3D GENERATION ON IMAGENET

Anonymous authors
Paper under double-blind review

ABSTRACT

All existing 3D-from-2D generators are designed for well-curated single-category datasets, where all the objects have (approximately) the same scale, 3D location and orientation, and the camera always points to the center of the scene. This makes them inapplicable to diverse, in-the-wild datasets of non-alignable scenes rendered from arbitrary camera poses. In this work, we develop *3D generator with Generic Priors (3DGP)*: a 3D synthesis framework with more general assumptions about the training data, and show that it scales to very challenging datasets, like ImageNet. Our model is based on three new ideas. First, we incorporate an *inaccurate* off-the-shelf depth estimator into 3D GAN training via a special depth adaptation module to handle the imprecision. Then, we create a flexible camera model and a regularization strategy for it to learn its distribution parameters during training. Finally, we extend the recent ideas of transferring knowledge from pretrained classifiers into GANs for patch-wise trained models by employing a simple distillation-based technique on top of the discriminator. It achieves more stable training than the analogs and speeds up the convergence by at least 40%. We explore our model on four datasets: SDIP Dogs 256^2, SDIP Elephants 256^2, LSUN Horses 256^2, and ImageNet 256^2 and demonstrate that 3DGP outperforms the recent state-of-the-art in terms of both texture and geometry quality.

Visualizations: https://u2wjb9xxz9q.github.io/

1 INTRODUCTION

Figure 1: Selected samples from EG3D (Chan et al., 2022) and our generator trained on ImageNet 256^2 (Deng et al., 2009). EG3D models the geometry in low resolution and renders either flat shapes (when trained with the default camera distribution) or repetitive "layered" ones (when trained with a wide camera distribution). In contrast, our model synthesizes the radiance field in the full dataset resolution and learns high-fidelity details during training. Zoom-in for a better view.

We witness remarkable progress in the domain of 3D-aware image synthesis. The community is developing new methods to improve the image quality, 3D consistency and efficiency of the generators (e.g., Chan et al. (2022); Deng et al. (2022); Skorokhodov et al. (2022); Zhao et al. (2022); Schwarz et al. (2022)). However, all the existing frameworks are designed for well-curated and aligned datasets consisting of objects of the same category, scale and global scene structure, like human or cat faces (Chan et al., 2021). Such curation requires domain-specific 3D knowledge about the object

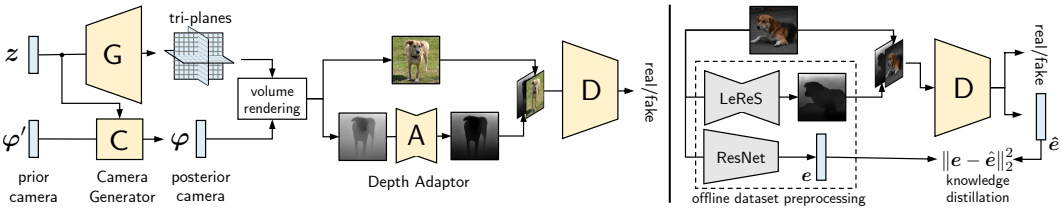


Figure 2: **Model overview**. Left: our tri-plane-based generator. To synthesize an image, we first sample camera parameters from prior distribution and pass them to the camera generator. This gives the posterior camera parameters, used to render an image and its depth map. The depth adaptor mitigates the distribution gap between the rendered and the predicted depth. Right: our discriminator receives a 4-channel color-depth pair as an input. A fake sample consists of the RGB image and its (adapted) depth map. A real sample consists of a real image and its estimated depth. Right: our two-headed discriminator predicting adversarial scores and image features for knowledge distillation.

category at hand, since one needs to infer the underlying 3D keypoints to properly crop, rotate and scale the images (Deng et al., 2022; Chan et al., 2022). This makes it infeasible to perform a similar alignment procedure for large-scale multi-category datasets that are inherently "non-alignable": there does not exist a single canonical position which all the objects could be transformed into (e.g., it is impossible to align a landscape panorama with a spoon).

To extend 3D synthesis to in-the-wild datasets, one needs a framework which relies on more universal 3D priors. In this work, we make a step towards this direction and develop a 3D generator with Generic Priors (3DGP): a 3D synthesis model which is guided only by (imperfect) depth predictions from an off-the-shelf monocular depth estimator. Surprisingly, such 3D cues are enough to learn reasonable scenes from loosely curated, non-aligned datasets, such as ImageNet (Deng et al., 2009).

Training a 3D generator on in-the-wild datasets comes with three main challenges: 1) extrinsic camera parameters of real images are unknown and impossible to infer; 2) objects appear in different shapes, positions, rotations and scales, hindering learning of the underlying 3D geometry; and 3) the dataset typically contains a lot of variation in terms of texture and structure, and is difficult to fit even for 2D generators. As shown in Fig 1 (left), state-of-the-art 3D-aware generators, such as EG3D (Chan et al., 2022), struggle to learn the proper geometry in such a challenging scenario. In this work, we develop three novel techniques to address those problems.

Learnable "Ball-in-Sphere" camera distribution. Most existing methods utilize a restricted camera model (e.g., (Schwarz et al., 2020; Niemeyer & Geiger, 2021b; Chan et al., 2021)): the camera is positioned on a sphere with a constant radius, always points to the world center and has fixed intrinsics. But diverse, non-aligned datasets violate these assumptions: e.g., dogs datasets have images of both close-up photos of a snout and photos of full-body dogs, which implies the variability in the focal length and look-at positions. Thus, we introduce a novel camera model with 6 degrees of freedom to address this variability. We optimize its distribution parameters during training and develop an efficient gradient penalty for it to prevent its collapse to delta distribution.

Adversarial depth supervision (ADS). A generic image dataset features a wide diversity of objects with different shapes and poses. That is why learning a meaningful 3D geometry together with the camera distribution is an ill-posed problem, as the incorrect scale can be well compensated by an incorrect camera model (Hartley & Zisserman, 2003), or flat geometry (Zhao et al., 2022; Chan et al., 2022). To instill the 3D bias, we provide the scene geometry information to the discriminator by concatenating the depth map of a scene as the 4-th channel of its RGB input. For real images, we use their (*imperfect*) estimates from a generic off-the-shelf monocular depth predictor (Miangoleh et al., 2021). For fake images, we render the depth from the synthesized radiance field, and process it with a shallow depth adaptor module, bridging the distribution gap between the estimated and rendered depth maps. This ultimately guides the generator to learn the proper 3D geometry.

Knowledge distillation into Discriminator. Prior works have shown benefits from transferring the knowledge from conventional 2D image encoders into a synthesis model (Sauer et al., 2022). The state-of-the-art techniques utilize pretrained image classifiers as the discriminator backbone and develop regularization strategies on top of them (Sauer et al., 2021; Kumari et al., 2022). These techniques, however, are only applicable when the discriminator has a similar input distribution compared to what the encoder was trained on. This makes it difficult to use efficient patch-based

discriminators (Skorokhodov et al., 2022) or passing depth maps as the 4-th channel to the discriminator. That is why, we develop a more general and efficient knowledge transfer strategy based on knowledge distillation. It consists in forcing the discriminator to predict features of a pre-trained ResNet50 (He et al., 2016) model, effectively transferring the knowledge into our model. This technique has just 1% of computational overhead compared to standard training, but allows to improve FID for both 2D and 3D generators by at least 40%.

First, we explore our ideas on *non-aligned* single-category image datasets: SDIP Dogs 256^2 (Mokady et al., 2022), SDIP Elephants 256^2 (Mokady et al., 2022), and LSUN Horses 256^2 (Yu et al., 2015). On these datasets, our generator achieves better image appearance (measured by FID Heusel et al. (2017)) and geometry quality than the modern state-of-the-art 3D-aware generators. Then, we train the model on all the $1,000$ classes of ImageNet (Deng et al., 2009), showing that multi-categorical 3D synthesis is possible for non-alignable data (see Fig. 1).

2 Related work

3D-aware image synthesis. Mildenhall et al. (2020) introduced Neural Radiance Fields (NeRF): a neural network-based representation of 3D volumes which is learnable from RGB supervision only. They ignited a stream of 3D-aware image and video generators (Schwarz et al., 2020; Niemeyer & Geiger, 2021b; Chan et al., 2021; Xue et al., 2022; Zhou et al., 2021; Wang et al., 2022; Gu et al., 2022; Or-El et al., 2021; Chan et al., 2022; Skorokhodov et al., 2022; Zhang et al., 2022; Xu et al., 2021; Bahmani et al., 2022), all of them being GAN-based (Goodfellow et al., 2014). Rendering low-resolution images is expensive, those works rely to either patch-wise training (e.g., Schwarz et al. (2020); Meng et al. (2021); Skorokhodov et al. (2022)), MPI-based rendering (e.g., Deng et al. (2022); Zhao et al. (2022)) or use a 2D convolutional upsampler (e.g., Niemeyer & Geiger (2021b); Gu et al. (2022)).

Learning the camera poses. NeRFs require known camera poses, obtained from multi-view stereo (Schönberger et al., 2016) or structure from motion (Schönberger & Frahm, 2016). Alternatively, a group of works have been introduced to either automatically estimate camera poses (Wang et al., 2021) or finetune them during training (Lin et al., 2021; Kuang et al., 2022). The problem we are tackling in this work is fundamentally different as it requires to learn not the camera poses from multi-view observations, but a *distribution* of poses, while having access to sparse, single-view data of diverse object categories. In this respect, the work closest to ours is CAMPARI (Niemeyer & Geiger, 2021a) as it also learns a camera distribution. Beyond this the similarity ends, as our "Ball-in-Sphere" camera distribution allows for more flexibility and supports multiple object categories.

GANs with external knowledge. Several works observed improved convergence and fidelity of GANs when using existing, generic image-based models (Kumari et al., 2022; Sauer et al., 2021; 2022; Mo et al., 2020), the most notable being StyleGAN-XL (Sauer et al., 2022), which uses a pre-trained EfficientNet (Tan & Le, 2019) followed by a couple of discriminator layers. A similar technique is not suitable in our case as pre-training a generic RGB-D network on a large scale RGB-D dataset is problematic due to the lack of data. Another notable example is FreezeD Mo et al. (2020), which proposes to distill discriminator features for GAN finetuning. Our work, on the other hand, rely on existing model for image classification.

Off-the-shelf depth guidance. GSN (DeVries et al., 2021) also concatenates depth maps as the 4-th channel of the discriminator's input, but they utilize ground truth depths, which are not available for large-scale datasets. DepthGAN (Shi et al., 2022) uses predictions from a depth estimator to guide training of a 2D GAN. Exploiting monocular depth estimators for improving neural rendering was also explored in concurrent work (Yu et al., 2022), however their goal is just geometry reconstruction. The core characteristic of our approach is taking the depth estimator imprecision into account by training a depth adaptor module to mitigate it (see §3.2).

3 Method

We build our generator on top of EpiGRAF (Skorokhodov et al., 2022) since its fast to train, achieves reasonable image quality and does not need a 2D upsampler, relying on patch-wise training instead (Schwarz et al., 2020). Given a random latent code $\boldsymbol{z}$, our generator G produces a tri-plane

representation for the scene. Then, a shallow 2-layer MLP predicts RGB color and density σ values from an interpolated feature vector at a 3D coordinate. Then images and depths are volumetrically rendered (Mildenhall et al., 2020) at any given çamera position. Differently to prior works (Chan et al., 2021; Niemeyer & Geiger, 2021b) that utilize fixed camera distribution, we sample camera from a trainable camera generator C (see §3.1). We render depth and process it via the depth adaptor (see §3.2), bridging the domains of rendered and estimated depth maps (Miangoleh et al., 2021). Our discriminator D follows the architecture of StyleGAN2 (Karras et al., 2020a), additionally taking either adapted or estimated depth as the 4-th channel. To further improve the image fidelity, we propose a simple knowledge distillation technique, that enriches D with external knowledge obtained from ResNet (He et al., 2016) (see §3.3). The overall model architecture is shown in Fig. 2.

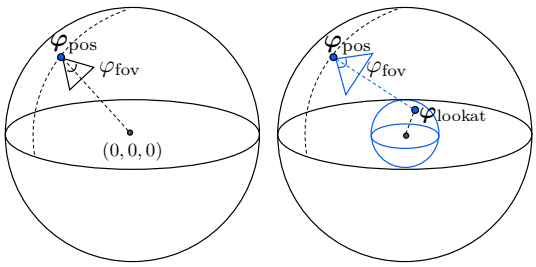
3.1 Learnable "Ball-in-Sphere" camera distribution

Limitations of Existing Camera Parameterization. The camera parameterization of existing 3D generators follows an overly simplified distribution — its position is sampled on a fixed-radius sphere with fixed intrinsics, and the camera always points to $(0, 0, 0)$. This parametrization has only two degrees of freedom: pitch and yaw ($\boldsymbol{\varphi}_{\text{pos}}$ in Fig. 3 (a)), implicitly assuming that all the objects could be centered, rotated and scaled with respect to some canonical alignment. However, natural in-the-wild 3D scenes are inherently non-alignable: they could consist of multiple objects, objects might have drastically different shapes and articulation, or they could even be represented only as volumes (like smoke). This makes the traditional camera conventions ill-posed for such data.

Learnable "Ball-in-Sphere" Camera Distribution. We introduce a new camera parametrization which we call "Ball-in-Sphere". Contrary to the standard parametrization, it has four additional degrees of freedom: the field of view φ_{fov}, and pitch, yaw and radius of the inner sphere, specifying the look-at point within the outer sphere ($\boldsymbol{\varphi}_{\text{lookat}}$ in Fig. 3 (b)). Combining with the standard parameters on the outer sphere, our camera parametrization has six degrees of freedom $\boldsymbol{\varphi} = [\boldsymbol{\varphi}_{\text{pos}} \parallel \varphi_{\text{fov}} \parallel \boldsymbol{\varphi}_{\text{lookat}}]$, where $\parallel$ denotes concatenation. Note, that compared with the standard camera, ours allows us to learn the *scale* of objects and scenes, enabling zooming into different parts of the scene as shown in Fig. 3 (b).

Instead of manually defining camera distributions, we learn the camera distribution during training for each dataset. In particular, we train the camera generator network C that takes camera parameters sampled from a sufficiently wide camera prior $\boldsymbol{\varphi}'$ and produces new camera parameters $\boldsymbol{\varphi}$. For a class conditional dataset, such as ImageNet where scenes have significantly different geometry, we additionally condition this network on the class label c and the latent code $\boldsymbol{z}$, i.e. $\boldsymbol{\varphi} = \mathsf{C}(\boldsymbol{\varphi}', \boldsymbol{z}, c)$. For a single category dataset we use $\boldsymbol{\varphi} = \mathsf{C}(\boldsymbol{\varphi}', \boldsymbol{z})$.

Camera Gradient Penalty. To the best of our knowledge, CAMPARI (Niemeyer & Geiger, 2021a) is the only work which also learns the camera distribution. It samples a set of camera parameters from a wide distribution and passes it to a shallow neural network, which produces a residual $\Delta\boldsymbol{\varphi} = \boldsymbol{\varphi} - \boldsymbol{\varphi}'$ for these parameters. However, we observed that such a regularization is too weak for the complex datasets we explore in this work, and leads to the distribution collapse (see Fig. 7). Note, that a similar observation was also made by Gu et al. (2022).



(a) Standard camera model (b) Our camera model

Figure 3: **Camera model**. (a) Conventional camera model is designed for aligned datasets and uses just 2 degrees of freedom. (b) The proposed "Ball-in-Sphere" parametrization has 4 additional degrees of freedom: field of view and the look at position. Variable parameters are shown in blue.

To prevent the camera generator from producing collapsed camera parameters, we seek a new regularization strategy. Ideally, it should prevent constant solutions, while at the same time, reducing the Lipschitz constant for C, which is shown to be crucial for stable training of generators (Odena et al., 2018). We can achieve both if we push the derivatives of the predicted camera parameters with respect to the prior camera parameters to either one or minus one,

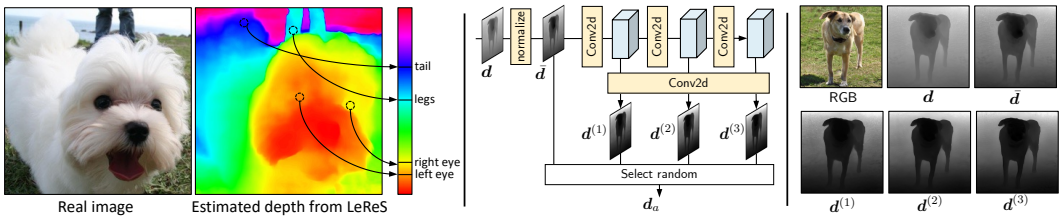


Figure 4: **Depth adapter.** Left: An example of a real image with its depth estimated by LeReS (Miangoleh et al., 2021). Note that the estimated depth has several artifacts. For example, the human legs are closer than the tail, eyes are spaced unrealistically, and far-away grass is predicted to be close. Middle: depth adapter meant to bridge the domains of predicted and NeRF-rendered depth. Right: a generated image with its adapted depth maps obtained from different layers of the adapter.

arriving at the following regularization term:

$$\mathcal{L}_{\varphi_i} = \left|\frac{\partial \varphi_i}{\partial \varphi_i'}\right| + \left|\frac{\partial \varphi_i}{\partial \varphi_i'}\right|^{-1}, \tag{1}$$

where $\varphi_i' \in \boldsymbol{\varphi}'$ is the camera sampled from the prior distribution and $\varphi_i \in \boldsymbol{\varphi}$ is produced by the camera generator. We refer to this loss as Camera Gradient Penalty. Note, that the first part of the loss prevents rapid changes in camera, facilitating stable optimization, while the second part avoids collapsed camera posteriors.

3.2 Adversarial depth supervision

To instill a 3D bias into our model, we develop a strategy of using depth maps predicted by an off-the-shelf estimator E (Miangoleh et al., 2021), for its advantages of being generic and readily applicable for many object categories.

The main idea is concatenating the depth map as a 4-th channel of the RGB as the input of the discriminator. The fake depth maps in this case is obtained with the help of neural rendering, while the real depth maps are estimated using monocular depth estimator E. However, naively utilizing the depth from E leads to training divergence. This happens because E could only produce relative depth, not metric depth. Moreover, monocular depth estimators are still not perfect, they produce noisy artifacts, ignore high-frequency details, and make prediction mistakes. Thus, we devise a mechanism that allows utilization of the imperfect depth maps. The central part of this mechanism is a learnable depth adaptor A, that should transform and augment the depth map obtained with neural rendering to look like a depth map from E.

More specifically, we first render raw depths $\boldsymbol{d}$ from NeRF via volumetric rendering:

$$\boldsymbol{d} = \int_{t_n}^{t_f} T(t)\sigma(r(t))t dt, \tag{2}$$

where $t_n, t_f \in \mathbb{R}$ are near/far planes, $T(t)$ is accumulated transmittance, and $r(t)$ is a ray. Raw depth is shifted and scaled from the range of $[t_n, t_f]$ into $[-1, 1]$ to obtain normalized depth $\bar{\boldsymbol{d}}$:

$$\bar{\boldsymbol{d}} = 2 \cdot \frac{\boldsymbol{d} - (t_n + t_f + b)/2}{t_f - t_n - b}, \tag{3}$$

where $b \in [0, (t_n + t_f)/2]$ is an additional learnable shift needed to account for the empty space in the front of the camera. Note that depth values obtained using E span over the entire valid range of $[0, 2^{16}]$, thus empty space does not appear in them. Therefore, we map them into the $[-1, 1]$ range to get depth $\boldsymbol{d}_r$ for real input.

Learnable Depth Adaptor. Although $\bar{\boldsymbol{d}}$ is distributed in the same range as $\boldsymbol{d}_r$, $\bar{\boldsymbol{d}}$ is still not suitable for discriminator D. The reason is that $\bar{\boldsymbol{d}}$ forces G to learn the prediction artifacts from E and creates an additional confusion for G, given E provides only relative depth. Therefore, objects in the different images with the same metric depth may still have different relative depth. To overcome such issue, we introduce the depth adaptor A to produce an adapted depth map $\boldsymbol{d}_a = \mathsf{A}(\bar{\boldsymbol{d}}) \in \mathbb{R}^{h \times w}$,

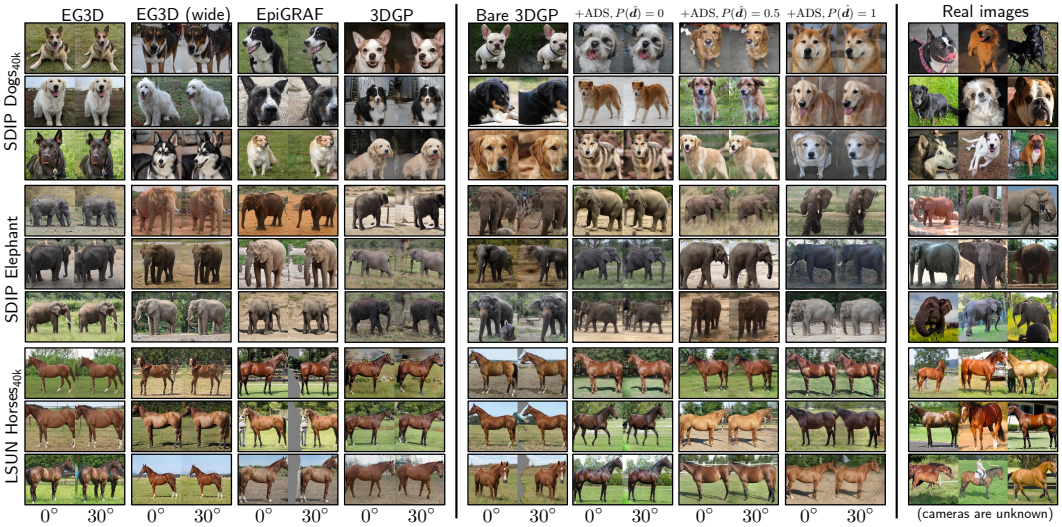


Figure 5: **Qualitative multi-view comparisons.** Left: samples from the models trained on single-category datasets with articulated geometry. Two views are shown for each sample. Middle: ablations following Tab. 1, where we change the probability of using the normalized rendered depth $P(\bar{\boldsymbol{d}})$. EG3D, EpiGRAF, and $P(\bar{\boldsymbol{d}}) = 0$ do not render realistic side views, due to the underlying flat geometry. Our full model instead generates realistic high-quality views on all the datasets. Right: randomly sampled real images. Zoom-in for greater detail.

where $h \times w$ is a number of sampled pixels. The depth (fake $\boldsymbol{d}_a$ or real $\boldsymbol{d}_r$) is concatenated with the RGB input and passed to D.

The depth adaptor A models artifacts produced by E, so that the discriminator should focus only on the relevant high level geometry. However, if A is too powerful, it could fake the depth completely, and the generator G will not receive any meaningful signal. To this end, we utilize a depth adaptor that is based on 3-layer convolutional network (shown in Fig 4). After each layer, we produce a separated depth map with different levels of adaptation: $\boldsymbol{d}_a^{(1)}$, $\boldsymbol{d}_a^{(2)}$and $\boldsymbol{d}_a^{(3)}$. The adapted depth $\boldsymbol{d}_a$ is randomly selected from the set of $\left\{\bar{\boldsymbol{d}}, \boldsymbol{d}_a^{(1)}, \boldsymbol{d}_a^{(2)}, \boldsymbol{d}_a^{(3)}\right\}$.

Such design can effectively learn good geometry while alleviating overfitting. For example, when the generator provides $\bar{\boldsymbol{d}}$ to the discriminator, it receives a strong signal for learning the geometry. On the other hand, if the discriminator sees highly adapted depth $\boldsymbol{d}_a^{(3)}$, it unlikely overfits to different unrelated estimation artifacts. Finally, $\boldsymbol{d}_a^{(1)}$and $\boldsymbol{d}_a^{(2)}$ provide some trade-off between geometry learning and precise adaptation.

3.3 Knowledge distillation for Discriminator

Knowledge from pretrained classification networks was shown to improve training stability and generation quality in 2D GANs (Sauer et al., 2021; Kumari et al., 2022; Sauer et al., 2022; Casanova et al., 2021). A popular solution proposed by Sauer et al. (2021; 2022) is to use an off-the-shelf model as a discriminator while freezing most of its weights. Unfortunately, this technique is not applicable in our scenario since we modify the architecture of the discriminator by adding an additional depth input (see §3.2) and condition on the parameters of the patch similarly to EpiGRAF (Skorokhodov et al., 2022). Thus, we devise an alternative technique that can work with arbitrary architectures of the discriminator. Specifically, for each real sample, we obtain two feature representations: e from the pretrained ResNet (He et al., 2016) network and $\hat{e}$ extracted from the final representation of our discriminator D. Our loss simply pushes $\hat{e}$ to e as follows:

$$\mathcal{L}_{\text{dist}} = \|\boldsymbol{e} - \hat{\boldsymbol{e}}\|_2^2 \,. \tag{4}$$

$\mathcal{L}_{\text{dist}}$ can effectively distill knowledge from the pretrained ResNet (He et al., 2016) into our D.

3.4 TRAINING

Our overall loss for generator G consists of two parts: adversarial loss and Camera Gradient Penalty for each camera parameter:

$$\mathcal{L}_{\mathsf{G}} = \mathcal{L}_{\text{adv}} + \sum_{\varphi_i \in \boldsymbol{\varphi}} \lambda_{\varphi_i} \mathcal{L}_{\varphi_i}, \tag{5}$$

where $\mathcal{L}_{\text{adv}}$ is the non-saturating loss (Goodfellow et al., 2014). We observe that a diverse distribution for camera origin is most important for leaning meaningful geometry, but it is also most prone to degrade to a constant solution. Therefore, we set $\lambda_{\varphi_i} = 0.3$ for $\boldsymbol{\varphi}_{\text{pos}}$, while set $\lambda_{\varphi_i} = 0.03$ for φ_{fov} and $\lambda_{\varphi_i} = 0.003$ for $\boldsymbol{\varphi}_{\text{lookat}}$.

The loss for discriminator D, on the other hand, consists of three parts: adversarial loss, knowledge distillation, and $\mathcal{R}_1$ gradient penalty (Mescheder et al., 2018):

$$\mathcal{L}_{\mathsf{D}} = \mathcal{L}_{\text{adv}} + \lambda_{\text{dist}} \mathcal{L}_{\text{dist}} + \lambda_r \mathcal{R}_1. \tag{6}$$

We use the same optimizer and hyper-parameters as EpiGRAF. We observe that for depth adaptor, sampling adapted depth maps with equal probability is not always beneficial found that using $P(\bar{\boldsymbol{d}}) = 0.5$ leads to better geometry. For additional details, see Appx B.

4 EXPERIMENTAL RESULTS

Datasets. In our experiments, we use 4 *non-aligned* datasets: SDIP Dogs (Mokady et al., 2022), SDIP Elephants (Mokady et al., 2022), LSUN Horses (Yu et al., 2015), and ImageNet (Deng et al., 2009). The first three are single-category datasets that contain objects with complex articulated geometry, making them challenging for standard 3D generators. We found it useful to remove outlier images from SDIP Dogs and LSUN Horseswith the instance selection technique (DeVries et al., 2020), reducing their size to 40K samples each. We refer to the filtered versions of these datasets as SDIP Dogs$_{40\text{k}}$, and LSUN Horses$_{40\text{k}}$, respectively. We then validate our method on ImageNet, a real-world, multi-category dataset containing $1,000$ diverse object classes, with more than $1,000$ images per category. To train all our 3D generators on ImageNet (including the baselines), we use the same filtering strategy for it, keeping 1/3 of its images (note that we always measure the metrics for it on the full ImageNet).

Evaluation. We rely on FID (Heusel et al., 2017) to measure the image quality and FID$_{2\text{k}}$, which is computed on 2,048 images instead of 50k (as for FID) for efficiency. For ImageNet, we additionally compute Inception Score (IS) (Salimans et al., 2016). Note that while we train on the filtered ImageNet, we always compute the metrics on the full one. There is no established protocol to evaluate the geometry quality of 3D generators in general, but the state-of-the-art ones are tri-plane (Chan et al., 2022; Sun et al., 2022) or MPI-based Zhao et al. (2022), and we observed that it is possible to quantify their most popular geometry failure case: flatness of the shapes. For this, we propose Non-Flatness Score (NFS) which is computed as the average entropy of the normalized depth maps histograms. We depict its intuition in Fig 6 and provide the details in Appx F.

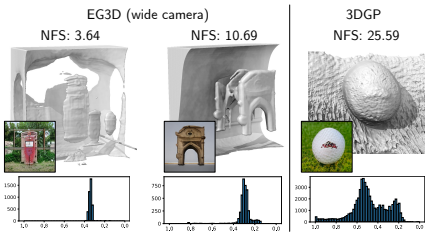


Figure 6: NFS for repetitive and diverse geometry with depth histograms.

4.1 3D GENERATION FOR SINGLE CATEGORY DATASETS

We show quantitative results for single category datasets in Tab. 1a, where we compare with EG3D (Chan et al., 2022) and EpiGRAF (Skorokhodov et al., 2022). EG3D was mainly designed for FFHQ (Karras et al., 2019) and uses the true camera poses inferred from real images as the camera distribution for the generator. But in our scenario, we do not have any knowledge about the true camera distribution, that is why to stay as close as possible to the setup EG3D was designed for, we use normal distribution for rotation and elevation angles with the same standard deviations as in FFHQ, which are equal to $\sigma_{\text{yaw}} = 0.3$ and $\sigma_{\text{pitch}} = 0.155$, respectively. Also, to make it learn better

Table 1: Comparisons on SDIP Dogs$_{40k}$, SDIP Elephants, and LSUN Horses$_{40k}$. (a) includes EG3D (with the standard and the wider camera range), EpiGRAF, and 3DGP. For completeness, we also provide 2D baseline StyleGAN2 (with KD). (b) includes the ablations of the proposed contributions. We report the total training cost for prior works, our model, and our proposed contributions.

(a) Comparison of EG3D, EpiGRAF and 3DGP.

Model	SDIP Dogs$_{40k}$		SDIP Elephants$_{40k}$		LSUN Horses$_{40k}$		Training cost
	FID$_{2k}$ ↓	NFS↑	FID$_{2k}$ ↓	NFS↑	FID$_{2k}$ ↓	NFS↑	(A100 days)
EG3D	16.2	11.91	4.78	2.59	3.12	13.34	3.7
+ wide camera	21.1	24.44	5.76	17.88	19.44	25.34	3.7
EpiGRAF	25.6	3.53	8.24	12.9	6.45	9.73	2.3
3DGP (ours)	8.74	34.35	5.79	32.8	4.86	30.4	2.6
StyleGAN2 (with KD)	6.24	N/A	3.94	N/A	2.57	N/A	1.5

(b) Impact of Adversarial Depth Supervision (ADS).

Bare 3DGP (w/o ADS, w/o C)	8.59	1.42	7.46	9.52	3.29	8.04	2.3
+ ADS, $P(\bar{\boldsymbol{d}}) = 0.0$	8.13	3.14	5.69	1.97	3.41	1.24	2.6
+ ADS, $P(\bar{\boldsymbol{d}}) = 0.25$	9.57	33.21	6.26	33.5	4.33	32.68	2.6
+ ADS, $P(\bar{\boldsymbol{d}}) = 0.5$	9.25	36.9	7.60	30.7	5.27	32.24	2.6
+ ADS, $P(\bar{\boldsymbol{d}}) = 1.0$	12.2	27.2	12.1	26.0	8.24	27.7	2.5

geometry, we trained it with a twice as wide camera distribution: $\sigma_{\text{yaw}} = 0.6$ and $\sigma_{\text{pitch}} = 0.3$. While it indeed help to improve the flatness of the shapes, it also considerably worsened the image quality, from 20% to to 500% as measured by FID$_{2k}$. For EpiGRAF, we could train it only with the default camera distribution of FFHQ and observed that it diverges with a wider one. Our model shows substantially better (at least $2\times$ than EG3D, slightly worse than StyleGAN2) FID$_{2k}$ and greater NFS on all the datasets. Low NFS indicates flat or repetitive geometry impairing the ability of the model to generate realistic side views. Indeed, as shown in Fig. 5 (left), both EG3D (with the default camera distribution) and EpiGRAF struggle to generate side views, while our method (3DGP) renders realistic side views on all three datasets.

Adversarial Depth Supervision. We evaluate the proposed Adversarial Depth Supervision (ADS) and the depth adaptor A. The only hyperparameter it has is the probability of using the non-adapted depth $P(\bar{\boldsymbol{d}})$ (see §3.2). We ablate this parameter in Tab. 1b. While FID scores are only slightly affected by varying $P(\bar{\boldsymbol{d}})$, we see substantial difference in the Non-Flatness Score. We first verify that NFS is the worst without ADS, indicating the lack of a 3D bias. When $P(\bar{\boldsymbol{d}}) = 0$, the discriminator is never presented with the rendered depth $\bar{\boldsymbol{d}}$, while the adaptor learns to fake the depth, leading to flat geometry. When $P(\bar{\boldsymbol{d}}) = 1$, the adaptor is never used, allowing the discriminator to easily determine the generated depth from the estimated depth, as there is a large domain gap (see Fig. 4), leading to reduced FID scores. We observe that the best results are obtained when $P(\bar{\boldsymbol{d}}) = 0.5$. The same observations can be visually verified in Fig. 5 (middle), where the Bare 3DGP, $P(\bar{\boldsymbol{d}}) = 0$ and $P(\bar{\boldsymbol{d}}) = 1$ are unable to render side views, while the model trained with $P(\bar{\boldsymbol{d}}) = 0.5$ features the overall best geometry and side views.

Knowledge Distillation. Here we further provide insights for discriminator D using our knowledge distillation strategy (see §3.3). We find that knowledge distillation provides an additional stability for adversarial training, along with the significant improvement in FID, which can be observed by comparing results of EpiGRAF in Tab. 1a and Bare 3DGP Tab. 1a. Additionally, we compare different knowledge distillation strategies in Appx C. However, as noted by Parmar et al. (2021), strategies which utilize additional classification networks, may provide a significant boost to FID, without corresponding improvement in visual quality.

"Ball-in-Sphere" Camera Distribution. In Fig. 7, we analyze different strategies for learning the camera distribution: sampling the camera $\boldsymbol{\varphi}$ from the prior $\boldsymbol{\varphi}'$ without learning, predicting residuals $\boldsymbol{\varphi} - \boldsymbol{\varphi}'$, and using our proposed camera generator C with Camera Gradient Penalty. For the first two cases, the learned distributions are highly concentrated around constant values. Not surprisingly, no meaningful side views can be generated as geometry becomes flat. In contrast, our Camera Gradient Penalty provides sufficient regularization to C, enabling it to converge to a sufficiently wide posterior, resulting into valid geometry and realistic side views.

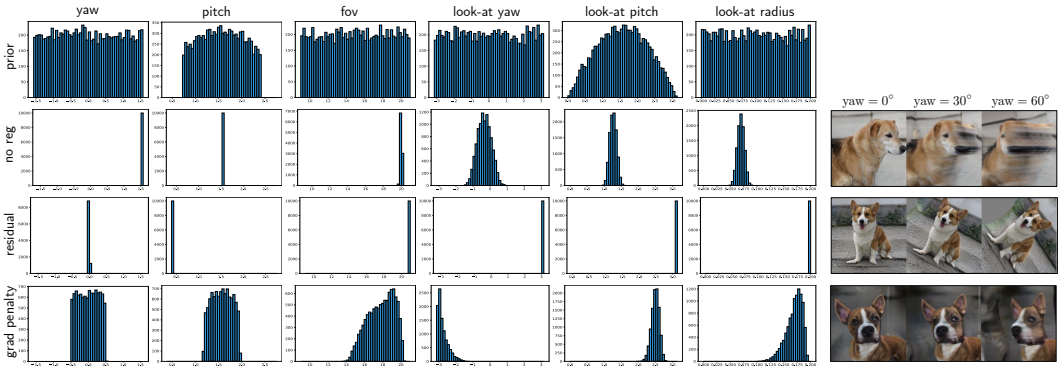


Figure 7: **Comparisons of regularization strategies.** Left: selected generic and wide prior for each of the 6 DoFs of our camera model (top row). Without any regularization (*no reg*) or with the residual-based model (*residual*) as in Niemeyer & Geiger (2021a), the camera generator collapses to highly concentrated distributions. In contrast, the proposed regularization (*grad penalty*) leads to a wider posterior. Right: random samples for each strategy. For (*no reg*) and (*residual*), no meaningful geometry is learned; only our method (*grad penalty*) leads to good geometry.

Table 2: Comparison between different generators on ImageNet 256^2 (note that 3DGP relies on the additional information in the form of depth supervision).

Method	Synthesis type	FID ↓	IS ↑	NFS↑	Training cost (A100 days) ↓
BigGAN	2D	8.7	142.3	N/A	60
StyleGAN-XL	2D	2.30	265.1	N/A	163+
ADM	2D	4.59	186.7	N/A	458
EG3D	3D-aware	26.7	61.4	3.70	18.7
+ wide camera	3D-aware	25.6	57.3	9.83	18.7
EpiGRAF	3D	47.56	26.68	3.93	15.9
3DGP (ours)	3D	26.47	73.09	27.52	28.2

4.2 3D SYNTHESIS ON IMAGENET

ImageNet (Deng et al., 2009) is significantly more difficult than single category datasets. Following prior work, we use class labels c as an additional guidance for the generator (Sauer et al., 2022; Brock et al., 2018). We modify the baselines accordingly. The quantitative results of ImageNet experiments are presented in Tab. 2. For the reference, we also report results of 2D generators, such as GAN-based methods of BigGAN (Brock et al., 2018) and StyleGAN-XL (Sauer et al., 2022) and a diffusion-based approach of ADM (Dhariwal & Nichol, 2021). As expected, 2D generators show superior FID and IS, compared to the 3D counterparts as the latter use their capacity to learn geometric details. As discussed before, the "wideness" of the camera distribution is an important factor to consider, hence, for fair comparisons, we report the standard camera ranges as well as wide ones. Despite our best efforts to find a reasonable camera distribution, both EG3D and EpiGRAF produce flat or repetitive geometry, confirming similar experiments on single category articulated datasets. 3DGP produces geometry with rich details (see Fig. 1).

5 CONCLUSION

In this work, we present the first 3D synthesis framework for in-the-wild, multi-category datasets, such as ImageNet. We demonstrate how to utilize generic priors in the form of (imprecise) monocular depth and latent feature representation to improve the visual quality and guide the geometry. Moreover, we propose the new "Ball-in-Sphere" camera model with a novel regularization scheme that enables learning meaningful camera distribution. On the other hand, our work still has several limitations, such as sticking background, lower visual quality compared to 2D generators, and no reliable quantitative measure of generated geometry. Additional discussion is provided in Appx A. This project consumed ≈12 NVidia A100 GPU years in total.

6 REPRODUCIBILITY STATEMENT

Most importantly, we will release 1) the source code and the checkpoints of our generator as a separate github repo; and 2) fully pre-processed datasets used in our work (with the corresponding extracted depth maps). In §3 and §4 and Appx B, and also in the figures throughout the text, we provided a complete list of architecture and optimization details needed to reproduce our results. In §5, we also We are also open to provide any further details on our work in public and/or private correspondence.

7 ETHICS STATEMENT

There are two main ethical concerns around the deep learning projects on synthesis: generation of fake content to mislead people (e.g. fake news or deepfakes[1]) and potential licensing, authorship and distribution abuse, which recently got controversy in the research community in the context of 2D image generation (by Stable Diffusion Rombach et al. (2022) and DALL-E Ramesh et al. (2022)) and code generation (by Github Copilot[2]). While the subpar synthesis quality of our generator makes it safe from such concerns, we still warn the community to build on top of the developed ideas with caution, since they could strengthen a potentially harmful system.

[1] https://en.wikipedia.org/wiki/Deepfake.
[2] https://github.com/features/copilot.

REFERENCES

Sherwin Bahmani, Jeong Joon Park, Despoina Paschalidou, Hao Tang, Gordon Wetzstein, Leonidas Guibas, Luc Van Gool, and Radu Timofte. 3d-aware video generation. *arXiv preprint arXiv:2206.14797*, 2022.

Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Arantxa Casanova, Marlène Careil, Jakob Verbeek, Michal Drozdzal, and Adriana Romero-Soriano. Instance-conditioned gan. In *Proceedings of the Neural Information Processing Systems Conference*, 2021.

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021.

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 2009.

Yu Deng, Jiaolong Yang, Jianfeng Xiang, and Xin Tong. Gram: Generative radiance manifolds for 3d-aware image generation. In *IEEE Computer Vision and Pattern Recognition*, 2022.

Terrance DeVries, Michal Drozdzal, and Graham W Taylor. Instance selection for gans. *Advances in Neural Information Processing Systems*, 2020.

Terrance DeVries, Miguel Angel Bautista, Nitish Srivastava, Graham W. Taylor, and Joshua M. Susskind. Unconstrained scene generation with locally conditioned radiance fields. *arXiv*, 2021.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Ainaz Eftekhar, Alexander Sax, Jitendra Malik, and Amir Zamir. Omnidata: A scalable pipeline for making multi-task mid-level vision datasets from 3d scans. In *ICCV*, 2021.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the Neural Information Processing Systems Conference*, 2014.

Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d aware generator for high-resolution image synthesis. In *International Conference on Machine Learning*, 2022.

Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

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*, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *Proceedings of the Neural Information Processing Systems Conference*, 2017.

Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *arXiv preprint arXiv:2006.06676*, 2020a.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020b.

Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *arXiv preprint arXiv:2106.12423*, 2021.

Mijeong Kim, Seonguk Seo, and Bohyung Han. Infonerf: Ray entropy minimization for few-shot neural volume rendering. In *CVPR*, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Zhengfei Kuang, Kyle Olszewski, Menglei Chai, Zeng Huang, Panos Achlioptas, and Sergey Tulyakov. Neroic: Neural rendering of objects from online image collections. In *Special Interest Group on Computer Graphics and Interactive Techniques*, 2022.

Nupur Kumari, Richard Zhang, Eli Shechtman, and Jun-Yan Zhu. Ensembling off-the-shelf models for gan training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021.

Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021.

Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, 2018.

S Mahdi H Miangoleh, Sebastian Dille, Long Mai, Sylvain Paris, and Yagiz Aksoy. Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9685–9694, 2021.

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 *European conference on computer vision*, 2020.

Sangwoo Mo, Minsu Cho, and Jinwoo Shin. Freeze the discriminator: a simple baseline for fine-tuning gans. *arXiv preprint arXiv:2002.10964*, 2020.

Ron Mokady, Michal Yarom, Omer Tov, Oran Lang, Daniel Cohen-Or, Tali Dekel, Michal Irani, and Inbar Mosseri. Self-distilled stylegan: Towards generation from internet photos. *arXiv preprint arXiv:2202.12211*, 2022.

Michael Niemeyer and Andreas Geiger. Campari: Camera-aware decomposed generative neural radiance fields. In *2021 International Conference on 3D Vision (3DV)*, 2021a.

Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021b.

Augustus Odena, Jacob Buckman, Catherine Olsson, Tom Brown, Christopher Olah, Colin Raffel, and Ian Goodfellow. Is generator conditioning causally related to gan performance? In *International conference on machine learning*, 2018.

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. *arXiv preprint arXiv:2112.11427*, 2021.

Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu. On buggy resizing libraries and surprising subtleties in fid calculation. *arXiv preprint arXiv:2104.11222*, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Proceedings of the Neural Information Processing Systems Conference*, 2016.

Axel Sauer, Kashyap Chitta, Jens Müller, and Andreas Geiger. Projected gans converge faster. *Advances in Neural Information Processing Systems*, 34:17480–17492, 2021.

Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 Conference Proceedings*, 2022.

Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proceedings of the European Conference on Computer Vision*, 2016.

Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Proceedings of the Neural Information Processing Systems Conference*, 2020.

Katja Schwarz, Axel Sauer, Michael Niemeyer, Yiyi Liao, and Andreas Geiger. Voxgraf: Fast 3d-aware image synthesis with sparse voxel grids. *ARXIV*, 2022.

Zifan Shi, Yujun Shen, Jiapeng Zhu, Dit-Yan Yeung, and Qifeng Chen. 3d-aware indoor scene synthesis with depth priors. 2022.

Ivan Skorokhodov, Sergey Tulyakov, Yiqun Wang, and Peter Wonka. Epigraf: Rethinking training of 3d gans. In *Proceedings of the Neural Information Processing Systems Conference*, 2022.

Jingxiang Sun, Xuan Wang, Yichun Shi, Lizhen Wang, Jue Wang, and Yebin Liu. Ide-3d: Interactive disentangled editing for high-resolution 3d-aware portrait synthesis. *arXiv preprint arXiv:2205.15517*, 2022.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, 2019.

Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF$--$: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021.

Yinghao Xu, Sida Peng, Ceyuan Yang, Yujun Shen, and Bolei Zhou. 3d-aware image synthesis via learning structural and textural representations. *arXiv preprint arXiv:2112.10759*, 2021.

Yang Xue, Yuheng Li, Krishna Kumar Singh, and Yong Jae Lee. Giraffe hd: A high-resolution 3d-aware generative model. *arXiv preprint arXiv:2203.14954*, 2022.

Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction. *arXiv preprint arXiv:2206.00665*, 2022.

Xuanmeng Zhang, Zhedong Zheng, Daiheng Gao, Bang Zhang, Pan Pan, and Yi Yang. Multi-view consistent generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

Xiaoming Zhao, Fangchang Ma, David Güera, Zhile Ren, Alexander G. Schwing, and Alex Colburn. Generative multiplane images: Making a 2d gan 3d-aware. In *Proc. ECCV*, 2022.

Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. Cips-3d: A 3d-aware generator of gans based on conditionally-independent pixel synthesis. *arXiv preprint arXiv:2110.09788*, 2021.

A LIMITATIONS

Lower visual quality compared to 2D generators. Despite providing a more reasonable representation of the underlining scene, 3D generators still have a lower visual quality compared to 2D generators. Closing this gap is essential for a wide adaptation of 3D generators.

Background sticking. One common 3D artifact of 3DGP is gluing of the foreground and the background. In other words our model predicts a single shape for both and there is no clear separation between the two. One potential cause of this artifact is the dataset bias, where most of the photos are frontal, thus it is not beneficial for the model to explore backward views. Another reason is the bias of tri-planes toward flat geometry. However, all our attempts to replace tri-planes with an MLP-based NeRF led to much worse results (see Appx D). Inventing a different efficient parametrization may be an important direction.

No reliable quantitative measure of generated geometry. In this work we introduce Non-Flatness Score as a proxy metric for evaluating the quality of the underlying geometry. However it can capture only a single failure case, specific for generators based on tri-planes: flatness of geometry. Devising a reasonable metric applicable to a variety of scenarios could significantly speed up the progress in this area.

Camera model C does not learn fine-grained control over the classes distribution. While our camera model is conditioned on the class label $\boldsymbol{c}$, and, in theory, it should be able to perform fine-grained control over the class focal length distributions (which is natural since landscape panoramas and close-up view of a coffee mug typically have different focal lengths), it does not do this, as shown in Fig. 8. We attribute this problem to the implicit bias of G to produce large-FoV images due to tri-planes parametrization. Tri-planes define a limited volume box in space, and close-up renderings with large focal length would utilize fewer tri-plane features, hence using less generator's capacity. This is why 3DGP attempts to perform modeling with larger field-of-view values.

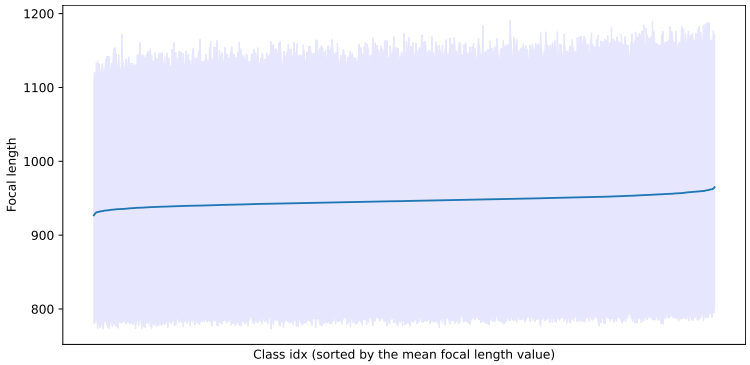


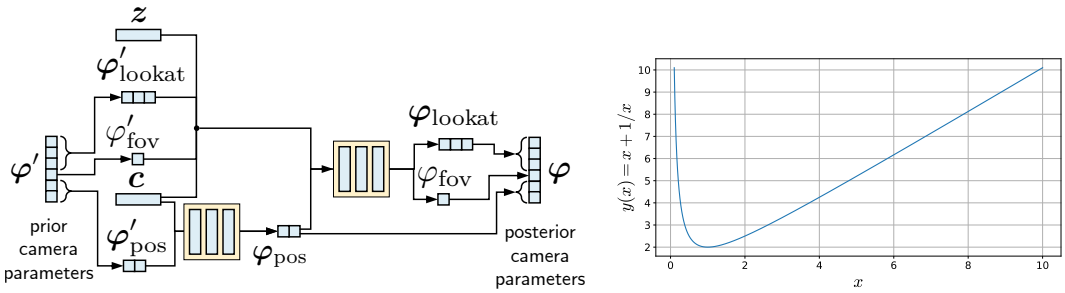
Figure 8: Focal length distribution on ImageNet 256 learned by C. The blue solid line is the mean values, while lower/upper curves are 0.05 and 0.95 percentiles, respectively.

B IMPLEMENTATION DETAILS

This section provides additional architectural and training details. Also note that we will release the source code. Our generator G consists of M, S and T. Mapping network M takes noise $\boldsymbol{z} \in \mathbb{R}^{512}$ and class label $c \in 0, ..., K-1$, where K is the number of classes, and produces the style code $\boldsymbol{w} \in \mathbb{R}^{512}$. Similar to StyleGAN2-ADA Karras et al. (2020a) and EpiGRAF Skorokhodov et al. (2022), M is 2-layer MLP network with LeakyReLU activations and 512 neurons in each layer. Synthesis network is a decoder network same as StyleGAN2 Karras et al. (2020b), except that it produce tri-plane feature features $\boldsymbol{p} = (\boldsymbol{p}^{xy}, \boldsymbol{p}^{yz}, \boldsymbol{p}^{xz}) \in \mathbb{R}^{3\times(512\times512\times32)}$. A

Table 3: Names of the networks used in our method:

G	Generator
M	Mapping network
S	Synthesis network
T	Tri-plane decoder
C	Camera generator
A	Depth adaptor
D	Discriminator
E	Depth Estimator
F	Feature Extractor



(a) **Camera generator** C. We condition C on class labels $\boldsymbol{c}$ when generating the camera position $\hat{\boldsymbol{\varphi}}$ since it might be different for different classes. And we condition it on $\boldsymbol{z}$ when generating the look-at position and field-of-view since it might depend on the object shape (e.g., there is a higher probability to synthesize a close-up view of a dog's snout rather than its tail). Each MLP consists on 3 layers with Softplus non-linearities.

(b) **Camera gradient penalty**. We structure the regularization term for C as a function $\mathcal{L}(|g|) = |g| + 1/|g|$ (see (1)), and this function is visualized above. This allows to prevent the collapse of C into delta distribution by it either producing constant values or very large/small values (which become constant after the sigmoid normalization).

Figure 9: Camera generator architecture and visualizing its corresponding regularization term.

feature vector $\boldsymbol{f}_{xyz} \in \mathbb{R}^{32}$ located $(x, y, z) \in \mathbb{R}^3$ is computed by projecting the coordinate back to the tri-plane representation, followed by bi-linearly interpolating the nearby features and averaging the features from different planes. Following EpiGRAF (Skorokhodov et al., 2022), tri-plane decoder is a two-layer MLP with Leaky-ReLU activations and 64 neurons in the hidden layer, that takes a tri-plane feature $\boldsymbol{f}_{xyz}$ as input and produced the color and density (RGB, σ) in that point. We use same volume rendering procedure as EpiGRAF (Skorokhodov et al., 2022).

Camera generator. Camera generator consists of linear layers with SoftPlus activations, it architecture is depicted in Fig. 9. We found it crucial to use SoftPlus activation and not LeakyReLU, since optimization of Camera Gradient Penalty for non-smooth functions is unstable for small learning rates (smaller 0.02). Our gradient penalty minimizes the function $\mathcal{L} = |g| + 1/|g|$, where $|g|$ is the input/output scalar derivative of the camera generator C. The motivation is to prevent the collapse of C into delta distribution and the intuition is the following. C can collapse into delta distribution in two ways: 1) by starting to produce the constant output for all the input values (this is being prevented by the first term $|g|$ and 2) by starting producing $\pm\infty$ for all the inputs, which are at the end converted to constants since we apply sigmoid normalization on top of its outputs to normalize them into a proper range (e.g., pitch is bounded in $(0, \pi)$) — this, in turn, is prevented by the second term $1/|g|$. The minimum value of this function is 2 (which is achieved when the gradient is linear), and the function itself is visualized in Fig. 9b.

Adversarial depth supervision. For depth adaptor, we use a three layer convolutional neural network with 5×5 kernel sizes (since it is shallow, we increase its receptive field by increasing the kernel size) with LeakyReLU activations and 64 filters in each layer. Additionally we use one shared convolutional layer that converts $64 \times h \times w$ features to the depth maps. We use the same architecture for D as EpiGRAF (Skorokhodov et al., 2022), but additionally concatenate a 1-channel depth to the 3-channel RGB input. Finally E and F is pretrainined LeReS (Miangoleh et al., 2021) and ResNet50 (He et al., 2016) networks without any modifications. We used the `timm` library to extract the features for real images.

Similarly to EpiGRAF (Skorokhodov et al., 2022) we set the $\mathcal{R}_1$ regularization (Mescheder et al., 2018) term $\lambda_r = 0.1$ and knowledge distillation term $\lambda_{\text{dist}} = 1$. $\mathcal{R}_1$ regularization helps to stabilize GAN training and is formulated as a gradient penalty on top of the discriminator's real inputs:

$$\mathcal{R}_1 = \frac{1}{2}\|\nabla_{\boldsymbol{x}} \mathsf{D}(\boldsymbol{x})\|_2^2 \longrightarrow \min_{\mathsf{D}}. \tag{7}$$

The (hand-wavy) intuition is that it makes the discriminator surface more flat in the vicinity of real points, making it easier for the generator to reach them. We train all the models with Adam

optimizer (Kingma & Ba, 2014) using the learning rate of $2e$-3 and $\beta_1 = 0.0, \beta_2 = 0.99$. Following EpiGRAF (Skorokhodov et al., 2022), our model uses patch-wise training with 64×64-resolution patches and uses their proposed β scale sampling strategy without any modifications. We use the batch size of 64 in all the experiments, since in early experiments we didn't find any improvements from using a large batch size neither for our model nor for StyleGAN2, as observed by Brock et al. (2018) and Sauer et al. (2022).

As being said in Section 4, we use the instance selection technique by DeVries et al. (2020) to remove image outliers from the datasets since they might negatively affect the geometry. This procedure works by first extracting a 2,048-dimensional feature vector for each image, then fitting a multivariate gaussian distribution on the obtained dataset and removing the images, which features have low probability density. For SDIP Dogs and LSUN Horses, we fit a multi-variate gaussian distribution for the whole dataset. For ImageNet, we fit a separate model for each class with additional diagonal regularization for covariance, which is needed due to its singularity: feature vector has more dimensions than the number of images in a class. We refer to (DeVries et al., 2020) for additional details.

Also note that we will release the source code and the checkpoints, which would additionally convey all other implementation details.

C KNOWLEDGE DISTILLATION COMPARISON

In this section we compare our knowledge distillation strategy with projected D (Sauer et al., 2021; 2022), another popular technique of utilizing existing classification models for GAN training. Note that projected D relies on pretrained EfficientNet (Tan & Le, 2019), it freezes most of the weights and adds some random convolutions layers that project intermediate outputs of EfficientNet to the features, that will be later processed by discriminator. Note that this strategy relies on specific architecture of Discriminator D, thus in order to adapt it for 3DGP we have to rely on two discriminators. First one is our disciminator D with additional depth input and conditioned on patch parameters and second one is projected D. In Tab 4 we show comparison between projected D and our knowledge distillation strategy on ImageNet (Deng et al., 2009) and SDIP Dogs_{40k} Mokady et al. (2022) datasets. First we compare these strategies for 2D generator architectures: StyleGAN3-t (large) (Karras et al., 2021) and StyleGAN2 (Karras et al., 2020a). It can be observed that projected D is not generic and it results heavily varies depending on generator architecture. Moreover training cost of projective D is higher. Finally we test this strategy for our 3D generator, and again we can observe that this technique lead to inferior results in both visual quality and training cost.

Table 4: Comparing our knowledge transfer strategy with the one from StyleGAN-XL.

Method	ImageNet 128^2 @ 10M		SDIP Dogs_{40k} 256^2 @ 5M		Restricts D's architecture?
	FID↓	Training cost ↓	FID ↓	Training cost ↓	
StyleGAN3-t (large)	28.1	11.9	6.42	7.3	No
- with Projected D	22.8	11.6	22.6	8.1	Yes
- with Knowledge Distillation	16.3	11.9	2.47	7.3	No
StyleGAN2	33.7	1.81	9.79	1.3	No
- with Projected D	160.5	3.83	4.10	2.1	Yes
- with Knowledge Distillation	20.75	1.83	2.08	1.4	No
3DGP	53.6	6.33	21.3	2.6	No
- with Projected D	105.9	8.0	10.5	4.2	Yes
- with Knowledge Distillation	27.8	6.83	4.51	2.6	No

D FAILED EXPERIMENTS

In this section we describe several ideas that have been tried in this project, which however did not work for us.

Gradient Penalty for Depth Adaptor. To prevent depth adaptor A from completely faking the depth and forcing it to still provide a useful signal to the generator, one could employ a gradient

penalty that will force gradient of A close to one. We tried it in two setups: window-to-pixel gradient and pixel-to-pixel gradient. Even with the large weight for this loss and shallow adaptor with $k = 3$ kernel size, it didn't enforce good geometry

3DGP with Projected D. Before devoting to knowledge distillation, we spend significant resources trying to incorporate projected D in our setup. For double discriminators setting, see Appx C, we tested: different weighting strategies, less frequent updates for projected D, enabling projected D after some geometry was already learned. We also tested single projected D, where we learned additional depth encoder that can consume our depth inputs. In all these experiments learned geometry was significantly inferior to the setting without projected D. Most of the time geometry becomes completely flat.

MLPs instead of tri-planes. We noticed that tri-planes are extremely biased towards flat generation. Therefore one natural idea would be to utilize different representations, one possible candidate is MLPs Chan et al. (2021). However our experiments with MLPs always lead to significantly inferior visual quality. Our hypothesis is that they just don't have enough capacity to model large scale datasets.

Few-shot NeRF regularization. We also try to improve learned geometry by incorporating ray entropy minimization loss from InfoNeRF Kim et al. (2022) with different weights. But the model always diverges. We hypothesize that this is because it is significantly more complicated to find proper geometry, if the model is stuck in local minimum where the object densities are very sharp.

Normal supervision. Another prominent idea was utilizing normal supervision, since generic networks for normals Eftekhar et al. (2021) are also widely available and provide good results on the arbitrary data. Since computing normals requires gradient of density σ with respect to (x, y, z) and pytorch does not provide second order derivatives for `F.grid_sample` [3], we developed a custom CUDA kernel that computes second derivative for `F.grid_sample`. Unfortunately we found that normals supervision is inferior to depth and it is much easier for the generator G to fake normals.

Preventing C collapse via variance/entropy/moments regularization. Before arriving to our current form we had been experimenting with other forms of regularization: maximizing variance with some small loss coefficient, or entropy (in the assumption that the posterior distribution is gaussion), or additionally pushing mean/skewness/kurtosis to the one of the gaussian distribution. But each time, the generator was finding the ways to "cheat" the regularization and managing to produce either a delta distribution or a mixture of delta distributions (it very "likes" doing so since, it is able to completely flat images and cheat the geometry in this regime).

E Additional samples

Since it is much easier to visualize the samples from NeRF-based generators as videos rather than RGB images, we provide all the additional visualizations on https://u2wjb9xxz9q.github.io/.

F Non-Flatness Score details

To detect and quantify the flatness of 3D generators, we propose a Non-Flatness Score (NFS). To compute NFS, we sample N latent codes with their corresponding radiance fields. For each radiance field, we perform integration following Eq. 2 to obtain its depth, however, we first set the 50% of lowest density to zero. This is necessary to cull spurious density artifacts.

We then normalize each depth map according to the corresponding near and far planes and compute a histogram with B bins, showing the distribution of the depth values. To analyze how much the depth values are concentrated versus spread across the volume, we compute its entropy for each distribution. Averaging over N depth maps gives the sought score:

$$\text{NFS} = \frac{1}{N}\sum_{i=1}^{N} \exp\left[-\frac{1}{h \cdot w}\sum_{j=1}^{B} b(\boldsymbol{d}^{(i)})_j\right], \tag{8}$$

[3] https://github.com/pytorch/pytorch/issues/34704

where $b(\boldsymbol{d}^{(i)})_j$ is the normalized number of depth values in the j-th bin of the i-th depth map $\boldsymbol{d}^{(i)}$, with $N = 256$ and $B = 64$. NFS does not directly evaluate the quality of the geometry. Instead, it helps to detect and quantify how flat the generated geometry is. In Fig. 6, we show examples of repetitive geometry generated by EG3D and a more diverse generation produced by 3DGP along with their depth histograms and NFS.