EPFL Semester Project Report

Yanbo Xu
yanbo.xu@epfl.ch

Abstract

This project aims at studying the 3D spatial property of latent space in 3D GAN, and possible disentangled representation for such latent space.

1. Introduction

The advancement of Generative Adversarial Networks (GAN) [4] has contributed to many important tasks. However, most usages of GAN are still confined in 2D scenario [[6], [7], [9], [8]] and limitations including multi-view consistency of generated results restrict the potential applications.

With the recent advancement of implicit neural rendering represented by NeRF [10], it has been shown that 3D consistency and geometric information could be well learned. By combining such a technique with GAN, it is possible to generate images with 3D propriety [[5], [2], [3]].

However, compared with 2D GANs, fields in 3D GAN such as disentanglement and truly 3D-based latent space are under studied. Since a disentangled structure and well-formulated spatial latent space are crucial for many downstream tasks [[9], [1], [13]], it could be beneficial to have a representation with such properties.

For instance, an object could be represented with geometric and style information. Additionally, the observing position and angle will also affect the perceived result. Therefore, it is natural to represent an image resulting from viewing an object using three factors: geometry and style of the object, and viewing position of the observer.

2. Related Works

In 2D GAN, the stuctrue of latent spaces has been wildly studied. StyleGAN [6] utilize the latent space to determine the channel mean and variance of feature maps via adaptive instance normalization (AdaIn). StyleMapGAN [8] constructs a 2D latent space with local correspondence. Instead of using one latent space, there are generators with multiple latent space with different information. SNI [1] and DAT [9] include two latent spaces, controlling structural and style information, respectively.

Despite the high image quality achieved by 2D GANs, viewing consistency is a challenging task. Recent development of implicit neural rendering provides potential solution to this problem. NeRF [10] is one of the most promising methods, which represents a static scene using a 5D vector-valued function. The input is the 3D location (x, y, z) and the 2D view direction (θ, ϕ), and the output is an emitted color $c = (r, g, b)$ and a volume density σ. With a given camera position, an image could be rendered by integrating the color along the ray emitted from the camera. The density will be used as the weight for the color during integration. Since the process is modeled with physical constraints, the rendered images are 3D consistent.

NeRF could only represent a static scene, and one way for generalization is using GAN. GRAF [12] combines implicit neural rendering with GAN, PiGAN [2] utilize SiREN to condition the implicit neural radiance filed on the latent space. Although guaranteed with 3D consistency, volumetric rendering requires heavy computation power and time. Therefore, the image quality of those methods could not be compared with current state-of-the-art 2D GANs.

Many recent approaches to this problem adopt hybrid structures. StyleNeRF [5] applies volume render in the early feature maps with small resolution, followed by upsampling blocks to generate high-resolution images. However, a regularizor based on NeRF is required to ensure 3D consistency during upsampling. Instead of using volume rendering in early layers, EG3D [3] performs the operation on a relatively high resolution feature map using a hybrid representation for 3D features generated by StyleGAN backbone, named tri-plane, which is capable of containing more information than an explicit structure such as voxel. StyleSDF [11] shares a similar spirit, but uses SiREN for its mapping network, and the mapped result is used as input feature map followed by a style-based generator for upsampling.

3. Method

3.1. Dual Tri-plane

A reasonable representation for objects is geometry and texture. To enable better control of those properties, we would take two tri-planes, named as geometric tri-plane Tri_G and style tri-plane Tri_S, respectively. Each tri-plane consists of three planes P_{XY}, P_{YZ}, P_{XZ}, each of which $\in \mathbf{R}^{H\times W\times C}$, where C is the number of channels in that plane.

For each point (x, y, z) in the tri-plane, the feature at that point $F_{x,y,z} \in R^C = P_{XY}(x, y) + P_{YZ}(y, z) + P_{XZ}(x, z)$, where $P_{AB}(a, b)$ is the value of point (a, b) in plane P_{AB} obtained by bilinear interpolation. The process of acquiring feature value given position is known as query $Q(x, y, z)$, where $Q(x, y, z) = F_{x,y,z}$.

We obtain the two tri-planes Tri_G and Tri_S by conditioning on two latent spaces representing geometric and style information, named as Z_G and Z_S. The latent code sampled from Z_G will be mapped using the mapping function Map_G approximated by MLPs to obtain Tri_G, similar to Tri_S. The process could be represented as follows:

$$Tri_G = Map_G(z_G), z_G \in Z_G \tag{1}$$

$$Tri_S = Map_S(z_S), z_S \in Z_S \tag{2}$$

3.2. 3D Latent Space

In most 2D GANs, the latent space $\in R^C$. Most work in 3D condition their latent codes on position. However, such conditioning processes will not guarantee consistency when rotating the camera position. In our setting, we will render the final latent space using volumetric rendering to ensure a real 3D latent space.

To be specific, we will render the final latent code $L(p, z) \in R^{H_L\times W_L\times C}$, with a camera pose p, and the latent code z $\in GorS$. For each pixel in the final latent code, a ray $r(t) = p + td$ will be obtained to render the latent code $l(r, z)$, where p is the camera origin and d is the view direction. $l(r, z)$ is the integrated value along the ray. For each point on the ray, we could get the corresponding feature $F_{x,y,z}$ using the corresponding tri-plane generated by latent code z using $Q(r(t))$. The final value for that pixel could be calculated using the volume rendering equation:

$$l(r, z) = \int_0^\infty o_z(r(t))Q(r(t))dt \in R^C,$$

$$\text{where } o_z(r(t)) = exp - \left(\int \sigma_z(r(s))ds\right)\sigma_z(r(t))$$

Note that o_z will be calculated using the geometric tri-plane and used for the rendering process on both geometric and style space.

By rendering, for a tuple of sampled values (p, z_G, z_S), we will get two latent codes L_G and $L_S \in R^{H_L\times W_L\times C}$, which will be used for further image generation.

3.3. Dual Space Generator

Given the rendered latent codes L_S and L_G, the generation process which outputs an image x could be described as:

$$x = \mathbf{G}(L_S, L_G) \tag{3}$$

To make the generation process scaleable to high resolution, we adopt an hybrid approach similar to [[5], [3]]. The idea of using one space for style and another for geometric is similar to DAT. Inspired by that, we design our generator structure as follows:

Since the style information is mostly controlled by the mean and variance of the channel, given the rendered style code $L_S \in R^{H_L\times W_L\times C}$, we calculate the mean $Mean_{L_S}$and variance Var_{L_S} of that latent code, both of which $\in R^C$. These values will be used to normalize channel-wise information using AdaIN.

On the other hand, the structural information could be controlled by pixel-wise information in the generator. Thus, for each layer of the feature map with shape $H \times W$, we will upsample or downsample the renderd geometric latent code L_G to $H \times W$ and perform pixel-wise operation on the feature map.

4. Experiments

4.1. Tri-plane for Single Scene Fitting

To verify the capacity of tri-plane, we use a single tri-plane to fit a static scene as in NeRF, which utilizes one tri-plane for one scene. Besides the tri-plane, positional encoding and fine network also play an important role in rendering. As shown in Fig. 1, if rendered without the positional encoding Fig. 1(a), the image will lack fine details, which is similar to the conclusion in NeRF [10]. The fine network could provide better detail information Fig. 1(b), but is less effective than positional encoding (Fig. 1(c)).

4.2. Dual Tri-Plane for Single Scene Fitting

We wish to have a more disentangled representation for a single scene. Furthermore, NeRF [10] renders an image from occupancy and rgb information, representing geometric and style information, respectively. Therefore, we utilize two separate two tri-planes to render an image. From Fig. 1(e), the result using dual tri-planes is the best even without the fine network. This shows that having an additional tri-plane improves the capacity.

Additionally, since two tri-planes control different information, it is possible to fix one plane and change another. As shown in Fig. 2, when changing geometric tri-plane,

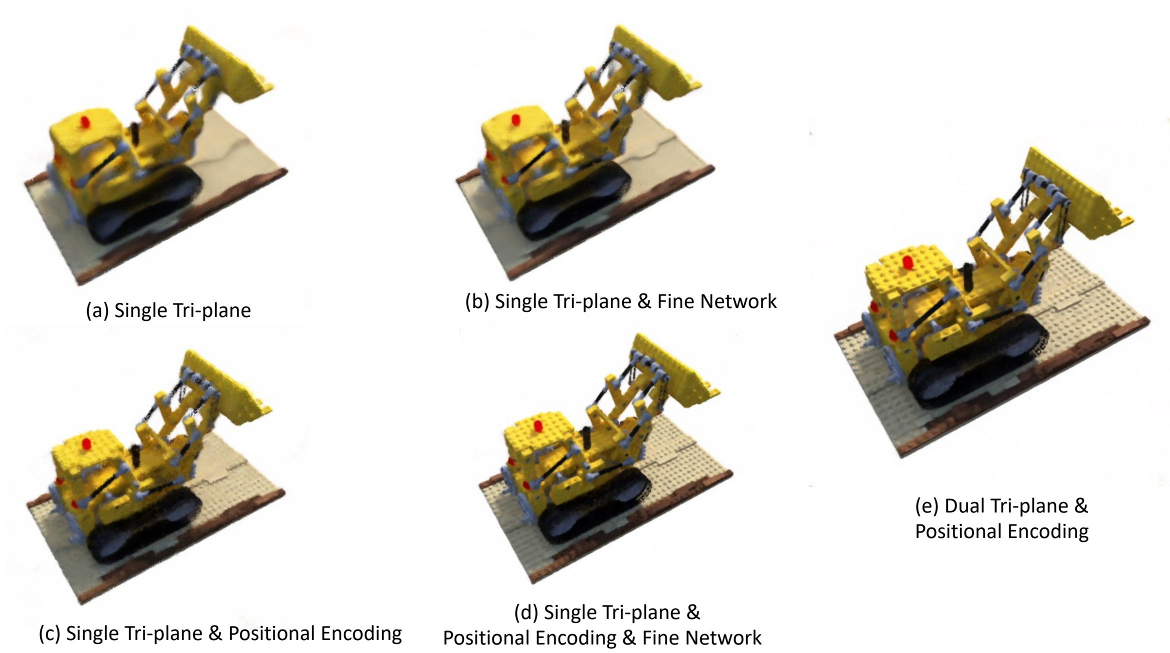

Figure 1. Single Scene fitting using tri-plane(s).

the shape of renderd image will change (structure, transparency). When changing the style tri-plane, only color will change.

Since the the network is only used to fit one scene, the results are not semantic meaningful when changing the codes. However, this experiment shows the possibility of disentangle a scene, and it also shows that the capacity of network benefits from adding another tri-plane.

4.3. Dual Tri-plane on GAN

To make the tri-planes semantically meaningful, generalization using large dataset is one solution. Therefore, we add tri-planes to styleGAN2 [7] framework. Fig. 3 shows the generated images trained on CelebA with resolution 128.

By swapping the camera position, geometric and style latent, we could see how each parameter influences the result.

The current result is not optimal. However, swapping style code could give reasonable result. Fig. 4 shows that the head pose, expression, and so on remain similar when changing the style code.

5. Further Plans

The structure in Sec. 4.3 is not optimal for now. We will continue on working on that part.

We plan to conduct experiments with our 3D latent space on 2D frameworks including single-space StyleMapGAN, and dual-space DAT. Other types of GAN architecture might also be experimented with. Additionally, pose regularization is crucial for our task, which will be added to the current framework. Our goal is to first study the effect of 3D latent space, then make it more disentangled. Finally, we could extend our framework to higher image quality such that it could be used for practical applications.

References

[1] Y. Alharbi and P. Wonka. Disentangled image generation through structured noise injection. In *CVPR*, 2020. 1

[2] E. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proc. CVPR*, 2021. 1

[3] 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 *CVPR*, 2022. 1, 2

[4] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *NeurIPS*, 27, 2014. 1

[5] J. Gu, L. Liu, P. Wang, and C. Theobalt. Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In *International Conference on Learning Representations*, 2022. 1, 2

[6] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1

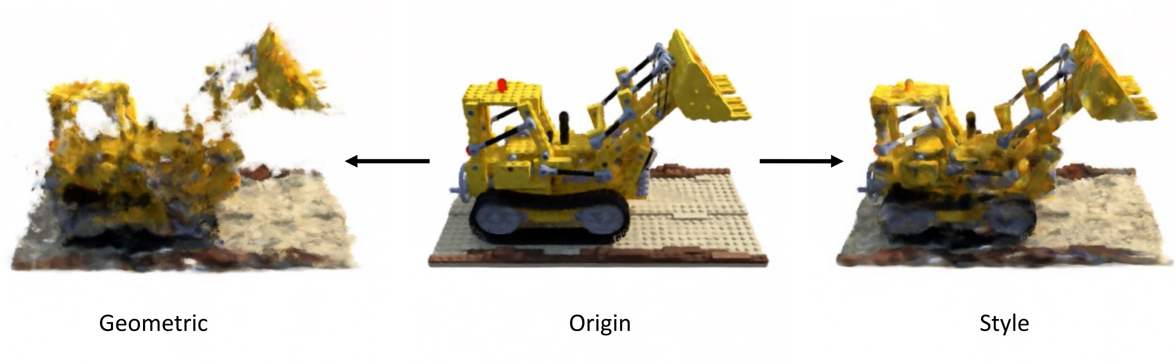

Figure 2. Interpolation of two tri-planes

Figure 3. Face randomly generated using our GAN model

Figure 4. Results of swapping the style code. All images share same camera position and geometric code.

[7] 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. 1, 3

[8] H. Kim, Y. Choi, J. Kim, S. Yoo, and Y. Uh. Exploiting spatial dimensions of latent in gan for real-time image editing. In *CVPR*, 2021. 1

[9] G. Kwon and J. C. Ye. Diagonal attention and style-based gan for content-style disentanglement in image generation and translation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13980–13989, 2021. 1

[10] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2

[11] R. Or-El, X. Luo, M. Shan, E. Shechtman, J. J. Park, and I. Kemelmacher-Shlizerman. StyleSDF: High-Resolution 3D-Consistent Image and Geometry Generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13503–13513, June 2022. 1

[12] K. Schwarz, Y. Liao, M. Niemeyer, and A. Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1

[13] Y. Xu, Y. Yin, L. Jiang, Q. Wu, C. Zheng, C. C. Loy, B. Dai, and W. Wu. TransEditor: Transformer-based dual-space GAN for highly controllable facial editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 1