Multi-Modal 3D GAN for Urban Scenes

Loïck Chambon[1] **Mickael Chen**[1] **Tuan-Hung Vu**[1,2] **Alexandre Boulch**[1,2]
Andrei Bursuc[1,2] **Matthieu Cord**[1,3] **Patrick Pérez**[1,2]
[1]Valeo.ai [2]ASTRA-Vision, INRIA [3]SCAI, Sorbonne University
{firstname.lastname}@valeo.com

Abstract

Recently, a number of works have explored training 3D-aware Generative Adversarial Networks (GANs) that include a neural rendering layer in the generative pipeline. Doing so, they succeed in building models that can infer impressive 3D information while being trained solely on 2D images. However, they have been mostly applied to images centered around an object. Transitioning to driving scenes is still a challenge, as not only the scenes are open and more complex, but also one usually does not have access to as many diverse viewpoints. Typically only the front camera view is available. We investigate in this work how 3D GANs are amenable are for such a setup, and propose a method to leverage information from LiDAR sensors to alleviate the detected issues.

1 Introduction

Generative adversarial networks (GANs) [5] have recently found their application in various downstream tasks, such as training data generation [1, 21], domain translation [9, 22], image edition [6, 13], or counterfactual explanation [8, 11]. Although they generate high quality views of centered objects, notably faces [10], GANs still struggle in generating complex images such as urban scenes. The two main challenges with urban scenes are **(1)** the composite nature that requires multi-object generation and **(2)** the geometric plausible constraints. Recent work [15] show that generators fail to acquire a good understanding of the 3D geometry and take shortcuts to resolve the task at the image level instead. Compositional bottom-up frameworks [4, 7, 15] aim to mitigate this issue. Such frameworks are better at generating scenes but at the expense of higher computational requirements, which significantly hinders their scalability to generate bigger and more complex scenes with more objects.

Recently, following the popularization of Neural Radiance Fields [14] (NeRF), a number of works have tried to incorporate 3D priors in image GANs by adding a differentiable renderer in the generation pipeline [2, 3, 15, 16, 18, 20]. Doing so would force the generator to learn consistent 3D representations of upstream scenes via the differentiable rendering module. The 3D-aware networks like EG3D [2], are remarkable in the sense that they are able to generate not only images from arbitrary viewpoints, but also the associated density fields, effectively proving that they have captured the 3D information solely from standard 2D image datasets like FFHQ [10]. However, so far, most have been applied only to single-object images and where camera positions relative to the object of interest are varied.

We here explore how 3D-aware NeRF-based models would behave in the urban setting. The main ingredient that makes possible 3D learning of the aforementioned models is that the objects in the datasets have been captured at various camera positions. Although existing urban datasets with images from moving cameras do satisfy the condition of having multiple camera positions, the variety of viewpoints is very limited compared to that in face datasets. Moreover, unlike FFHQ where faces can be anchored as the coordinate center to facilitate learning, urban scenes are open with no limits. Also to handle far-away objects especially the sky at infinity, we would need dedicated strategies.

Machine Learning for Autonomous Driving Workshop at the 36th Conference on Neural Information Processing Systems (NeurIPS 2022), New Orleans, USA.

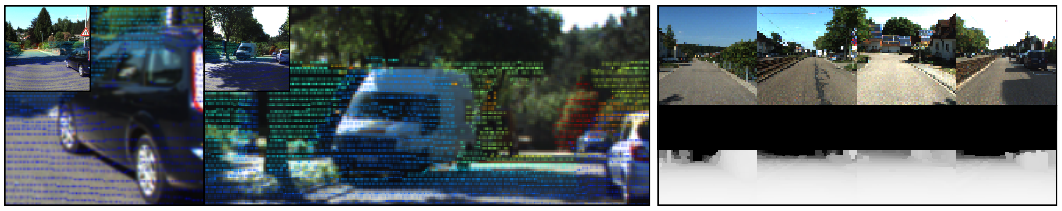
Figure 1: **Samples from the real data**. (*Left*) Zoomed in images on which 3D point clouds have been projected. The color of the points reflects the depth. We see three types of missing points: 1) the top part of the image is empty; 2) there are no points beyond 80m; 3) highly reflexive surfaces can produces missing or noisy acquisitions. We also see misalignment between the image and the point clouds, caused by either the change of viewpoints between the cameras and the LiDAR sensor, or by movement during the acquisition delay. (*Right*) RGB images with their corresponding pseudo-depth maps constructed from LiDAR point clouds.

To compensate for those difficulties, we note that acquisition vehicles tend to embark, in addition to cameras, a LiDAR sensor that provides 3D information and is aligned with the camera views. In this work, we investigate how one can efficiently incorporate the LiDAR data into the 3D-aware GAN pipeline, and demonstrate with experiments using the KITTI-360 dataset [12], that doing so does allow to alleviate the issues.

2 Method

We first succinctly describe how our 3D-aware image generator works. We then discuss how we incorporate the 3D information present in the data with the images, as depth maps. And finally, we explain how we adapt the generation pipeline to simulate those maps correctly, as required by the adversarial training objective.

3D-aware Generator. Following findings from EG3D [2], we use tri-plane as 3D representations. In this approach, a plane k is the output of a StyleGAN-2 generator [10] that we use as a vector map to associate each point $P \in \mathbb{R}^2$ to a feature vector $F_k(P)$. As the map is discrete, linear interpolation is used to handle continuous inputs. To represent a volume, we use three planes, hence the name, that each corresponds to a canonical plane in the Euclidean space. Given a 3D point (x, y, z), the feature vector $F(x, y, z)$ is built by concatenation of the vectors $F_{xy}(x, y)$, $F_{yz}(y, z)$, $F_{zx}(z, x)$ given by the maps at the location of the projection of (x, y, z) on each plane. This effectively provides a compact representation of a 3D space as any coordinates can be associated to a feature vector.

A neural renderer can transform feature vectors $F(x, y, z)$ into colors and densities that can be used to generate any RGB view via volumic rendering, as in NeRF [14].

Combining Images and LiDAR Point clouds. Integrating LiDAR point clouds into such a pipeline can be done in multiple ways, for instance by conditioning the generation to point clouds, or by adding point cloud specific modules such as a graph neural network based discriminator. Instead, we choose to project the 3D point clouds to the camera viewpoint, constructing a depth map aligned with the image. We concatenate this depth map to the channels of the RGB image, effectively building a pseudo-RGBD image. The objective is then to train a 3D-aware GAN, to generate those pseudo-RGBD images. This method stays as close as possible to a standard image GAN setup, taking full advantage of the powerful StyleGAN-2 backbone. However, because of the adversarial training the method implies, we still need a way to simulate the pseudo-depth map when generating samples. Also, because of some acquisition artifacts that are difficult to simulate, some further adaptations are necessary (see Figure 1). Those aspects are discussed in the rest of the Section.

Generating the Pseudo-Depth Map. For the real data, the pseudo-RGBD image is constructed by using the 3D points acquired by the LiDAR sensor and projecting them to the camera viewpoint (Figure 1). To mimic this pipeline in the tri-plane space, we have to simulate a virtual sensor with the same parameters as the real LiDAR sensor, compute the 3D points that would be seen by this virtual sensor, and project them to the virtual camera. Each step is fairly straightforward, given that we

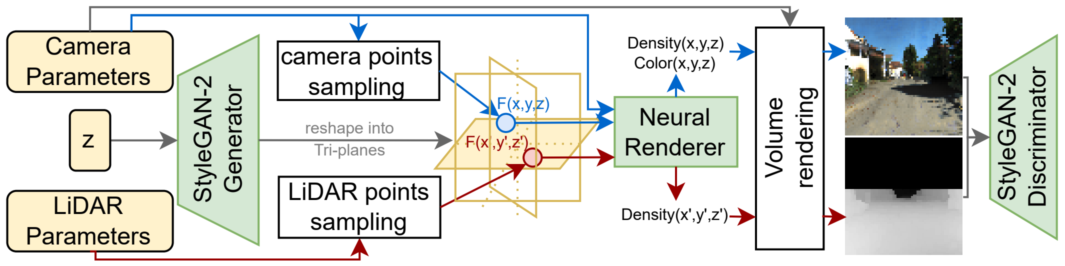

Figure 2: **Overview of the architecture**. Only the generator, the discriminator, and the neural renderer are trained. The output of a StyleGAN-2 generator is used a tri-planes representation that associates to any point (x, y, z) a feature vector $F(x, y, z)$. These feature vectors are then transformed into a density and a color value by the neural renderer. Given the parameters of a sensor, a volume renderer can thus query the color and density of all the points it needs to build an RGBD image. The pipeline for the image modality is displayed with blue and grey arrows red and grey arrows are used for the pseudo-depth modality. Note that during training, to conform with the way the real data is constructed, the depth image is rendered from the viewpoint of the camera, but sampling in the tri-planes is from the perspective of the LiDAR sensor.

already have a neural renderer that can output densities for any coordinates, and a volume renderer that can aggregate the densities along a ray to output a pseudo-depth. However, knowing which rays to sample to simulate a realistic LiDAR sensor is not trivial. Unlike cameras, the directions of the rays are not organized uniformly on a grid, as they are affected by movements during the acquisition. Still, for simplicity, we opted to use a fixed uniform grid for the directions of the simulated LiDAR rays. This causes a slight discrepancy between the two sampling systems (real and simulated) that we minimize by projecting the real 3D points on their closest simulated ray. We then reproject the points to the camera viewpoint in order to have an aligned pseudo-RGBD image.

Final tweeks. We noticed that the model could suffer from instability during training. To solve the issue, we randomly dropped 80% of the depth maps during training, for both generated and real dataset data, before feeding the images to the discriminator. Moreover, to make sure to remove any remaining unwanted artifacts that could be picked up by the discriminator, we apply a min-pool filter on the depth images. We also mask out its top 45% part, for which we have no real LiDAR acquisition. Samples of the constructed data that is seen by the discriminator are shown in Figure 1, while the full pipeline is displayed in Figure 2.

3 Experiments

For our experiments, we use the KITTI-360 dataset [12]. The dataset includes $152,946$ images from front stereo cameras, that we center crop and resize to 64x64 resolution, as well as point clouds from LiDAR scans. Note that while the dataset has aligned images from two cameras, we do not use this information.

We compare our approach to the baseline 3D GAN that does not use 3D points. Without the LiDAR information, the generated depth maps (Figure 3) show that the model struggles figuring out the correct relative positions in the scene. The sky is put in the front while the road serves as background and the objects are all set within a small range of distances, far away. When the LiDAR information is added (Figure 4), the depth maps become much sharper and contrasted, displaying the whole range of values that are present in the original data (Figure 1). The road is correctly placed on the horizontal plane, and the objects are correctly distributed along the depth axis. Only the sky is still misplaced, but not as dramatically.

We complete our observations with an assessment of the quality of the generated images using the Fréchet Inception Distance (FID). We achieve a FID of 12.2 without LiDAR and 14.7 with LiDAR.

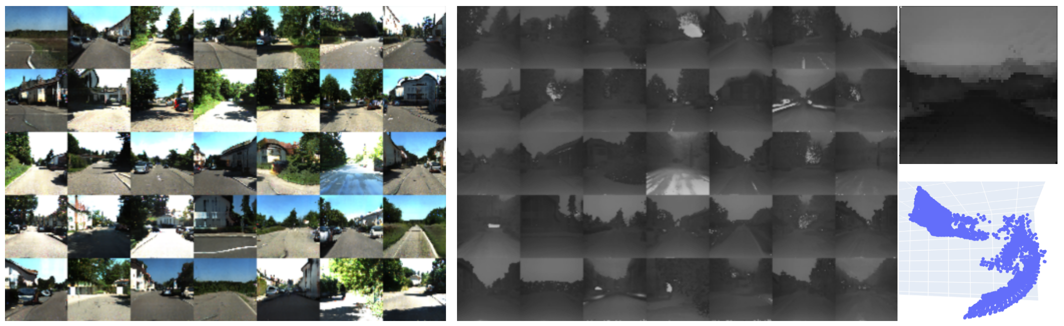

Figure 3: Generated samples from the baseline 3D GAN trained without LiDAR information. (*Left*) Generated RGB images. (*Center*) Corresponding depth maps. (*Right*) A generated depth map with a side view of the associated point cloud (the camera is placed on the left and looking right). For depth images, black is far, white is near.

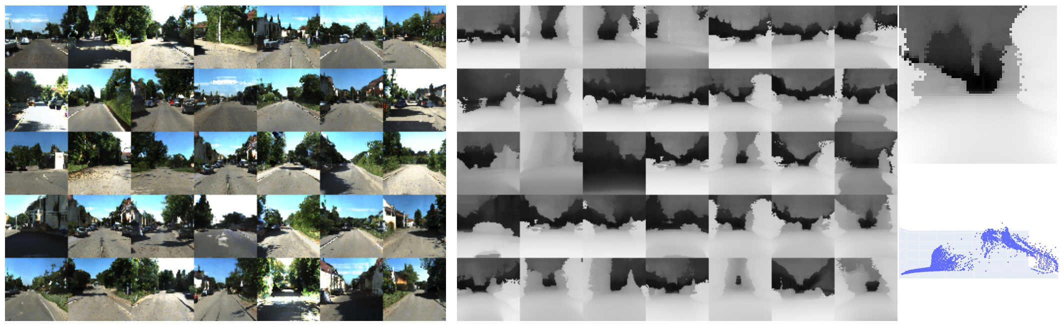

Figure 4: Samples from the model trained with pseudo-depth maps, organized as in Figure 3.

4 Conclusion

We showed that a naïve adaptation of 3D-aware GANs to driving datasets would not be able to correctly capture the 3D layout of a scene. In particular, the geometry does not constrain the model to place the road and the sky correctly. We then proposed a method to take advantage of the LiDAR modality available in many datasets. With this additional information, our method is able to get a better understanding of the 3D world and produces more accurate depth maps.

One remaining challenge is to handle the placement of the sky. Indeed, its position at a virtually infinite distance makes it particularly difficult to handle without guidance. Also, since LiDAR sensors are generally oriented towards the ground, they are not useful to solve this specific problem. To address it, future work can take inspiration from approaches that have successfully applied NeRF to urban scenes. For instance, Urban Radiance Field [17] takes advantage of semantic annotations. Then, it remains to be seen how the method scales-up to higher resolutions. De facto standard method, consisting of adding an upsampling module in the pipeline [2, 16], or a patch-based training approach [19] can be considered.

Acknowledgement. This work was performed using HPC resources from GENCI–IDRIS (Grant 2021-AD011013664).

References

[1] Victor Besnier, Himalaya Jain, Andrei Bursuc, Matthieu Cord, and Patrick Pérez. This dataset does not exist: Training models from generated images. In *ICASSP*, 2020. 1

[2] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*, 2022. 1, 2, 4

[3] Eric R. Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. Pi-GAN: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *CVPR*, 2021. 1

[4] Dave Epstein, Taesung Park, Richard Zhang, Eli Shechtman, and Alexei A. Efros. BlobGAN: Spatially disentangled scene representations. In *ECCV*, 2022. 1

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In *NeurIPS*, 2014. 1

[6] Erik Härkönen, Aaron Hertzmann, Jaakko Lehtinen, and Sylvain Paris. Ganspace: Discovering interpretable GAN controls. In *NeurIPS*, 2020. 1

[7] Dor Arad Hudson and Larry Zitnick. Compositional transformers for scene generation. In *NeurIPS*, 2021. 1

[8] Paul Jacob, Éloi Zablocki, Hedi Ben-Younes, Mickaël Chen, Patrick Pérez, and Matthieu Cord. STEEX: steering counterfactual explanations with semantics. In *ECCV*, 2022. 1

[9] Liming Jiang, Changxu Zhang, Mingyang Huang, Chunxiao Liu, Jianping Shi, and Chen Change Loy. TSIT: A simple and versatile framework for image-to-image translation. In *ECCV*, 2020. 1

[10] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *CVPR*, 2020. 1, 2

[11] Oran Lang, Yossi Gandelsman, Michal Yarom, Yoav Wald, Gal Elidan, Avinatan Hassidim, William T. Freeman, Phillip Isola, Amir Globerson, Michal Irani, and Inbar Mosseri. Explaining in style: Training a GAN to explain a classifier in stylespace. In *ICCV*, 2021. 1

[12] Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *arXiv preprint arXiv:2109.13410*, 2021. 2, 3

[13] Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. Editgan: High-precision semantic image editing. In *NeurIPS*, 2021. 1

[14] 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. In *ECCV*, 2020. 1, 2

[15] Michael Niemeyer and Andreas Geiger. GIRAFFE: representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 1

[16] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. StyleSDF: High-Resolution 3D-Consistent Image and Geometry Generation. In *CVPR*, 2022. 1, 4

[17] Konstantinos Rematas, Andrew Liu, Pratul P. Srinivasan, Jonathan T. Barron, Andrea Tagliasacchi, Tom Funkhouser, and Vittorio Ferrari. Urban radiance fields. In *CVPR*, 2022. 4

[18] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: generative radiance fields for 3d-aware image synthesis. In *NeurIPS*, 2020. 1

[19] Ivan Skorokhodov, Sergey Tulyakov, Yiqun Wang, and Peter Wonka. EpiGRAF: Rethinking training of 3d gans. *arXiv preprint arXiv:2206.10535*, 2022. 4

[20] Yinghao Xu, Sida Peng, Ceyuan Yang, Yujun Shen, and Bolei Zhou. 3D-aware image synthesis via learning structural and textural representations. In *CVPR*, 2022. 1

[21] Yuxuan Zhang, Huan Ling, Jun Gao, Kangxue Yin, Jean-Francois Lafleche, Adela Barriuso, Antonio Torralba, and Sanja Fidler. Datasetgan: Efficient labeled data factory with minimal human effort. In *CVPR*, 2021. 1

[22] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 1