2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

NeuralFloors++: Consistent Street-Level Scene Generation From BEV Semantic Maps

Valentina Muşat†, Daniele De Martini‡, Matthew Gadd‡ and Paul Newman
Mobile Robotics Group (MRG), University of Oxford, † Corresponding author ‡*Equal contribution*
{valentina,daniele,mattgadd,pnewman}@robots.ox.ac.uk

***Abstract*— Learning autonomous driving capabilities requires diverse and realistic training data. This has led to exploring generative techniques as an alternative to real-world data collection. In this paper we propose a method for synthesising photo-realistic urban driving scenes, along with semantic, instance and depth ground-truth. Our model relies on Bird's Eye View (BEV) representations due to their compositionality and scene content control capabilities, reducing the need for traditional simulators. We employ a two-stage process: first, a 3D scene representation is extracted from BEV semantic, instance and style maps using a neural field. After rendering the semantic, instance, depth and style maps from a ground-view perspective, a second stage based on a diffusion model is used to generate the photo-realistic scene. We extend our prior work - NeuralFloors, to include multiple-view outputs, style manipulation for finer control at the object level through instance-wise style maps and cross-frame consistency via auto-regressive training. The proposed system is evaluated extensively on the KITTI-360 dataset, showing improved realism and semantic alignment for generated images.**

I. Introduction

Generative AI has made exceptional progress in the last few years, especially in image and video generation, where scene synthesis has popular applications in creative industries, gaming, and robotics. Synthetic data generation has become increasingly crucial for training computer vision models, particularly in safety-critical applications like Autonomous Driving (AD). Here, generated data paired with its source Ground Truth (GT) offers a cheap, scalable, and easier-to-obtain data source that complements real-world data collection. Additionally, it eliminates exposure to real-world risks, thus allowing for the generation of challenging and rare-case scenarios that would otherwise be unattainable.

Traditional simulators, while widely employed, often lack scalability and realism due to their reliance on manually created 3D assets and artists for appearance modelling. Recent advancements in Neural Radiance Fields (NeRFs) and diffusion-based models have shown remarkable potential in synthesizing realistic scenes, making them promising candidates for data-driven simulators. Inspired by this, we present a method for synthesising outdoor urban driving photo-realistic ground-view scenes paired with GT semantic, instance, depth and style maps, starting from a Bird's Eye View (BEV) representation – shown in Fig. 1. Our approach is factorized into two stages: 1) learning a 3D representation of the scene conditional on a BEV representation through a neural field, and 2) generating ground-view photo-realistic scenes based on the output of the first stage, using a diffusion-based model. Our work, NeuralFloors++, extends NeuralFloors [1] to multi-view-consistent output and incorporates scenes containing objects in motion. We introduce style manipulation by employing an instance-wise average style BEV map inspired by pix2pixHD [2] and enhance cross-frame consistency by introducing auto-regressive training in the second stage. In contrast to CC3D [3], which employs a global latent style code, our approach allows for finer style control at the object level, and provides a wider range of ground-view modalities. Finally, by leveraging a BEV semantic map that is isometric and inherently compositional in 2D, we enhance compositionality and control of the scene contents bypassing the need for a traditional simulator.

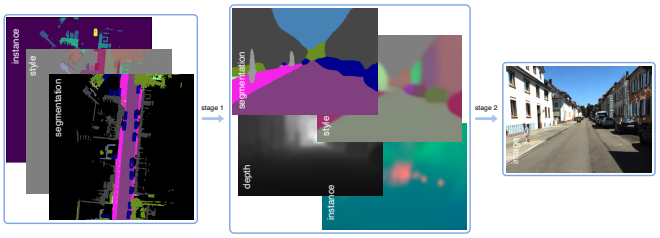

Fig. 1: NeuralFloors++ generates photo-realistic ground-view scenes (*Right*) paired with ground-truth semantic, instance, depth and style maps (*Middle*), starting from a BEV representation (*Left*).

To summarise, our key contributions are as follows: 1) a system that allows ground-view scene synthesis from a BEV representation that can 2) output a consistent style which is 3) controllable and offers object-level manipulation. Our approach is 4) able to synthesise a diverse set of ground-view modalities that can be further used to train downstream tasks. We conduct thorough experiments on the KITTI-360 dataset [4], evaluating our method using a variety of metrics, including quantitative measures for depth, segmentation, perceptual quality, and we present a set of qualitative analyses and ablation studies.

II. Related Work

Traditional data acquisition, whilst offering the most realistic data for training and testing, suffers from limited diversity, complexity and scarce pixel-level GT, especially

*Supported by a DeepMind Engineering Science Scholarship and EPSRC Programme Grant "From Sensing to Collaboration" (EP/V000748/1). The authors also acknowledge the use of Hartree Centre resources and the University of Oxford Advanced Research Computing facility in this work.

when moving from 2D to 3D. Asset-based, classic 3D simulators [5] have been a common choice for gathering data due to their physically-grounded outputs, virtually unlimited GT data and high level of control; however, they require laborious asset building, making the process difficult to scale, while the inherent gap between the complexities of the real world and the approximations made by the simulation engines limits their applicability. To address this, researchers have proposed data-driven simulators, with the latest approaches leveraging Large Language Models (LLMs) to generate environments [6]. Since the success of Generative Adversarial Networks (GANs) [7], works such as pix2pixHD [2] have shown great capabilities of synthesizing high-resolution, realistic images from segmentation maps, with geometric [8] and temporal consistency [9] being enforced by extra conditioning. Finally, semi-parametric methods combine the benefits of both learnt and data-driven approaches by relying on existing data to encourage realism [10], [11].

A. 3D-aware scene generation

Whereas photorealism has been the initial priority, interpolation, disentanglement and compositionality of attributes and contents are paramount to gaining control over the synthesis process. However, as image synthesis takes place in 2D, capturing effects such as shading, occlusions, objects' poses, and appearance interactions requires an understanding of the 3D world. Various approaches have tackled 3D geometry-aware synthesis by employing inductive biases in combination with differentiable rendering [12], [13], [14].

GIRAFFE [15] proposes to tackle compositionality in 3D via compositional generative feature fields, where the scene is composed of background and foreground objects, generated by separate Multi-Layer Perceptrons (MLPs). Subsequently, GIRAFFE HD [16] improved the architecture by employing a StyleGAN2-like renderer [17], enabling synthesis of high-quality scenes. While most 3D generative models have focused on object-centric scenes, GSN [18] generates unbounded indoor scenes but does not tackle compositionality, making it difficult to control the scene content. CC3D [3] and NeuralFloors [1], instead, tackle outdoor scenery and enable scene-content control via BEV semantic map conditioning.

B. Cross-view and cross-modality scene generation

GSN [18] trains an unconditional generative model to render scenes from a freely-moving camera via a NeRF and a latent floorplan representation. In their setup, a 1D Gaussian noise latent code representing the whole scene is mapped to a 2D grid of latent codes from which features are sampled using 2D coordinates. However, NeuralFloors [1] and CC3D [3] argue that this representation is not expressive enough to offer the level of control, compositionality, and interpretability generally required for scene generation, especially from a robotics application point of view. To overcome this limitation, they propose to condition the model on a top-down semantic-segmentation map representing the scene.

NeuralFloors [1] relies on a triplane representation from a single floorplan projection plane as in EG3D [19], whereas CC3D [3] extrudes the floorplan representation into a 3D grid of features. BerfScene [20] aims to tackle the same task but uses a 2D Fourier feature map of grid coordinates as input, while the BEV map and a latent code are used to modulate the encoder-decoder network. InfiniCity [21] also relies on a neural renderer to generate the ground-view but requires hard-to-acquire CAD models to represent the scene.

Outdoor scene generation in a BEV to ground-view cross-view setup has also been tackled by non-NeRF approaches such as BEVGen [22] and BEVControl [23]. BevGen learns the image formation process implicitly with an autoregressive transformer fed with multi-view images, BEV semantic layout and token direction vectors. BevControl conditions a diffusion model with a BEV sketch geometrically projected in the camera space and relies on cross-view attention to generate geometry- and appearance-consistent images.

C. Diffusion-based models

Latent Diffusion Models (LDMs) have become popular alternatives to GANs, which generally suffer from mode collapse and inherent training instability. Popular works such as ControlNet [24], T2I-Adapter [25] and InstanceDiffusion [26] unlock flexible user control of the synthesis process by conditioning a diffusion model with, e.g., edges, poses, depth, or normal maps. While this approach works well for still images, controllable video generation is more challenging as objects must follow spatial and temporal constraints. VideoComposer [27] addresses this by incorporating motion vectors and a spatiotemporal encoder employing cross-frame attention, while Lumiere [28] introduces a Space-Time UNet architecture that generates the full video sequence at once.

D. Contemporary works

Our work is based on NeuralFloors [1], which generates ground-view images conditioned on the BEV semantic map of the scene contents. To relax the assumption of paired semantic BEV maps and ground-view RGB images, the model is trained in two stages, each leveraging an architecture specialised for the task. The first stage involves a neural field to lift and extract a 3D representation of the scene, while the second stage relies on an LDM for high-quality image synthesis. However, NeuralFloors is trained on single-view images only, possibly leading to geometric ambiguity, while the second stage does not model frame-to-frame consistency. NeuralFloors++ builds on top of this work by training on multi-view data in the first stage and enhancing frame-to-frame consistency in the second stage by a style map inspired by pix2pixHD [2] and auto-regressive training.

The closest work to ours is CC3D [3], which is also conditional on the top-down semantic layout of the scene. However, there are key differences: while NeuralFloors reshapes the 2D feature maps into a tri-plane representation as in EG3D [19], CC3D reshapes them into a 3D feature grid. Additionally, CC3D is trained end-to-end, synthesising the ground-view RGB images directly with a dual discriminator,

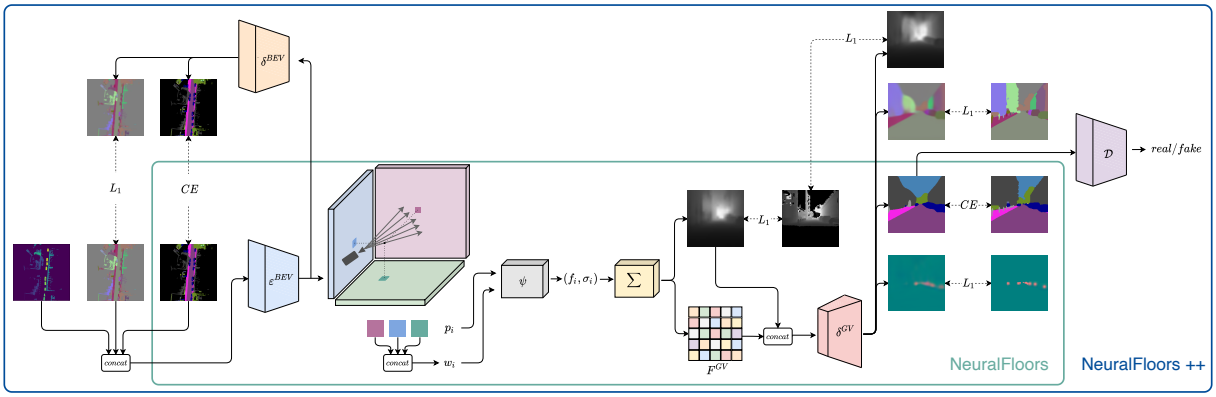

Fig. 2: **Stage 1 architecture.** BEV semantic, instance and style maps are encoded into a latent representation from which per-point features w_i are sampled via bi-linear interpolation. A neural field ψ learns a 3D representation of the scene and outputs per-point features and densities that are further aggregated by a volume renderer to obtain expected depth and feature maps. A decoder predicts the corresponding ground-view semantic, instance, style and upscaled depth maps. Additionally, a BEV decoder reconstructs the semantic and style maps from the latent representation, and a segmentation discriminator judges whether the ground-view segmentation is real.

in contrast with our 2-stage method. Moreover, CC3D manipulates the style of the generated scene through a 1D global latent noise vector, while we employ a 2D instance-wise style map spatially aligned with the BEV semantic map, allowing for finer control of style at the object level. Finally, we introduce a semantic-map discriminator to discern between ground-view generated and GT semantic maps.

III. Method

Our methodology borrows from NeuralFloors [1], where the setup is a two-stage approach, with each stage employing architectures specialised for its task. The first stage learns a mapping from BEV to ground-view representations of the scene, while the second synthesises highly photo-realistic ground-view images using the output of the first stage.

A. Semantic lifting

We leverage a ray-based point sampler following a pinhole camera model, coupled with a neural field to learn a 3D representation of the scene, conditional on 2D maps, and employ a volumetric renderer to produce ground-view maps. The system diagram for the first stage is depicted in Fig. 2.

We start from a BEV semantic segmentation map S^{BEV}, an instance ID map Q^{BEV}, and an instance average style map H^{BEV}. These inputs are representative of the local scene we want to render from ground-view, encompassing semantic and appearance information. An encoder ε^{BEV} learns to encode them into a latent representation W^{BEV}, which is reshaped into 3 orthogonal planes $W^{BEV} = \{W^{XZ}, W^{XY}, W^{YZ}\}$, similar to EG3D [19] tri-plane representation. Additionally, we enforce BEV layout consistency by reconstructing S^{BEV} and H^{BEV} via a decoder δ^{BEV}.

A ray-based sampler is used to cast R rays, with P 3D points $\{p_i | p_i = (x_i, y_i, z_i)\ \forall i = 1, ..., P\}$ along each ray r. For each point, we sample 3 latent features via bi-linear interpolation from the 3 orthogonal planes, which are further concatenated to obtain an aggregated per-point latent feature $w_i = w_i^{XZ} \oplus w_i^{XY} \oplus w_i^{YZ}$. The latent feature w_i and the corresponding 3D coordinate p_i – concatenated with its positional-encoding $\rho(p_i)$ – are given as input to a neural field ψ parametrised by an MLP, to produce a final corresponding feature and density $(f_i, \sigma_i) = \psi(p_i \oplus \rho(p_i), w_i)$.

Given the volume of densities and features, the ground-view feature image is obtained by aggregating the weighted features f_i of all the points p_i across each ray $r \in R$:

$$\hat{F}(r) = \sum_{i=1}^{P} \alpha_i T_i f_i \text{ where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \delta_j\right) \quad (1)$$

with T_i denoting the transmittance along the ray, $\alpha_i = 1 - \exp(-\sigma_i \delta_i)$ the opacity of point i, and δ_j the distance from point j to point $j+1$. Similarly, to approximate the ground-view expected depth, the distances t_i to points p_i are weighted and aggregated along each ray $r \in R$:

$$\hat{D}(r) = \sum_{i=1}^{P} \alpha_i T_i t_i \quad (2)$$

By sampling rays from the ego-perspective of the vehicle, we obtain the expected ground-view depth $\hat{D}^{GV}$ and ground-view feature $\hat{F}^{GV}$ images, which are further concatenated as input to the decoder δ^{GV} to predict the corresponding ground-view segmentation, instance, style and upscaled depth maps $\hat{S}^{GV}$, $\hat{E}^{GV}$, $\hat{H}^{GV}$ and $\hat{D}_+^{GV}$. For the ground-view instance map we follow a formulation similar to [29], and define $E^{GV} = (C^{GV}, O^{GV})$, where C^{GV} is the instance centres map (centres of mass), and O^{GV} is the instance pixels offsets map (each blob pixel's offset from its corresponding centre along the image axes).

Finally, to encourage the model to produce segmentation with natural shapes, we pass the predicted segmentation map $\hat{S}^{GV}$ to a segmentation discriminator $\mathcal{D}$, which predicts whether $\hat{S}^{GV}$ comes from a distribution of real or synthesised segmentation maps. This component is added to balance the effects of a strict loss criteria applied to outputs with multiple plausible representations. More specifically, given the arrangement of scene contents in a BEV map, there are many configurations of shapes in the ground-view map that satisfy its layout. However, while applying segmentation supervision on the ground-view output (Tab. V), the model

is forced to reconstruct a particular configuration (out of many possible configurations), without knowing which one it should. Although a discriminator does not directly solve the one-to-many mapping, it can be used to balance the effects of strong supervision.

B. Image synthesis

Given the remarkable results of denoising diffusion models [30], we follow [1] and employ a latent diffusion model for the task of ground-view map to RGB image synthesis.

A diffusion model consists of two processes: a forward diffusion process adds noise ϵ drawn from a Gaussian distribution iteratively to an input x to obtain a diffused version x_t, and a reverse diffusion process recovers the original input gradually, by predicting the added noise $\hat{\epsilon}$ using a model ϵ_θ and subtracting it from x_t. Unlike diffusion models applied directly in the pixel space, latent diffusion models are applied in a lower-dimensional latent space that is more computationally efficient but still preserves content richness. In this case, an encoder and decoder learn to encode images in a latent representation z and decode them back. During inference, a diffused latent embedding z_t is sampled from a Gaussian distribution and ϵ_θ is used to incrementally denoise z_t, from which the decoder produces an RGB image.

Our method extends the denoising network ϵ_θ to additionally condition on $z_c = \phi(S^{GV}, E^{GV}, D^{GV}, H^{GV}, I^{GV}_{past})$, where I^{GV}_{past} is a previous RGB image and ϕ is an additional convolutional network used to embed the ground-view data, thus the predicted noise becoming $\hat{\epsilon} = \epsilon_\theta\big(z_t, z_c, t\big)$.

C. Instance-wise average style encoding

To encode style, we draw inspiration from pix2pixHD [2] and construct instance-wise averaged style maps. The purpose of using a style map that has the same spatial resolution as the other BEV inputs is that, as opposed to using a global style code [3], this method offers precise delimitation between objects styles and thus, finer control. Ground-view colour images are given as input to an encoder ε_{style} to produce a latent style map which is upsampled via bi-linear interpolation to the original size of the image. Using the corresponding GT segmentation and instance maps, we select the averaged style features corresponding to unique objects within an image and construct a bank of latent style codes. For classes that do not have individual instances, the average style is calculated across the semantic class in the image. Thus we obtain BEV and ground-view instance-wise averaged style maps H^{BEV} and H^{GV} that encode instance and class-specific style information.

To build the BEV and corresponding ground-view GT style maps, we first construct an empty BEV and ground-view map of the same spatial resolution as the input for each observation set. For a particular instance of an object in the BEV, we randomly pick a style code from the bank of latent style codes particular to that class. Then for all pixel locations that belong to the instance, we broadcast the style code in both the BEV and ground-view maps. Similarly, we broadcast a style code for all segmentation pixels that belong to a particular class but do not have instance information.

D. Losses

For the BEV and ground-view segmentation supervision, we apply Cross-Entropy loss[1]:

$$\mathcal{L}_S = -\mathbb{E}\left[\sum_{n=1}^{N} \alpha_n \sum_{i,j}^{H\times W} h_{i,j,n} \log \hat{S}^*_{i,j,n}\right] \tag{3}$$

where α_n weights each class. Indicator $h_{i,j,n} = 1$ if pixel $(i,j) \in n$ in the GT segmentation map otherwise $h_{i,j,n} = 0$.

For the BEV and ground-view style supervision, we apply an L1 loss between the GT and predicted style maps:

$$\mathcal{L}_H = |H^* - \hat{H}^*| \tag{4}$$

For ground-view instance map reconstruction, we apply L1 loss between GT and predicted centers and offsets:

$$\mathcal{L}_C = |C^{GV} - \hat{C}^{GV}|;\ \mathcal{L}_O = |O^{GV} - \hat{O}^{GV}| \tag{5}$$

For depth reconstruction and training stability, we apply an L1 loss between GT and predicted depth maps:

$$\mathcal{L}_D = M^D \odot |D^{GV} - \hat{D}^{GV}| \tag{6}$$

where mask M^D is used to perform pixel-wise masking if D^{GV} is not within $(t_n, t_f]$ bounds.

Additionally, an L1 loss is applied between the GT depth map and the upscaled depth predicted by decoder δ^{GV}:

$$\mathcal{L}_{D_+} = M^D \odot |D^{GV} - \hat{D}^{GV}_+| \tag{7}$$

All losses of stage 1 are additionally masked in the sparse random view, to account for pixels that do not belong to a particular class (void).

Additionally, an adversarial loss is used to train the generator $\mathcal{G}$ and discriminator $\mathcal{D}$:

$$\mathcal{L}_A = \mathbb{E}[\log \mathcal{D}\big(S^{GV}\big)] + \mathbb{E}[\log\big(1 - \mathcal{D}\big(\hat{S}^{GV}\big)\big)] \tag{8}$$

where the generator is $\mathcal{G} = \delta^{GV} \circ \sum \circ\ \psi \circ \varepsilon^{BEV}$ and its output is $\hat{S}^{GV} = \mathcal{G}(S^{BEV}, Q^{BEV}, H^{BEV})$.

The final loss $\mathcal{L}_{total}$ for stage 1 is a sum of the losses above weighted by their λs, while in stage 2, the training objective of the latent diffusion model is:

$$\mathcal{L}_{LDM} = \mathbb{E}_{z,\epsilon\sim\mathcal{N}(0,1),t}\big[||\epsilon - \hat{\epsilon}||_2^2\big] \tag{9}$$

where ϵ_θ parametrised by a convolutional neural network is trained to predict the noise $\hat{\epsilon} = \epsilon_\theta(z_t, z_c, t)$.

IV. Experimental Setup

A. Data

We train and evaluate the experiments on KITTI-360 [4] – a widely-known dataset that contains complex real-world urban driving scenes, with 20 semantic classes following the Cityscapes [31] convention. Whereas NeuralFloors [1], being based solely on the accumulated point cloud for BEV creation, discards the data with objects in motion, we overcome this problem by using 3D bounding boxes for objects in motion, allowing us to use the full dataset. To create

[1]With the notation $(\cdot)^*$, we denote that a variable $(\cdot)$ can either be $(\cdot)^{BEV}$ or $(\cdot)^{GV}$

the paired data, we load the point cloud and the bounding boxes in Open3D [32], and at every k-th original pose in the KITTI-360 dataset, we generate a set of observations from different view-points. Thus, we sample 4 additional random poses with up to $1.5\,\mathrm{m}$ of lateral, $0.5\,\mathrm{m}$ forward and $40°$ yaw displacements, resulting in observation sets each with 1 dense and 4 sparse ground-view GT maps. For each observation, we render both BEV and ground-view semantic, instance and depth maps. Since the 3D bounding boxes in the ground-view perspective do not have natural shapes, we randomly sample points around the centre of each object's box. We use the first 8 sequences for training and reserve the last sequence for model selection and testing. We chose a step size $k = 5$ for the train set and of $k = 1$ for the test set, resulting in $73\,285$ frames for training, 1800 for model selection and $13\,330$ for testing. The last sequence does not overlap with any of the training sequence and model selection and testing partitions are spatially distinct.

B. Metrics

We follow the literature and evaluate the perceptual quality of the generated images (at resolutions 64^2, 256^2, 512^2) using Fréchet Inception Distance (FID) [33] and Kernel Inception Distance (KID) [34], and Fréchet Video Distance (FVD) [35] for the sequence counterpart. To measure the ability of the model to reconstruct the semantic structure of the scene, we follow [1] and report the Mean Intersection Over Union (mIoU) between the predicted ground-view and GT ground-view semantic maps. Similarly, we report mIoU-alignment, which evaluates the mIoU between ground-view semantic maps extracted from the predicted ground-view images and GT ground-view RGB images, using a pre-trained segmentation model (Deeplabv3+ [36]). The goal is to measure how well the generated ground-view images perform on a segmentation downstream task as compared to the real images, but without introducing undesired variance from the performance of the pre-trained segmentation model itself. Similarly to [1], we report Root Mean Squared Error (RMSE) (metres) between the predicted upscaled ground-view and GT ground-view depth maps to check the accuracy of the synthesised depth. Both single-view and multi-view experiments are evaluated on multi-view data.

C. Baselines

We compare our model with recent work such as NeuralFloors [1] and competing method CC3D [3], but also prior art GSN [18]. The GSN model is trained unconditionally based on normally-distributed noise following the original training scheme, using the extended dataset containing dynamic objects. For CC3D [3], we report two sets of results, one after running the official pre-trained model on our validation data (CC3D pre-trained) and one after training the model from scratch (CC3D trained) – both are either evaluated or trained and evaluated on the extended dataset. Finally, we report our new proposed approaches NeuralFloors++ (SD) and NeuralFloors++ (T2I-adapter) in comparison to NeuralFloors [1], which is trained and evaluated on the previous static dataset.

D. Implementation details

Stage 1 encoders (ε^{BEV}, ε^{style}), decoders (δ^{BEV}, δ^{GV}) and stage 2 VAE and LDM of NeuralFloors++ (SD) are initialised from publicly available weights (SD-v-1-4) [30]. The NeuralFloors++ (SD) LDM is extended with two convolutional layers (ϕ) that embed the conditional inputs, which are concatenated with the standard noisy latent variable and fed into the LDM denoising network. Stage 2 of NeuralFloors++ (T2I-adapter) is the implementation of [25], where we train the Adapter (ϕ) but also fine-tune the UNet, which is initialised from SD-XL-v1.0. For the segmentation-discriminator, we use the OASIS discriminator [37] backbone, which now receives as input a one-hot encoding of the segmentation map and outputs a real/fake signal.

We set $t_n = 0$ and $t_f = 80$ meters. The BEV map has a size of 512×512, covering an area of 80×80 meters. Encoder ε^{BEV} receives input of shape $25 \times 512 \times 512$ and outputs W^{BEV} of shape $96 \times 128 \times 128$. Decoder δ^{BEV} receives W^{BEV} and outputs $24 \times 512 \times 512$. We sample 2136 rays with 200 points per ray. D^{GV} shape is $1 \times 24 \times 89$ and F^{GV} is $16 \times 24 \times 89$. Decoder δ^{GV} receives as input 17 channels and outputs $28 \times 192 \times 712$. Discriminator $\mathcal{D}$ receives input of $20 \times 192 \times 712$ and outputs $1 \times 192 \times 712$. Stage 1 output is resized, center-cropped and zero-padded to 512×512. Stage 2 NeuralFloors++ (SD) receives input of shape $29 \times 512 \times 512$, downscaled to $29 \times 256 \times 256$ where ϕ encodes to $4 \times 64 \times 64$ before it is concatenated with the noise, and finally outputs an image of size $3 \times 512 \times 512$. Stage 2 NeuralFloors++ (T2I-adapter) receives inputs at the native 192×712 output resolution of Stage 1 and outputs an image of size 192×712. Style encoder ε^{style} receives the original KITTI-360 image of shape $3 \times 376 \times 1408$ and outputs $4 \times 47 \times 176$ which is upscaled bi-linearly to $4 \times 376 \times 1408$. The bank of style embeddings consists of $269,843$ latent codes organised by class.

E. Training details

We empirically set the weights α of rare classes *[bicycle, person, rider, pole, traffic sign]* to *[4.0, 5.0, 4.0, 4.0, 4.0]*. We use batch size 1 and Adam optimiser. Each stage model is trained individually on 4 NVIDIA V100 GPUs with 32 GB of VRAM. Inference takes 0.33s, 29s and 5.5s per observation for Stage 1, Stage 2 NeuralFloors++ (SD) and Stage 2 NeuralFloors++ (T2I-adapter) respectively, on a single GPU. Stage 1 is trained with a learning rate of 10^{-4}, uses the DeepSpeed [38] library and has 117 M trainable parameters. Stage 2 of NeuralFloors++ (SD) is trained with a learning rate of 10^{-4}, $t = 50$ diffusion steps at inference and has 943.2 M params, while stage 2 NeuralFloors++ (T2I-adapter) uses $t = 20$ steps and has 2.737 B params. We use the following weights to scale losses: $\lambda_{\mathcal{L}_S} = 10$, $\lambda_{\mathcal{L}_J} = 10$, $\lambda_{\mathcal{L}_C} = \lambda_{\mathcal{L}_O} = 100$, $\lambda_{\mathcal{L}_D} = \lambda_{\mathcal{L}_{D_+}} = 1$, $\lambda_{\mathcal{L}_A} = 1$.

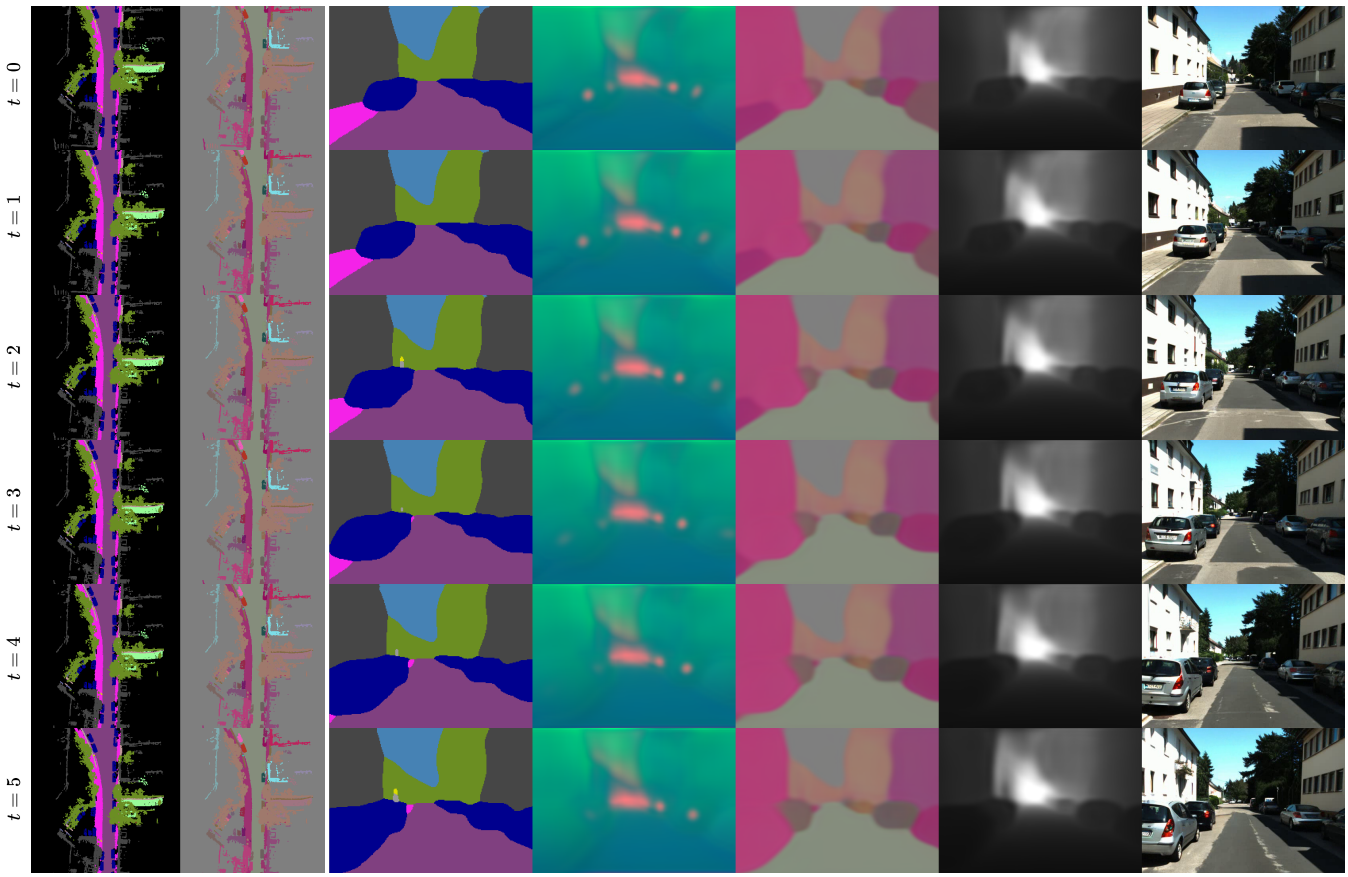

Fig. 3: Consecutive frame predictions from BEV inputs with NeuralFloors++ (T2I-adapter) model.

Model	mIoU ↑	mIoU-align ↑	RMSE ↓	FID ↓ 64^2	FID ↓ 256^2	FID ↓ 512^2	KID ↓ 64^2	KID ↓ 256^2	KID ↓ 512^2	FVD ↓ 64^2	FVD ↓ 256^2	FVD ↓ 512^2
GSN	-	-	-	203.42	-	-	0.2512(46)	-	-	1254.77	-	-
CC3D pre-trained	-	10.12	-	55.24	104.59	-	0.0396(11)	0.0905(18)	-	486.88	1043.94	-
CC3D trained	-	11.47	-	97.05	125.85	-	0.0503(28)	0.0853(25)	-	491.80	1004.41	-
NeuralFloors	32.06	28.11	6.897	42.23	66.50	65.81	0.0266(11)	0.0351(12)	0.0322(12)	454.95	834.95	868.29
NeuralFloors++ (SD)	**35.82**	29.15	**6.730**	25.49	36.51	48.63	0.0157(11)	0.0166(8)	0.0229(9)	**206.90**	603.81	600.29
NeuralFloors++ (T2I-adapter)	**35.82**	**29.33**	**6.730**	**20.60**	**29.41**	**42.55**	**0.0149(11)**	**0.0162(10)**	**0.0219(11)**	208.57	**386.07**	**397.27**

TABLE I: Baselines comparison regarding segmentation, depth and perceptual quality at different image sizes. Parentheses indicate smallest digit uncertainty – e.g. 0.0396 ± 0.0011 in the second row for KID.

V. Results

In Tab. I, as evaluated by FID (203.42), KID (0.2512) and FVD (1254.77), GSN seems to be the least performing model, as also emphasized in [3] and [1]. Compared to NeuralFloors, our proposed model performs better in terms of semantic correctness, as both mIoU and mIoU-alignment have improved from 32.06 to 35.82 and from 28.11 to 29.33 respectively, while RMSE dropped from 6.897 to 6.730, highlighting the benefit of improved complex data and multi-view training. In terms of perceptual quality of both the generated images and video – FID, KID and FVD are improved at all resolutions. In terms of semantic structure, since CC3D [3] does not output ground-view segmentation map nor it is trained for the output to follow a particular ground-view semantic structure, we can only compare the segmentation of the images from the off-the-shelf segmenter. In this case, both CC3D pre-trained and CC3D trained models have close mIoU-alignment of 10.12 and 11.47 respectively, which are much lower than our 29.33. Regarding perceptual quality, metrics also significantly improve, especially at the 256^2 resolution. While these results are reported for our data, which makes use of KITTI-360 to the full extent, CC3D [3] also report an FID of 65.6 at the 256^2 resolution on their curated dataset, where turning cases are removed.

As it can be noted in Fig. 3, NeuralFloors++ (T2I-adapter) can synthesise consecutive frames of high quality and diverse modalities. Additionally, in Figs. 4 and 5 we present qualitative results of style control and compositionality.

Fig. 4: We demonstrate style control by randomly sampling 2 sets of latent codes (*2nd* and *4th* columns) for the objects, using the same BEV semantic map. In the generated ground-view images, the layout of objects is consistent, while the style is distinct.

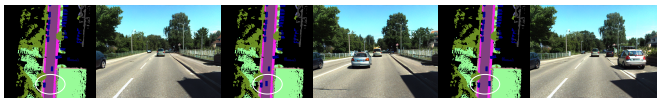

Fig. 5: We demonstrate compositionality and editability capabilities by manipulating the input map. We add and move a car from left to right in the input BEV semantic map re-rendered the scene.

VI. Ablation Study

We conduct an ablation study to evaluate and justify the choice of our final model, by iteratively adding new features to a base case, with + indicating incremental additions to the settings of the previous row.

Since we use a real-world dataset, some classes are less common. Thus, in Tab. II, we analyse the effects of assigning a larger weight to under-represented classes (small-class weighting), compared to the baseline with equal weighting (equal weighting) and observe an improvement in mIoU from 32.69 to 33.74. We additionally investigate whether a model with larger capacity (F^{GV} from $4 \times 24 \times 89$ to $16 \times 24 \times 89$) leads to improvements in mIoU (+ larger model).

Model	mIoU ↑
equal weighting	32.69
small-class weighting	33.74
+ larger model	33.89

TABLE II: Single-view ablations group 1.

In Tab. III, starting from small-class weighting as a base setup, we find that both enforcing BEV semantic reconstruction (+ BEV segmentation reconstruction) and adding BEV instance information (+ input BEV instance map) lead to an increase in mIoU. We additionally generate an upscaled depth map from decoder δ^{GV} (+ depth upscaled). The addition of this output leads to a reduction in mIoU, but represents an important component of the set of generated ground-view data, so we include it in the final model setup.

Model	mIoU ↑
small-class weighting	33.74
+ BEV segmentation reconstruction	34.20
+ input BEV instance map	34.76
+ depth upscaled	33.65

TABLE III: Single-view ablations group 2.

In Tab. IV, starting from small-class weighting, we check the effects of adding BEV style maps as input, and style map reconstruction (+ style input & reconstruction). We observe a drop in performance (from 33.74 to 25.97) and investigate whether this is due to a model bottleneck by employing a larger model (F^{GV} from $4 \times 24 \times 89$ to $16 \times 24 \times 89$) (+ larger model). We observe a recovery in mIoU (34.49) and notice that the initial base setup did not suffer from the same drawback (+ larger model in Tab. II, 33.89). We conclude that this is an indication of a bottleneck when diversifying the output signal. Finally, we analyse the performance of a single-view model with all components introduced so far (+ all components): small-class weighting, BEV segmentation reconstruction, input BEV instance map, upscaled depth prediction, segmentation discriminator, BEV style map input and ground-view style map reconstruction on a larger model, leading to the best performance in this ablation group (34.55).

Model	mIoU ↑
small-class weighting	33.74
+ style input & reconstruction	25.97
+ larger model	34.49
+ all components	34.55

TABLE IV: Single-view ablations group 3.

In Tab. V, starting from small-class weighting setup, we first train a simple multi-view model (+ multi-view) and set a new baseline of mIoU 35.36. As outlined in Sec. IV-A, our multi-view data is comprised of observations from the original poses, which are dense and contain the class sky, and random observations around these, which come from sparse LiDAR point clouds, with sky segmentation being absent. We empirically observe that our model learns to output dense segmentation maps with sky present for the original poses but fails to synthesize sky class in any of the random observations. To mitigate this, for each observation set, we randomly select one of the camera poses and the BEV associated with it, and express the poses of all other cameras with respect to this new reference camera (+ randomised). As mIoU cannot be computed for class sky in the random views, we conduct a qualitative inspection. We then add all of the single view components (+ all components), which leads to an mIoU of 34.93. Finally, we add the segmentation discriminator (+ segmentation discriminator), leading to a final mIoU of 35.82. We select this as our final model.

Model	mIoU ↑
small-class weighting	33.74
+ multi-view	35.36
+ randomised	35.46
+ all components	34.93
+ segmentation discriminator	35.82

TABLE V: Multi-view ablations.

In Tab. VI, we start with a baseline of FVD 868.29 based on the previous approach (NeuralFloors). We test the benefit of additionally conditioning the synthesis model via instance-wise style maps (NeuralFloors + style) and note an improvement in FVD. As consecutive frames are highly correlated, we take advantage of this characteristic by conditioning the synthesis of the current frame $\hat{I}_t^{GV}$ on the previous generated frame $I_{past}^{GV} = \hat{I}_{t-1}^{GV}$ (NeuralFloors + AR(1)). However, as this ablation is evaluated on the output data from stage 1, style and content drifts that are present in initial frames possibly affect the synthesis of subsequent frames, leading to less perceptually-pleasing results. We test a similar setup where the extra conditional input frame is randomly chosen between $t-1$ and $t-8$ (NeuralFloors + AR(*)) and note an improvement in FVD to 692.36. Finally, we evaluate both a past generated frame and the style map conditioning (NeuralFloors + style + AR(*)), and observe an

improved FVD of 600.29. We select this as our final model.

Model	FVD ↓
NeuralFloors	868.29
NeuralFloors + style	668.48
NeuralFloors + AR(1)	1167.64
NeuralFloors + AR(*)	692.36
NeuralFloors + style + AR(*)	600.29

TABLE VI: Stage 2 ablations evaluated at 512^2, on the output of stage 1.

VII. Conclusions

We have presented improvements in multiple-view consistency, style manipulation, and fine object-level control when synthesising ground-view images from BEV representations. These contributions have lead to improved realism and alignment for the generated images and associated modalities. With the addition of scene compositionality and style control, the proposed system has important applications in training AD downstream tasks without the need of a simulator. It is worth noting that our system may encounter challenges when dealing with tunnels or other structures that obscure the scene entirely. Although in this study we focus on 2D representations for simplicity and useability, other BEV inputs (depth or height) can be used to improve performance.

References

[1] V. Muşat, D. D. Martini, M. Gadd, and P. Newman, "Neuralfloors: Conditional street-level scene generation from bev semantic maps via neural fields," *IEEE Robotics and Automation Letters*, 2024.

[2] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *CVPR*, 2018.

[3] S. Bahmani, J. J. Park, D. Paschalidou, X. Yan, G. Wetzstein, L. J. Guibas, and A. Tagliasacchi, "Cc3d: Layout-conditioned generation of compositional 3d scenes," *ArXiv*, vol. abs/2303.12074, 2023.

[4] Y. Liao, J. Xie, and A. Geiger, "Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *ArXiv*, vol. abs/2109.13410, 2021.

[5] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *CoRL*, 2017.

[6] Y. Yang, F.-Y. Sun, L. Weihs, E. VanderBilt, A. Herrasti, W. Han, J. Wu, N. Haber, R. Krishna, L. Liu, *et al.*, "Holodeck: Language guided generation of 3d embodied ai environments," in *CVPR*, 2024.

[7] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

[8] H. Alhaija, S. Mustikovela, A. Geiger, and C. Rother, *Geometric Image Synthesis*, 2019.

[9] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, G. Liu, A. Tao, J. Kautz, and B. Catanzaro, "Video-to-video synthesis," *arXiv preprint arXiv:1808.06601*, 2018.

[10] X. Qi, Q. Chen, J. Jia, and V. Koltun, "Semi-parametric image synthesis," in *CVPR*, 2018.

[11] V. Musat, D. D. Martini, M. Gadd, and P. Newman, "Depth-sims: Semi-parametric image and depth synthesis," *International Conference on Robotics and Automation*, 2022.

[12] P. Henzler, N. J. Mitra, and T. Ritschel, "Escaping plato's cave: 3d shape from adversarial rendering," in *The IEEE International Conference on Computer Vision*, October 2019.

[13] T. Nguyen-Phuoc, C. Li, L. Theis, C. Richardt, and Y.-L. Yang, "Hologan: Unsupervised learning of 3d representations from natural images," in *ICCV*, 2019.

[14] T. H. Nguyen-Phuoc, C. Richardt, L. Mai, Y. Yang, and N. Mitra, "Blockgan: Learning 3d object-aware scene representations from unlabelled images," *Advances in neural information processing systems*, 2020.

[15] M. Niemeyer and A. Geiger, "Giraffe: Representing scenes as compositional generative neural feature fields," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2021.

[16] Y. Xue, Y. Li, K. Singh, and Y. Lee, "Giraffe hd: A high-resolution 3d-aware generative model," in *CVPR*, 2022.

[17] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of StyleGAN," in *Proc. CVPR*, 2020.

[18] T. DeVries, M. Á. Bautista, N. Srivastava, G. W. Taylor, and J. M. Susskind, "Unconstrained scene generation with locally conditioned radiance fields," *CoRR*, vol. abs/2104.00670, 2021.

[19] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. D. Mello, O. Gallo, L. Guibas, J. Tremblay, S. Khamis, T. Karras, and G. Wetzstein, "Efficient geometry-aware 3D generative adversarial networks," in *arXiv*, 2021.

[20] Q. Zhang, Y. Xu, Y. Shen, B. Dai, B. Zhou, and C. Yang, "BerfScene: Generative novel view synthesis with 3D-aware diffusion models," in *arXiv*, 2023.

[21] C. Lin, H.-Y. Lee, W. Menapace, M. Chai, A. Siarohin, M.-H. Yang, and S. Tulyakov, "Infinicity: Infinite-scale city synthesis," 2023.

[22] A. Swerdlow, R. Xu, and B. Zhou, "Street-view image generation from a bird's-eye view layout," *ArXiv*, vol. abs/2301.04634, 2023.

[23] K. Yang, E. Ma, J. Peng, Q. Guo, D. Lin, and K. Yu, "Bevcontrol: Accurately controlling street-view elements with multi-perspective consistency via bev sketch layout," *ArXiv*, vol. abs/2308.01661, 2023.

[24] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *IEEE International Conference on Computer Vision*, 2023.

[25] C. Mou, X. Wang, L. Xie, J. Zhang, Z. Qi, Y. Shan, and X. Qie, "T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models," *ArXiv*, vol. abs/2302.08453, 2023.

[26] X. Wang, T. Darrell, S. S. Rambhatla, R. Girdhar, and I. Misra, "Instancediffusion: Instance-level control for image generation," *arXiv preprint arXiv:2402.03290*, 2024.

[27] X. Wang, H. Yuan, S. Zhang, D. Chen, J. Wang, Y. Zhang, Y. Shen, D. Zhao, and J. Zhou, "Videocomposer: Compositional video synthesis with motion controllability," in *International Conference on Neural Information Processing Systems*, 2023.

[28] O. Bar-Tal, H. Chefer, O. Tov, C. Herrmann, R. Paiss, S. Zada, A. Ephrat, J. Hur, Y. Li, T. Michaeli, O. Wang, D. Sun, T. Dekel, and I. Mosseri, "Lumiere: A space-time diffusion model for video generation," *ArXiv*, vol. abs/2401.12945, 2024.

[29] B. Cheng, M. D. Collins, Y. Zhu, T. Liu, T. S. Huang, H. Adam, and L.-C. Chen, "Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation," in *CVPR*, 2020.

[30] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," 2021.

[31] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[32] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[33] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *International Conference on Neural Information Processing Systems*, 2017.

[34] M. Bińkowski, D. J. Sutherland, M. Arbel, and A. Gretton, "Demystifying MMD GANs," in *International Conference on Learning Representations*, 2018.

[35] T. Unterthiner, S. van Steenkiste, K. Kurach, R. Marinier, M. Michalski, and S. Gelly, "Fvd: A new metric for video generation," in *DGS@ICLR*, 2019.

[36] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *ECCV*, 2018.

[37] E. Schönfeld, V. Sushko, D. Zhang, J. Gall, B. Schiele, and A. Khoreva, "You only need adversarial supervision for semantic image synthesis," in *International Conference on Learning Representations*, 2021.

[38] J. Rasley, S. Rajbhandari, O. Ruwase, and Y. He, "Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters," in *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.