additional data collection or labeling cost. To analyze the impact of viewpoint changes, we leverage synthetic data to mitigate other gaps (content, ISP, etc). Our approach is then trained on real data and evaluated on synthetic data, enabling evaluation on diverse target rigs. We release all data for use in future work. Our method is able to recover an average of 14.7% of the IoU that is otherwise lost when deploying to new rigs.

1. Introduction

Neural networks (NNs) are becoming ubiquitous across domains. Safety critical applications, such as autonomous vehicles (AVs), rely on these NN to be robust to out of distribution (OOD) data. Yet, recent work has drawn attention to the susceptibility of NNs to failure when exposed to OOD data, such as adversarial corruptions [10], unseen weather conditions [15], and new geographic regions [6]. While each of these pose a significant challenge for safety critical applications, we focus on another distribution shift, which, thus far, has been understudied in the research literature – changes in camera viewpoint between train data and test data. Because camera viewpoint changes are realistic in AVs, we study their impact on AV perception tasks.

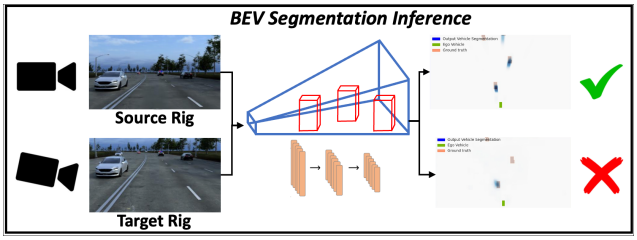

Figure 1. **Impact of Changed Camera Viewpoint:** We find that the performance of state-of-the-art methods for bird's eye view (BEV) segmentation quickly drop with small changes to viewpoint at inference. Above we see predictions from Cross View Transformers [29] trained on data from a source rig (top). The target rig pitch is reduced by 10° (bottom), leading a 17% drop in IoU.

AVs use cameras around the ego-vehicle to perceive their surroundings. Using images from each camera, NNs detect and segment objects in the scene, such as vehicles, pedestrians, roads, and more. This information is used by trajectory planners to decide how the ego-vehicle navigates. Camera viewpoint for AVs may differ between train and test in several real-world scenarios. First, the camera viewpoint may change over time due to wear and tear or damage. Second, camera viewpoint may change due to installation variation. Third, and most relevant for our work, if a single NN is to be deployed across different types of vehicles, it must be able to generalize to the camera viewpoints of each car. Collecting and labeling train data for each target rig is not scalable and quickly becomes intractable for AV companies wishing to scale across many types of vehicles due to cost, thus motivating our work to transform collected data into the viewpoint of diverse target rigs to use for training.

The goal of this paper is to bring understanding and a

*Work done during an internship at NVIDIA.
†Corresponding author: Jose M. Alvarez (josea@nvidia.com).

first approach to a real-world problem in the AV space that has yet to receive attention in the research literature – generalization from a source to target camera rig. We focus on bird's eye view (BEV) segmentation from RGB data to motivate how changing camera viewpoint can affect AV perception models. We study this problem by conducting an in-depth analysis on the impact changing the camera viewpoint at inference time has on recent BEV segmentation models. Our findings indicate that even small changes in camera placement at inference time degrade BEV segmentation accuracy, as illustrated in Fig. 1. We then propose a method to improve generalization to a target rig by simulating views in the target perspective. We show that incorporating data generated from novel view synthesis into training can significantly reduce the viewpoint domain gap, bringing the BEV segmentation model to the same level of accuracy as when there is no change in camera viewpoint, without having to collect or label any additional data. We compare our approach with other strategies, such as augmenting the camera extrinsics and labels during training, and find that our approach leads to better accuracy. Little work has focused on the impact of viewpoint changes for AV perception, and, to the best of our knowledge, we are the first to study the impact of diverse camera viewpoint changes on 3D AV perception tasks, such as BEV segmentation. We hope that this paper will encourage more research on the important problem of *viewpoint robustness* in AV.

Our paper makes the following contributions:

- We highlight the understudied problem of *viewpoint robustness* in bird's eye view segmentation for autonomous vehicles (AV) through an in-depth analysis revealing that recent models fail to generalize to different camera viewpoints at inference time.
- We propose a viewpoint augmentation framework for AV; we develop a novel view synthesis method that can be used to transform training data to target viewpoints and show that it improves the robustness of bird's eye view segmentation models to viewpoint changes.
- We provide datasets that can be used to benchmark future work on viewpoint robustness in AV.

Because real-world AV datasets from a diverse set of camera rigs are not publicly available, we use simulated data both for (1) training and evaluation in our analysis and (2) evaluation of our proposed technique. Our synthetic datasets can be used for future efforts to benchmark the generalization abilities of different AV perception methods to viewpoint changes. Datasets are publicly available on our project page. The first dataset, rendered from CARLA, consists of both training and testing data, allowing for isolated analysis of the impact of viewpoint changes on BEV segmentation models (example images in Fig. 2). The second dataset, rendered with NVIDIA DRIVE Sim [20], is significantly more photorealistic and consists of test sets from a diverse set of camera viewpoints. Thus, it can be used to evaluate models trained on real data, as we show in Section 5. Both datasets include 3D bounding box labels.

2. Related Work

2.1. Viewpoint Robustness

Recent work has drawn attention to the susceptibility of NNs to misclassify when presented with distributions not seen during training. Madan *et al.* [14] show that both convolutional- and transformer-based classifiers are fooled by small viewpoint changes, and they introduce a search strategy for finding adversarial viewpoints, which leads to misclassifications over 71% of the time. Similarly, [13] shows that small viewpoint changes degrade classification performance, especially when paired with out of distribution (OOD) categories, and demonstrates that increasing the diversity of training data is an effective strategy to mitigate this issue. Do *et al.* [4] use homography to move images closer to the distribution of training data at inference time. Coors *et al.* [3] study the impact of viewpoint changes for 2D semantic segmentation for AV, but do not explore 3D tasks. In contrast, we focus on providing a thorough analysis and a solution to the problem of viewpoint robustness for 3D AV perception tasks, focusing on BEV segmentation.

2.2. Novel View Synthesis

Novel view synthesis (NVS) provides a way to render images from unseen viewpoints of a scene, and thus could be used to improve the robustness of perception models to viewpoint changes. Many methods have been proposed for NVS in recent years [18, 16, 19], many of which are based on Neural Radiance Fields (NeRF) [17]. However, NeRF still faces two challenges that limit its applicability for our use case: (1) getting NeRF to generalize to dynamic scenes, which are common in AV, is an open research problem, and while there is promising work in this direction [28, 22], the setup is often too constrained and simplified to fit the AV problem setting, and (2) NeRF is challenging to scale due to lack of generalizability, so multiple NeRFs must be trained to perform NVS across scenes. While there is work aimed at generalizing NeRF [12], it remains an open problem and current methods are often constrained. Other methods for NVS rely on monocular depth estimation and can generalize across scenes when the depth estimation network is trained on diverse data. We leverage Worldsheet in our work [11], which is described in more detail in Sec. 4.1.

2.3. Bird's Eye View Segmentation

Bird's eye view (BEV) segmentation — the task of segmenting a scene in the top-down view (BEV) from 2D im-

Figure 2. **Datasets rendered in CARLA across viewpoints:** For the analysis part of our work, we use CARLA to simulate different viewpoints. We rendered datasets from a total of 36 viewpoints, a few of which are highlighted above, including the source rig (extrinsics from nuScenes [2] dataset), +12° yaw, +12° pitch, +21 inch height, and -12° pitch and +18 inch height together.

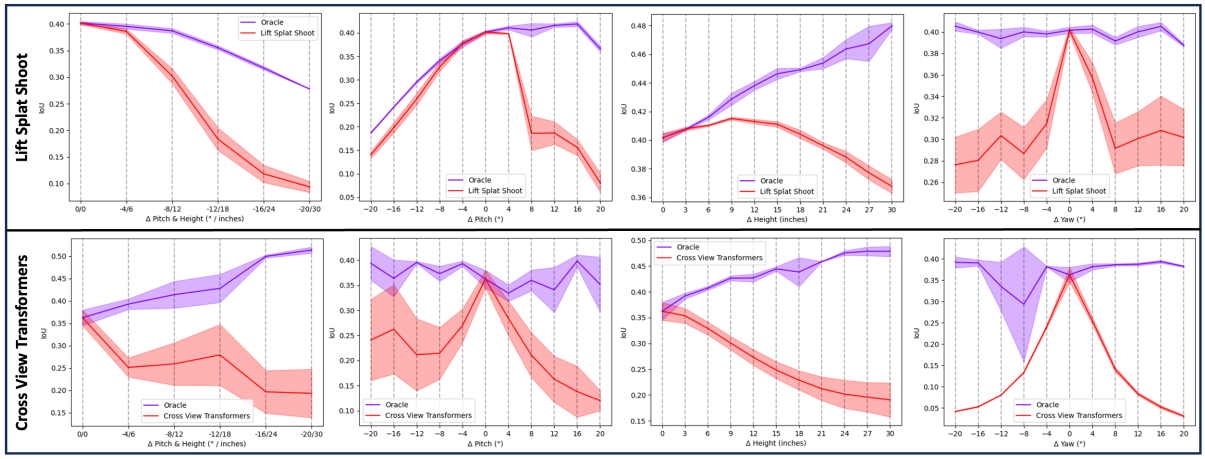

Figure 3. **Analysis of impact of viewpoint changes in CARLA:** We train a source BEV model using Lift Splat Shoot (LSS) [21] and Cross View Transformers (CVT) [29], denoted at point 0 on the x axis of each graph. We then test the model across different target rigs where the camera pitch, yaw, height, or pitch and height are changed, as denoted by the different points along the x axes. We also trained each model on the target rig directly and refer to this model as the "oracle", as it reflects the expected upper bound IoU for each viewpoint.

ages — is a useful task for benchmarking AV perception [21, 24, 1]. BEV segmentation requires a 2D to 3D unprojection to predict the position of objects surrounding the ego-vehicle from the BEV perspective. BEV segmentation models usually consist of an image encoder, which extracts the features from images from the camera rig, and a decoder, which uses the image features to predict the objects of interest in the BEV coordinate frame. Existing methods condition on the extrinsics and intrinsics of each camera in different ways. Lift-Splat-Shoot (LSS) [21] and Orthographic Feature Transform (OFT) [25] unproject features into a point cloud according to each camera's intrinsic and extrinsic parameters. LSS performs sum pooling along each pillar in the map-view, while OFT performs average pooling. Other methods, such as Cross View Transformers (CVT) [29], treat camera intrinsics and extrinsics as a feature, rather than explicitly unprojecting. We use LSS and CVT to conduct benchmarks, since these two methods encompass both convolutional and transformer-based architectures and explicit and implicit geometric representations.

3. Measuring the Impact of Camera Viewpoint Variations on BEV Segmentation

Method: In this section, we introduce our approach and results for measuring the impact of changing the camera viewpoint at inference time for BEV segmentation models trained on a single, source rig. We use simulated data from CARLA [5] for this analysis for two reasons: (1) using simulated data allows us to isolate the domain gaps between training and testing such that only camera viewpoint changes, and (2) real AV datasets with large differences in camera position are not publicly available. Examples of different camera viewpoints rendered in CARLA are shown in Fig. 2. For simplicity and ease of interpretation of our results, we conduct all experiments on a single camera rig, containing a front facing camera, which we refer to as the source rig. We first train a BEV segmentation model on data rendered from the source rig. For this rig, we use the camera parameters of sessions from the nuScenes dataset [2]. Then, we render train and test datasets from different target rigs, which contain variations to the yaw, pitch, height, or pitch and height of the camera. The train datasets are used to train an oracle for each target rig, while the test datasets are used to evaluate the model trained on the source rig in comparison to the oracle. For completeness, we sweep over a large range of each extrinsic and render a train and test dataset on regular intervals. For pitch and yaw, we sweep from -20° to 20°, rendering a dataset every 4°. For height, we sweep from 0 in to 30 in, rendering a dataset every 3 in. For height and pitch together, we sweep from 0° and 0 in to

-20° and 30 in, rendering a dataset at every -4° and 6 in.

To understand the domain gap introduced by changes in camera position, we test the "source model", which is the model trained on data from the source rig, across each test dataset, where each test dataset contains changes to either yaw, pitch, height, or pitch and height together. We then compare the test accuracy of the source model to the test accuracy of the oracle model, which was only trained on data from the target rig. The oracle model serves as an upper bound on model performance since there is no domain gap between the train and test datasets.

Model Details: We conduct our analysis using two BEV segmentation models, Lift Splat Shoot (LSS) and Cross View Transformers (CVT). LSS uses an explicit geometric operation to map objects in the camera coordinate system to the bird's eye coordinate system. It does this by using each camera's intrinsic and extrinsic parameters to construct a frustum shaped point cloud per camera where predicted objects are placed inside. A convolutional encoder maps images to features and depths, which are unprojected into the frustum, and a cumulative summing operation is done over the features in the vertical pillars of the frustum before the decoder then predicts the final segmentation. In contrast, CVT uses a transformer to learn features over images, extrinsics, and intrinsics. The extrinsic and intrinsic parameters are used to condition the segmentation network, such that it implicitly learns correlations between the parameters and positions of objects relative to the ego-vehicle. We use these two architectures because they cover both explicit and implicit geometric representations and convolutional and transformer backbones, allowing us to test the impact of each on generalization to viewpoint changes.

Results: Results of our analysis are shown in Fig. 3. We see that the performance of both LSS and CVT suffers drastically with even small changes to camera viewpoint, whether it be pitch, yaw, height, or pitch and height together. Because of the architecture of LSS, which includes cumulative summing in the vertical pillars within each frustum, changes to camera height have a relatively small impact on downstream BEV segmentation performance in comparison to other viewpoint changes. CVT lacks this generalization to changes in camera height because it does not sum features in the height dimension, but rather conditions on the camera extrinsics. We also note that because the training dataset is acquired in simulation, the extrinsics of the source rig have no noise or calibration error, and, thus, are always the same during training. As a result, we found that CVT learns to ignore the extrinsic embedding during training, indicating that the degradations we see to test performance in Fig. 3 are the result of the images being out of distribution. In contrast, our experiments in Sec. 5 involve training CVT on real world data, which has calibration error, and, as a result, CVT learns to use the extrinsic embedding to inform predictions, but still lacks generalization to target rigs.

During our analysis, we also found that while changes to yaw have a negative impact on performance on LSS, the resulting segmentation predictions are transformed based on the difference in yaw between training and testing. To mitigate this, a post-processing step can be applied where the predictions are rotated to the viewpoint of the target rig. Post-processing can be used to mitigate the effect of changes in yaw, but does not generalize to other extrinsic parameters, such as pitch and height.

Lastly, we note two biases in the oracle models. First, we observe that the LSS oracle model trained on negative pitches performs poorly. Second, both LSS and CVT achieve higher test IoU when trained and tested with rigs with a larger camera height. While higher IoU could be explained by fewer occlusions due to a higher viewpoint, and thus more ground truth pixels, the number of ground truth objects is consistent across each of the test datasets (7 objects per frame on average), and so this bias is not explained by differences in the number of ground truth pixels. We note these biases, but they are not the main focus of our work.

Training Details: We train each BEV segmentation model three times and show the mean and standard deviation in test IoU in Fig. 3. Each model is trained on 25,000 images rendered from the front center camera (same camera parameters as in nuScenes) with the CARLA Simulator [5]. Train datasets are created for all camera viewpoints tested so that an oracle model can be constructed. For evaluation, we use 5,000 test images from each target rig, where the target rigs include changes to camera pitch, yaw, height, or pitch and height together, and are rendered from a different CARLA map than the training sets. Each model is trained for 30 epochs. We will release all 36 train and test datasets with this paper. The 36 datasets include train and test data for the source rig, 10 pitch rigs, 10 yaw rigs, 10 height rigs, and 5 height and pitch rigs.

4. Viewpoint Robustness via NVS

We present a new method that improves generalization of BEV segmentation models to different camera positions using novel view synthesis (NVS). As described in Sec. 3, BEV segmentation models fail to generalize to even small changes in camera viewpoint. However, collecting new data from each target rig, especially when AV companies may wish to deploy models across many types of cars, is impractical due to the cost of collection and annotation. Thus, we focus on NVS as it provides an opportunity to reuse labeled data from the source rig by transforming it into the viewpoint of each target rig. We can then train a new model on the transformed data for each target rig. We first define our NVS method. The key difference between our NVS method and past work is how we generalize to complex, dy-

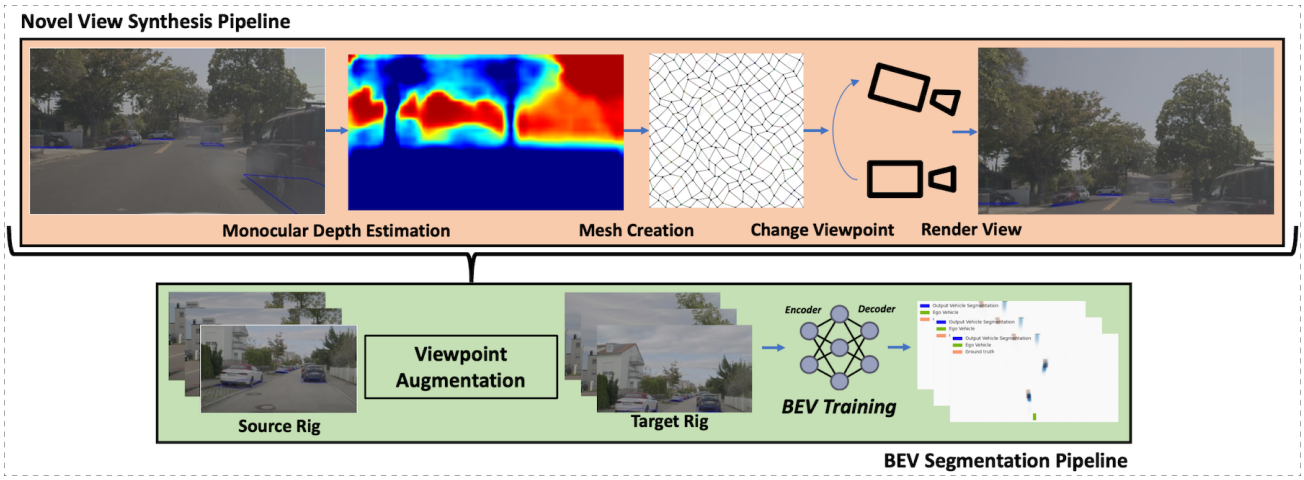

Figure 4. **Proposed Pipeline**. Current methods for bird's eye view (BEV) segmentation are trained on data captured from one set of camera rigs (the source rig). At inference time, these models perform well on that camera rig, but, according to our analysis, even small changes in camera viewpoint lead to large drops in BEV segmentation accuracy. Our solution is to use novel view synthesis to augment the training dataset. We find this simple solution drastically improves the robustness of BEV segmentation models to data from a target camera rig, even when no real data from the target rig is available during training.

namic AV scenes. Then, we show how the transformed data can be used to train BEV segmentation models for diverse target rigs without access to real data from the target rig. We use real data to train our NVS and BEV segmentation models. To evaluate over diverse target rigs, we use synthetic data rendered with NVIDIA DRIVE Sim since real data only provides one rig setting. We compare test performance achieved with models trained with data transformed to the target viewpoint vs. only data from a source rig. Our approach is summarized in Fig. 4.

4.1. Preliminaries

We build off of Worldsheet [11], a recent method for single image NVS of *static scenes*, extending it to work on complex AV scenes that have *dynamic* objects and occlusions. While NeRF-type approaches generate impressive NVS results, generalizing to dynamic scenes and across many scenes is still an active area of research. Worldsheet, on the other hand, is able to generalize across scenes, which is why we choose to use it in our work. The goal of Worldsheet is to build a 3D scene mesh, M, by warping a $W \times H$ lattice grid onto the scene based on predicted depths and vertex offsets. Given an input image, I, a ResNet-50 [9] is trained to predict depth, z, and grid offset of each vertex, $V_{(x,y)}$ at each (x, y) in I. z and $V_{(x,y)}$ are used to build $M = (\{V_{(x,y)}\}, \{F\})$, where F are the mesh faces. A differentiable texture sampler is then used to splat the RGB pixel intensities from the original image onto the mesh's UV texture map. The pipeline is trained end-to-end on a multi-view consistency loss. Given two views of the scene, an input and a target, the mesh is predicted from the input view and then projected to the target view based on the target

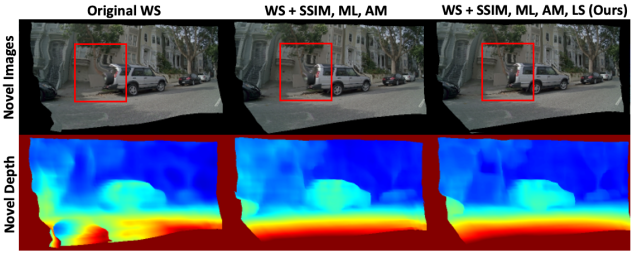

Figure 5. **NVS Qualitative Comparison:** We compare the unrectified NVS results (top) and depth results (bottom) from Worldsheet [11] (right) to our method (middle and left). SSIM is SSIM loss, ML is min loss, AM is automasking, LS is lidar supervision.

camera pose, θ_t. The target view is rendered and compared to the GT with L1 and perceptual losses. A pix2pixHD generator inpaints parts of the scene in the generated target view that were not visible in the input. In contrast, we omit the pix2pixHD generator and use lidar depth supervision (LS), SSIM loss [27], automasking (AM) & minimum loss (ML) over neighboring frames [7] to build an NVS model that generalizes to complex, dynamic, AV scenes.

4.2. Novel View Synthesis for AV Data

Overview: Because AV sessions, S, are composed of temporally sequential images, $\{I_0, I_1, ..., I_n\} \in S$, temporal consistency, rather than multi-view consistency, can be enforced between neighboring images to train our NVS model, assuming a sufficiently high frame rate so parts of the scene are visible in the input and target images. For every input image, I_n, we enforce consistency between I_{n-1} and I_{n+1} by transforming I_{n-1} and I_{n+1} into the viewpoint of I_n and comparing each predicted novel view $\hat{I}_n$ to GT I_n:

Figure 6. **Novel View Synthesis Qualitative Results:** Shown above are the novel view synthesis results (rectified) obtained with our method. We transform images from the source rig to each of the target viewpoints and then use them for BEV segmentation training.

$$\begin{aligned}\{\hat{\mathbf{I}}_n^{n+1}, \hat{\mathbf{D}}_n^{n+1}\} &= render(\{V_{(x,y)}^{n+1}\}, \{F^{n+1}\}, T^{n+1}) \\ \{\hat{\mathbf{I}}_n^{n-1}, \hat{\mathbf{D}}_n^{n-1}\} &= render(\{V_{(x,y)}^{n-1}\}, \{F^{n-1}\}, T^{n-1}) \\ L_{im} &= \frac{1}{\mathcal{P}} \sum_{i=1}^{\mathcal{P}} \min(|I_{n,i} - \hat{\mathbf{I}}_{n,i}^{n+1}|, |I_{n,i} - \hat{\mathbf{I}}_{n,i}^{n-1}|)\end{aligned} \tag{1}$$

where V are vertices, F are mesh faces, and T is the texture map. We render the meshes built from I_{n-1} and I_{n+1} in I_n's viewpoint, forming novel view renderings $\hat{I}_n \in (\hat{\mathbf{I}}_n^{n+1}, \hat{\mathbf{I}}_n^{n-1})$ and their corresponding depth maps $\hat{\mathbf{D}}_n^{n+1}, \hat{\mathbf{D}}_n^{n-1}$. We then compute the per-pixel image loss L_{im}, where $\mathcal{P}$ is the valid pixel number and $I_{n,i}$ is the i-th pixel of I_n. Different from NeRF, worldsheet applies a single-layer mesh to synthesize novel views. In the discontinuous depth regions(*e.g.*, boundaries), distortion might happen. To make the training more robust, we apply L_1 and SSIM loss between the GT image I_n and the re-rendered image $\hat{I}_n$, where we follow the same setting in [7].

Occlusion Handling: Inspired by unsupervised ego-video depth estimation work [7], we compute two losses between $(I_n, \hat{\mathbf{I}}_n^{n-1})$ and $(I_n, \hat{\mathbf{I}}_n^{n+1})$, and pick up the minimal loss (ML) between them in a pixel-wise way. Intuitively, as the car is moving, some parts of the scene might be occluded in the last or next frame. However, they are less likely to be occluded in both two frames. Therefore, applying minimal losses help prevent occlusions from affecting the training loss. We also use auto-masking [7] to ignore pixels that violate camera motion assumptions, *e.g.*, ego-car shadows.

Depth Supervision: Unlike other applications where only an RGB sensor is available, AVs are often equipped with lidar during data collection. We assume that lidar observations are available when training our NVS model. Thus, we can leverage lidar supervision (LS), rendering lidar into a ground truth sparse depth map [23], D_n for every image, I_n. To further improve the quality of the lidar depth maps, we use two types of automasking (AM). First, we use a pre-trained sky segmentation network [26] to mask out the sky and set the depth for this part of each training image to infinity. Second, we use MaskRCNN [8] to predict masks of the "close-by" cars so that they are ignored in the depth loss, due to the fact that the lidar detector is mounted higher than the camera and it typically cannot see the close cars.

Approach	Im. L1 ↓	PSNR ↑ (dB)	SSIM ↑	Depth L1 ↓
WS (original)	0.145	22.602	0.595	0.00763
WS + SSIM, ML, AM	0.141	22.819	0.606	0.00707
WS + SSIM, ML, AM + LS (Ours)	**0.138**	**22.936**	**0.608**	**0.00657**

Table 1. **NVS Ablation:** We ablate our changes, which improve NVS and depth over Worldsheet (WS). We test with 1K images.

We then apply two depth losses, an L1 loss between the predicted depth and GT lidar depth (*direct* depth loss) and the L1 loss between the predicted depth and ground truth depth after the prediction is projected into the viewpoint of the cameras at frame $n+1$ and $n-1$ (*rendered* depth loss). As above, we also use minimal loss for depth supervision:

$$\begin{aligned}L_D^{direct} &= \frac{1}{\mathcal{P}} \sum_{i=1}^{\mathcal{P}} |D_{n-1,i} - F_{depth}(I_{n-1,i})| + \\ &\quad |D_{n+1,i} - F_{depth}(I_{n+1,i})| \\ L_D^{rendered} &= \frac{1}{\mathcal{P}} \sum_{i=1}^{\mathcal{P}} min(|D_{n,i} - \hat{\mathbf{D}}_{n,i}^{n+1}|, |D_{n,i} - \hat{\mathbf{D}}_{n,i}^{n-1}|)\end{aligned} \tag{2}$$

Fig. 5 shows how our method (SSIM, ML, AM, LS) improves depth estimation and NVS compared to Worldsheet. These improvements are quantitatively validated in Table 1.

Inpainting: We train and test our NVS model using images from a 120° f-theta camera. The images are then rectified to 50° after NVS, such that missing parts of the scene not

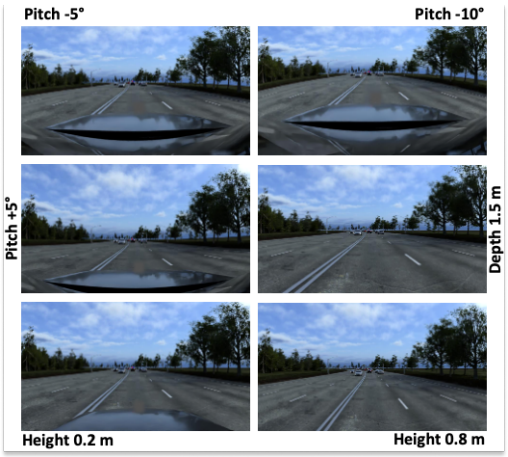

Figure 7. **Evaluation Data:** We use images from NVIDIA DRIVE Sim [20] to evaluate our method on a diverse set of target rigs. Shown here are example test images with different viewpoints.

in the field of view of the final image. As a result, no image inpainting is needed. Our NVS results are shown in Fig. 6.

4.3. Augmenting BEV Segmentation Training

The focus of our paper is not on NVS quality, but on the impact using NVS generated data can have on the problem of *viewpoint robustness* in AV. Given a labeled BEV segmentation training dataset, D_{source}, of size N, we use our NVS method to transform n images from D_{source} to the viewpoint of the target rig, obtaining D^{pred}_{target} of size n. This transformation is done by (1) estimating the depth each image, (2) creating meshes, (3) changing the viewpoint of the cameras, and 4) rendering each image in the viewpoint of the target rig. Finally, we construct a new BEV dataset, D_{final} of size N, containing the n transformed images from D^{pred}_{target} and $N - n$ images from D_{source}. The number of transformed images, n, is a hyperparameter and in our experiments we transform 25%, 50%, or 100% of D_{source} to the viewpoint of the target. The reason we do not always transform all N images is the NVS model may introduce other domain gaps; an ablation on this is done in Sec. 6. We train both the NVS model and BEV segmentation model on a real-world dataset, described in Sec. 5.1. An overview of the training pipeline is shown in Fig. 4.

5. Experiments and Results

We show the effectiveness of our method by using it to train BEV segmentation models for diverse target rigs, without any access to real data from the target rig during training. We first train our NVS model to transform data from the source rig to the target viewpoint. Next, we transform some or all of the source rig training data to the target rig. Finally, we train the BEV segmentation model for the target rig using a combination of transformed data and source data. All training is done on real world data, but evaluation is done with NVIDIA DRIVE Sim, allowing us to test across target rigs that are not available in public datasets.

Extrinsic	Δ View	Source	Source*	Extr Aug	Ours
-	0	0.170	0.170	0.155	-
Pitch	-10°	0.014	0.078	0.126	**0.165**
Pitch	-5°	0.037	0.141	0.128	**0.161**
Pitch	+5°	0.016	0.076	0.028	**0.173**
Depth	1.5 m	0.017	0.156	0.150	**0.174**
Height	0.2 m	0.094	0.175	0.145	**0.177**
Height	0.8 m	0.003	0.170	0.132	**0.214**

Table 2. **Results:** We report the IoU of the CVT model trained on a source rig and tested across target rigs where pitch, depth, and height are changed (source). We then compare against two baselines, described in text. Last, we compare with our method, which is trained with some data transformed to the target rig view. The first row shows IoU of the source evaluated on sim data from the same viewpoint, and is our best estimate of oracle performance.

5.1. Datasets

Training: We train both the NVS and the BEV segmentation model on an internal dataset of 43 real AV sessions. We subsample the images from each video at a higher frame rate for our NVS training dataset than our BEV segmentation training dataset, yielding 250,000 and 30,000 training images respectively. All images are captured from a 120° f-theta lens camera. Prior to BEV segmentation training, we rectify the images to 50°. Examples of rectified images from the source rig are shown in the first column of Fig. 6.

Evaluation: We use simulated data from challenging scenes for the evaluation since real datasets with large viewpoint changes are not available and collecting them across many views is impractical. Simulated data could be used for train and test, but generating sufficiently large and diverse simulated train datasets is difficult. To mitigate the domain gap of training on real data and testing on simulated data, we use NVIDIA DRIVE Sim. Example images are shown in Fig. 7. To measure the domain gap, we trained a model on real data and evaluated it on both a real test dataset and a simulated test dataset from the source rig. The gap was 7.5% IoU, which is acceptable for our work, since we are concerned with relative changes in IoU, not absolute IoU.

5.2. Experiment Details

We demonstrate our method by transforming the dataset from the source rig, D_{source}, to the viewpoint of six target rigs, training a BEV segmentation model for each, and evaluating the model on simulated data from the target rig. We conduct experiments with a single camera rig. The target rigs include pitch -10°, -5°, and 5°, depth 1.5 m, and height 0.2 m and 0.8 m. Examples of source rig data transformed to each of the target rigs with the NVS model are shown in

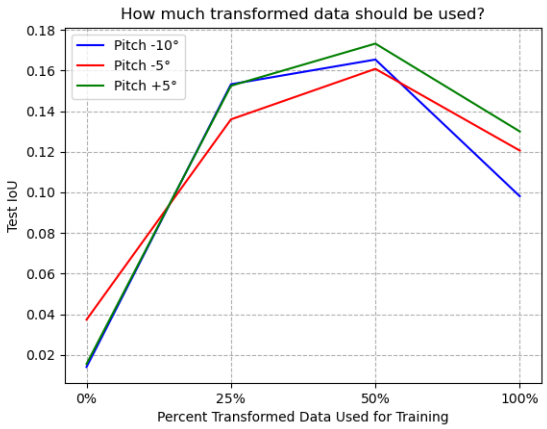

Figure 8. **Ablation: Varying percent transformed training data:** We observe that transforming 25-50% of the training dataset to the viewpoint of the target rig results in the best test IoU.

Fig. 6. We note that, quantitatively, the NVS quality is best for changes in pitch and lowest for large changes in height. Despite lower quality for some transformed viewpoints, we show that the transformed data still leads to significant improvements in BEV segmentation accuracy for each target rig. For each target rig, we train a Cross View Transformers (CVT) model three times, with 25%, 50%, and 100% of D_{source} transformed to the target rig viewpoint. We also train CVT on source rig data for comparison.

5.3. Baselines

We compare against two baseline approaches:

– Using Train Extrinsics at Inference Time (*Source):** By passing in the train extrinsics to the BEV segmentation model at inference time, we find that, despite the image itself being from a different rig, performance improves.

– Extrinsic Augmentations (*Extr. Aug.*): Rather than augmenting the training images to be from the viewpoint of the target rig, we instead apply random rotations to both the extrinsic matrix and 3D bounding box labels together within the bounds of extrinsics of the target rigs.

5.4. Results

We find that our approach of training BEV segmentation models with 25%, 50%, or 100% data transformed into the view of the target rig significantly improves BEV segmentation accuracy compared to training with only data from the source rig, leading to the same level of accuracy as when there is no viewpoint change. Results are shown in Table 2. We report the best IoU from the models trained with 25%, 50%, or 100% transformed data, but note that all top performing models use only 25% or 50% transformed data, and the rest of the training data remains from the source rig. We observe that both baselines also significantly improve the IoU compared to the model trained only on source data, but not as much as our NVS approach. We also compute the IoU of the model trained only on the source rig and tested on synthetic data from the same viewpoint to serve as a reference upper bound for expected performance when there is no viewpoint gap, shown in the first row of Table 2. This upper bound is more reliable than training and testing on simulated data, which results in an average of 35.4% IoU across views due to the lack of domain gap and limited diversity, resulting in visually similar train and test data.

Lastly, we conducted an experiment in which we trained a model on $\frac{1}{2}$ source rig and $\frac{1}{6}$ +5° pitch, $\frac{1}{6}$ +1.5 m depth, and $\frac{1}{6}$ +0.2 m height data, resulting in 0.19 mean test IoU across views in Tab. 2 (0.206 for train views and 0.178 for other views). This result suggests training on multiple views can improve IoU over training only on the target view. Altogether, our results support our hypothesis that using NVS to transform labeled train data from the viewpoint of a source rig to that of a target rig and then training a BEV segmentation model with that data can enable the creation of BEV segmentation models for target rigs without the associated cost of collecting and annotating data from each target rig.

6. Discussion

We observe that, despite some NVS transformations leading to artifacts, e.g. the +0.8 m height transformation, the images still significantly help downstream BEV segmentation models to generalize to the desired target rig. In addition to our main results, we also conduct two ablation studies on our method, which are described below.

Amount of Transformed Data: An open question is how much data from the source rig dataset should be transformed to the viewpoint of the target rig. While transforming all of the data may lead to a content gap due to NVS being imperfect, transforming too little may not expose the BEV segmentation model to enough examples of data from the target rig viewpoint. In our experiments, we train BEV segmentation models with 25%, 50%, and 100% transformed data. Shown in Fig. 8 is the IoU as a function of the amount of transformed training data. We see that IoU consistently increases as more transformed data is added to training until 50%. The model trained with 100% underperforms, most likely due to other domain gaps introduced by NVS.

Interpolation and Extrapolation: In our work, we focus on generating target rig specific BEV segmentation models without the cost of data collection. However, one may wish to create a single BEV segmentation model that generalizes to multiple camera rigs. We investigate whether our approach can enable that by testing how models trained with two viewpoints interpolate between those viewpoints and extrapolate beyond those viewpoints. We test all combinations of the pitch models trained with 50% transformed data

and 50% source rig data, averaging test performance for interpolatation and extrapolation. An example of interpolation is testing a model trained on 0° and -10° pitch on -5° pitch, while an example of extrapolation is testing a model trained on 0° and -5° pitch on -10° pitch. On average, we find interpolation performance is 14.9% IoU and extrapolation performance is 14.8% IoU, suggesting the proposed method can improve generalization beyond the target rig.

7. Conclusion

We find that changing camera viewpoint, even by small amounts, has a significant impact on BEV segmentation models that have not been trained on that viewpoint. As AVs become more ubiquitous and companies scale across different vehicle types, this problem, which we dub *viewpoint robustness*, will become critical to address. Our work makes a first attempt at improving viewpoint robustness using data generated from our method for NVS. We find that augmenting the BEV segmentation train dataset with data generated from the viewpoint of the target camera rig improves generalization to the target rig. As part of our work, we propose a method for NVS and show that it can be used to effectively mitigate the viewpoint domain gap.

Acknowledgements: We thank Alperen Degirmenci for his valuable help with AV data preparation and Maying Shen for her valuable support with experiments.

References

[1] David Acuna, Jonah Philion, and Sanja Fidler. Towards optimal strategies for training self-driving perception models in simulation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[2] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[3] Benjamin Coors, Alexandru Paul Condurache, and Andreas Geiger. Nova: Learning to see in novel viewpoints and domains. In *2019 International Conference on 3D Vision (3DV)*, pages 116–125. IEEE, 2019.

[4] Tien Do, Khiem Vuong, Stergios I Roumeliotis, and Hyun Soo Park. Surface normal estimation of tilted images via spatial rectifier. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 265–280. Springer, 2020.

[5] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[6] Nathan Drenkow, Numair Sani, Ilya Shpitser, and Mathias Unberath. Robustness in deep learning for computer vision: mind the gap? *arXiv preprint arXiv:2112.00639*, 2021.

[7] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J Brostow. Digging into self-supervised monocular depth estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3828–3838, 2019.

[8] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

[11] Ronghang Hu, Nikhila Ravi, Alexander C Berg, and Deepak Pathak. Worldsheet: Wrapping the world in a 3d sheet for view synthesis from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12528–12537, 2021.

[12] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. Geonerf: Generalizing nerf with geometry priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18365–18375, 2022.

[13] Spandan Madan, Timothy Henry, Jamell Dozier, Helen Ho, Nishchal Bhandari, Tomotake Sasaki, Frédo Durand, Hanspeter Pfister, and Xavier Boix. When and how do cnns generalize to out-of-distribution category-viewpoint combinations? *arXiv preprint arXiv:2007.08032*, 2020.

[14] Spandan Madan, Tomotake Sasaki, Tzu-Mao Li, Xavier Boix, and Hanspeter Pfister. Small in-distribution changes in 3d perspective and lighting fool both cnns and transformers. *arXiv preprint arXiv:2106.16198*, 2021.

[15] Claudio Michaelis, Benjamin Mitzkus, Robert Geirhos, Evgenia Rusak, Oliver Bringmann, Alexander S Ecker, Matthias Bethge, and Wieland Brendel. Benchmarking robustness in object detection: Autonomous driving when winter is coming. *arXiv preprint arXiv:1907.07484*, 2019.

[16] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16190–16199, 2022.

[17] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[18] Norman Müller, Andrea Simonelli, Lorenzo Porzi, Samuel Rota Bulò, Matthias Nießner, and Peter Kontschieder. Autorf: Learning 3d object radiance fields from single view observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3971–3980, 2022.

[19] Michael Niemeyer, Jonathan T Barron, Ben Mildenhall, Mehdi SM Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *Proceedings of the IEEE/CVF Con-*

ference on Computer Vision and Pattern Recognition, pages 5480–5490, 2022.

[20] NVIDIA. Nvidia drive sim. `https://developer.nvidia.com/drive/simulation`, 2021.

[21] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 194–210. Springer, 2020.

[22] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021.

[23] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020.

[24] Thomas Roddick and Roberto Cipolla. Predicting semantic map representations from images using pyramid occupancy networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[25] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*, 2018.

[26] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. *arXiv preprint arXiv:2005.10821*, 2020.

[27] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[28] Tianhao Wu, Fangcheng Zhong, Andrea Tagliasacchi, Forrester Cole, and Cengiz Oztireli. D2 nerf: Self-supervised decoupling of dynamic and static objects from a monocular video. *arXiv preprint arXiv:2205.15838*, 2022.

[29] Brady Zhou and Philipp Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13760–13769, 2022.