GaussiGAN: Controllable Image Synthesis with 3D Gaussians from Unposed Silhouettes

Youssef A. Mejjati[1], Isa Milefchik[5], Aaron Gokaslan[2], Oliver Wang[3], Kwang In Kim[4], James Tompkin[5]
[1]University of Bath, [2]Cornell, [3]Adobe, [4]UNIST, [5]Brown University

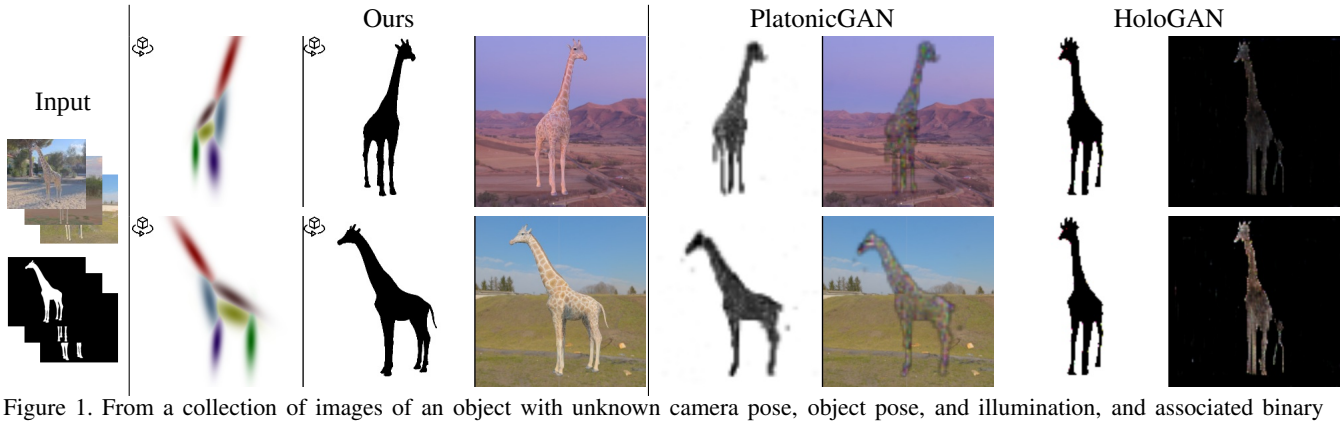

Figure 1. From a collection of images of an object with unknown camera pose, object pose, and illumination, and associated binary segmentation masks, we reconstruct an implicit 3D Gaussian representation which is then used to generate a detailed 2D mask and texture (Ours). This approach allows for rendering arbitrary camera pose, and matching lighting to a provided background image. PlatonicGAN [3] (*mid right*) can generate inconsistent voxel reconstructions, and complex texturing is a challenge. HoloGAN [13] (*far right*) struggles to represent high-quality masks, and training on texture can lead to less structured 3D spaces. *Please view in Adobe Acrobat to see animations.*

Abstract

We present an algorithm to reconstruct a coarse representation of objects from unposed multi-view 2D mask supervision. Our approach learns to represent object shape and pose with a set of self-supervised canonical 3D anisotropic Gaussians, via a perspective camera and a set of per-instance transforms. We show that this robustly estimates a 3D space for the camera and object, while recent state-of-the-art voxel-based baselines struggle to reconstruct either masks or textures in this setting. We show results on synthetic datasets with realistic lighting, and demonstrate an application of object insertion. This helps move towards structured representations that handle more real-world variation in learned object reconstruction.

1. Introduction

Recovering 3D object representations from unstructured 2D data in an unsupervised setting is an open problem. Data contain camera pose variations, object pose variations, lighting variations, background variations, and other innumerable differences to factor. Current image generation methods employ geometric proxies with differentiable projection mechanisms to learn deep occupancy or appearance vectors without camera pose supervision, often via voxels like the recent HoloGAN [13] and PlatonicGAN [3].

We use a mixture of anisotropic 3D Gaussians as a coarse implicit geometry proxy. This contains a canonical object, plus per-instance camera and per-Gaussian transformation parameters that describe camera and object pose. This representation is compact vs. voxels, and still inherits correct perspective projection properties. We assume only known 2D object segmentation masks as supervision (eliminating the problem of unknown background), and demonstrate the advantages of our representation on rendered data that contains varying camera pose, object pose, and illumination.

By using reconstruction and self-supervised transform losses, we can robustly estimate a representation that maintains a 3D space. Through training, our representation learns to associate object parts with Gaussians without part-level supervision. Additionally, we show that factoring object pose variation into a canonical representation plus deformation parameters improves representation quality. We use the learned Gaussians to condition RGB image generation, and show disentangling of pose, view-dependent texture, and shading variation caused by lighting. In comparisons to baseline methods (Fig. 1), we show that our representation can lead to more consistent 2D texture generation and higher-quality masks and RGB images. Looking forward, such an approach may be a step towards more flexible hybrid learned object representations that can model complex real-world variation from natural image collections.

2. Related work

3D Object Representations. Learned 3D object representations exist for taking 3D input data like point clouds [1], volumes [15], or meshes [4] and generating

3D output data. These include techniques to fit sets of Gaussians to 3D shapes using 3D supervision [2], and by combining 3D supervision with multi-view silhouette losses [19]. Some works use pre-defined detailed canonical 3D meshes for 2D images [21], e.g., to learn surface parameterizations [7]. Other works learn representations from 2D input data via 3D representations, but require camera information at training time [11]. For instance, DeepVoxels [16] projects RGB values on known camera rays to learn a deep voxel space that reproduces 2D inputs when projected and decoded. Other works require object-specific pose data, such as human skeletons [6]. Without camera poses, Lei et al. build surface parameterizations for 3D objects [8].

For image generation, few works take only 2D input and *no* camera or object pose information for supervision—this is a harder problem as there is no explicit constraint on the 3D space. Liao et al. use cube and sphere mesh proxies to represent multiple simple scene objects [9]. Schwarz et al. generate radiance fields from unposed 2D images for synthetic 3D objects [15]. HoloGAN uses a deep voxels with an implicit rotation space [13], and PlatonicGAN uses discrimination on random rotations to learn a generative voxel space [3]. These two works are closest to our setting, except we use an implicit 3D Gaussian representation along with a conditioned 2D generator for fine-scale detail.

3. Learning Gaussian proxies for shape & pose

Input masks and anisotropic 3D Gaussians. We start with a dataset of 256×256 binary segmentation masks $\boldsymbol{m} \in \mathcal{M}$ of an object under varying unknown camera parameters and object poses. We also require a given number K of unnormalized anisotropic 3D Gaussians $\{\mathcal{G}_k\}_{k=1}^K$ (Fig. **??**). Each Gaussian $\mathcal{G}_k$ has mean vector $\boldsymbol{\mu}_k \in \mathbb{R}^3$ and covariance matrix $\boldsymbol{\Sigma}_k \in \mathbb{R}^{3\times3}$ with its density declared as:

$$\mathcal{G}_k(\mathbf{x}) = \exp\left(-(\mathbf{x}-\boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\boldsymbol{\mu}_k)\right). \quad (1)$$

Camera. We also declare a general perspective pinhole camera with intrinsic matrix $\mathbf{K}$, rotation $\mathbf{R}$, and translation $\mathbf{t}$ such that camera matrix $\mathbf{P}$ is represented as $\mathbf{K}[\mathbf{R}, \mathbf{t}]$. To project a 3D anisotropic Gaussian into our camera's image plane to produce a 2D anisotropic Gaussian, we use the analytically-differentiable perspective projection function π of Sridhar et al. [17]. This is valid for general perspective cameras, unlike orthographic approaches [3] or Gaussian-based approaches that are only valid under paraperspective [19] projection models and so are less applicable to real-world cameras. Please see supplemental material for details of π. In our experiments, $\mathbf{K}$ is fixed across instances and approximately matches that in the data.

Canonical Gaussians. Given a 256-dimensional constant [5] as input, we use a fully connected network $E_{\mathcal{G}^c}$ to predict the canonical 3D Gaussians $\mathcal{G}_k^c$ each parameterized by a mean and covariance $(\boldsymbol{\mu}^c, \boldsymbol{\Sigma}^c)$ (Fig. 2, green, top).

Per-instance Gaussian transforms. Given an input mask $\boldsymbol{m}$, we extract a latent vector representing pose $\mathbf{z} \in \mathbb{R}^8$ via a convolutional encoder network $E_{\boldsymbol{m}}$. Then, from $\mathbf{z}$, we use a fully connected network to predict two transformations: 1) A camera transformation $\mathbf{T}_O$ that moves the camera with respect to the canonical model; in our experiments, we mainly consider a yaw rotation $\mathbf{R}_\phi$. 2) K Gaussian local transformations $\mathbf{T}_k$ consisting of scale, translation, and rotation $(\mathbf{s}_k, \mathbf{t}_k, \boldsymbol{\theta}_k)$ with each in $\mathbb{R}^3$ (Fig. 2, green, bottom).

Given the canonical parameters $(\boldsymbol{\mu}_k^c, \boldsymbol{\Sigma}_k^c)$, we obtain the per-instance Gaussians $\mathcal{G}_k$ with parameters $(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ via:

$$\begin{aligned}\boldsymbol{\mu}_k &= \mathbf{R}_\phi(\boldsymbol{\mu}_k^c + \mathbf{t}_k)\\ \boldsymbol{\Sigma}_k &= (\mathbf{R}_\phi\mathbf{R}_{\theta_k}\mathbf{U}_k\mathbf{s}_k\mathbf{S}_k)(\mathbf{R}_\phi\mathbf{R}_{\theta_k}\mathbf{U}_k\mathbf{s}_k\mathbf{S}_k)^\top, \end{aligned} \quad (2)$$

where $\mathbf{R}_{\theta_k}$ is the rotation matrix form of $\boldsymbol{\theta}_k$, and $\mathbf{S}_k$ and $\mathbf{U}_k$ are obtained via eigenvalue decomposition of $\boldsymbol{\Sigma}_k^c$: $\boldsymbol{\Sigma}_k^c = (\mathbf{S}_k\mathbf{U}_k)(\mathbf{S}_k\mathbf{U}_k)^\top$. $\mathbf{S}_k$ is a diagonal matrix. The square of its (j,j)-th entry represents the j-th eigenvalue of $\boldsymbol{\Sigma}_k$. This allows us to control the scale and rotation of each individual Gaussian via the matrices $\mathbf{U}_k$ and $\mathbf{S}_k$.

Conditional mask synthesis. Even a large number of Gaussian proxies will fail to reconstruct sharp edges and fine detail within a mask. As such, we use a conditional mask generator $G_{\boldsymbol{m}}$ to add back the detail using upsampling transposed convolutions (Fig. 2, yellow). Given the 3D Gaussians for an instance, we project them to 2D Gaussians on the image plane of our camera: $\pi(\mathcal{G}_k) = (\boldsymbol{\mu}_k^\pi, \boldsymbol{\Sigma}_k^\pi)$. Then, using the 2D version of Eq. 1, we sample the density of each projected Gaussian on a raster grid to create K Gaussian maps $\{\boldsymbol{g}_k\}_{k=1}^K$. These are input to $G_{\boldsymbol{m}}$ to condition the synthesis of predicted mask $\boldsymbol{m}'$, which is the learned reconstruction of $\boldsymbol{m}$. We enforce a stronger effect in $G_{\boldsymbol{m}}$ by using layer-wise conditioning via Gaussian maps at $32^2, 64^2, 128^2,$ and 256^2 resolutions.

3.1. Losses

We encourage our network to reconstruct an object using multiple losses, with overall energy to minimize given by:

$$\begin{aligned}\mathcal{L}(E_{\mathcal{G}^c}, E_{\boldsymbol{m}}, G_{\boldsymbol{m}}, D_{\boldsymbol{m}}) = \lambda_1\mathcal{L}_{\text{Rec}} + \lambda_2\mathcal{L}_{\boldsymbol{g}} +\\ \lambda_3\mathcal{L}_{\widehat{\mathcal{G}}} + \lambda_4\mathcal{L}_{\widehat{\boldsymbol{g}}} + \lambda_5\mathcal{L}_{\text{Adv}} + \lambda_6\mathcal{L}_{\text{FM}}\end{aligned} \quad (3)$$

Reconstruction loss. We encourage synthesized mask $\boldsymbol{m}'$ to reconstruct input instance mask $\boldsymbol{m}$ with an L_1 loss: $\mathcal{L}_{\text{Rec}}(\boldsymbol{m}, \boldsymbol{m}') = \|\boldsymbol{m} - \boldsymbol{m}'\|_1$.

Density loss. Even though they cannot represent fine detail in $\boldsymbol{m}$, we still wish for all projected Gaussians to 1) cover regions of the mask without overlap, and 2) cover as much of the mask as possible. We encourage this via:

$$\mathcal{L}_{\boldsymbol{g}}(\boldsymbol{m}, \boldsymbol{g}) = \left\|\boldsymbol{m} - \Sigma_{k=1}^K \boldsymbol{g}_k\right\|_1. \quad (4)$$

The sum over sampled 2D Gaussians is equivalent to a grayscale version of the colored parts visualization in Figure 3. Here, both inputs are in the range $[0, 1]$, and we take $\boldsymbol{g}$ at our mask resolution of 256×256.

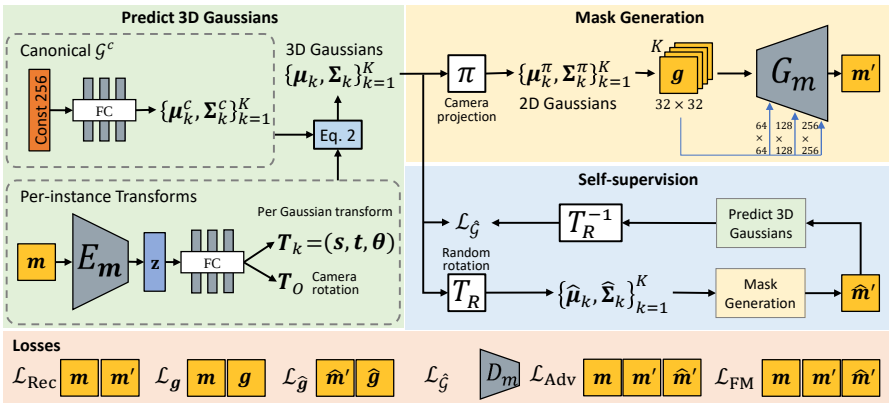

Figure 2. Learning a K-part anisotropic 3D Gaussian representation from masks $\boldsymbol{m}$. *Green:* We combine a canonical representation with scale, rotation, and translation transforms. *Yellow:* We project these to K 2D maps. $\boldsymbol{g}$ conditions network $G_{\boldsymbol{m}}$ to generate a detailed mask $\boldsymbol{m}'$ as a reconstruction of $\boldsymbol{m}$. *Blue:* To learn a meaningful 3D space, we self supervise reconstruction by forcing a random rotation of our estimated 3D Gaussians to also produce a plausible mask $\widehat{\boldsymbol{m}}'$ *and* for its 3D Gaussian prediction to be consistent after the inverse rotation.

Self-supervised transform mask loss. We wish for the 3D space expressed through our recovered object Gaussians and camera transform parameters in $\mathbf{T}_O$ to be consistent across varying camera views even though we only have mask supervision. Thus, we randomly sample a 3D transformation $\mathbf{T}_R$, again mainly as a yaw rotation, and apply it via Eq. 2 to produce rotated 3D Gaussians $\widehat{\mathcal{G}} = (\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})$. As before, these are then projected via π to 2D parameters $(\widehat{\boldsymbol{\mu}}^\pi, \widehat{\boldsymbol{\Sigma}}^\pi)$, then sampled into 2D maps $\widehat{\boldsymbol{g}}$, and finally via $G_{\boldsymbol{m}}$ to generate a mask $\widehat{\boldsymbol{m}}'$ (Fig. 2, blue).

As $\widehat{\boldsymbol{m}}'$ does not correspond to a known input instance, we cannot directly enforce $\mathcal{L}_{\text{Rec}}$. Instead, we encourages the projected novel view Gaussians $\widehat{\boldsymbol{g}}$ to be consistent with the synthesized novel view $\widehat{\boldsymbol{m}}'$ via a second density loss: $\mathcal{L}_{\widehat{\boldsymbol{g}}}(\widehat{\boldsymbol{m}}', \widehat{\boldsymbol{g}}) = \|\widehat{\boldsymbol{m}}' - \sum_{k=1}^K \widehat{\boldsymbol{g}}_k\|_1$. Without this loss, $\boldsymbol{g}$ can describe well the input mask $\boldsymbol{m}$, but the rotated $\widehat{\boldsymbol{g}}$ may not describe well the generated mask $\widehat{\boldsymbol{m}}'$.

Self-supervised transform inverse 3D Gaussian loss. We can also pass $\boldsymbol{m}'$ back through our 3D Gaussian prediction stages (Fig. 2, green) to recover an estimate of the proxies under random transform $\mathbf{T}_R$. Then, we can invert this transform and penalize a loss against our initial estimate of the 3D Gaussians. With slight notation abuse: $\mathcal{L}_{\widehat{\mathcal{G}}}(\mathcal{G}, \widehat{\mathcal{G}}') = \|\mathcal{G} - \mathbf{T}_R^{-1}(\eta(\widehat{\boldsymbol{m}}'))\|_1$, where η predicts 3D Gaussians for a mask.

Adversarial loss. Training using only reconstruction losses tends to produce blurry images, so we adopt adversarial training. $G_{\boldsymbol{m}}$ attempts to generate realistic masks to fool a discriminator $D_{\boldsymbol{m}}$, while $D_{\boldsymbol{m}}$ attempts to classify generated masks separately from real training masks. Within this, we also discriminate against our self-supervised transform masks $\widehat{\boldsymbol{m}}'$: these should also fool $D_{\boldsymbol{m}}$. We use a hinge-GAN loss $\mathcal{L}_{\text{Adv}}$ for better training stability [10, 18, 12]:

$$\mathcal{L}_{\text{Adv}}(G_{\boldsymbol{m}}, D_{\boldsymbol{m}}) = \mathbb{E}_{\widehat{\boldsymbol{m}}'}[\min(0, -G_{\boldsymbol{m}}(\widehat{\boldsymbol{m}}')) - 1] + \quad (5)$$
$$2\mathbb{E}_{\boldsymbol{m}}[\min(0, G_{\boldsymbol{m}}(\boldsymbol{m}) - 1)] + \mathbb{E}_{\boldsymbol{m}'}[\min(0, -D_{\boldsymbol{m}}(\boldsymbol{m}')) - 1],$$

To reconstruct the 3D shape within a consistent world space, along with $\boldsymbol{m}$ and $\boldsymbol{m}'$, we find that it is sufficient to give the discriminator a mask $\widehat{\boldsymbol{m}}'$ generated from only one random rotation per instance (as similarly found by Henzler et al. [3]), rather than multiple random rotations.

Feature match loss. We improve sharpness by enforcing that real and generated images elicit similar deep feature responses in each layer l of the discriminator $D_M^{(l)}$ [14, 20]:

$$\mathcal{L}_{\text{FM}}(D_{\boldsymbol{m}}) = \mathbb{E}_{\boldsymbol{m}, \boldsymbol{m}', \widehat{\boldsymbol{m}}'}\Big[\Sigma_{l=1}^L \big\|D_{\boldsymbol{m}}^{(l)}(\widehat{\boldsymbol{m}}') - \bar{D}_{\boldsymbol{m}}^{(l)}(\boldsymbol{m})\big\|_2^2 + \big\|D_{\boldsymbol{m}}^{(l)}(\boldsymbol{m}') - \bar{D}_{\boldsymbol{m}}^{(l)}(\boldsymbol{m})\big\|_2^2\Big], \quad (6)$$

where $\bar{D}_{\boldsymbol{m}}^{(l)}$ is the moving average of feature activations in layer l, and L is the number of layers.

Loss Importance via Ablations We show the importance of each component and loss term in Figure 3.

4. Experiments

Datasets. We path trace 3D objects into RGB images and binary masks using ten real-world captured 360° HDR lighting maps of outdoor natural environments. For each image, we randomly rotate the camera around the up vector at a fixed radius away from the object (as per [13]). For animated datasets with pose variation (120–400 frames), we randomly sample frames. As sampling is random, each pose is *not* seen across views or in any temporal or rotation order, and we discard object and camera poses during training.

Results and Baselines. Across datasets, our approach quickly learns a set of Gaussians. These condition our 2D mask generation to produce highly-detailed silhouettes that respect the 3D Gaussian space. The Gaussian representation is still usefully recovered as input data decreases 64× in number, though detail in the mask reduces. For texture, when silhouettes almost overlap across the projected 3D space, such as front/back giraffe views, texture generation can lose detail. However, variation across lighting is disentangled from view and pose, allowing objects to be added to backgrounds with matched lighting and with adjusted pose. Please see our supplemental video for more results.

For 3D representation and image generation, we compare to HoloGAN [13]. Even with many images, the voxel-based HoloGAN struggles to generate high-quality masks

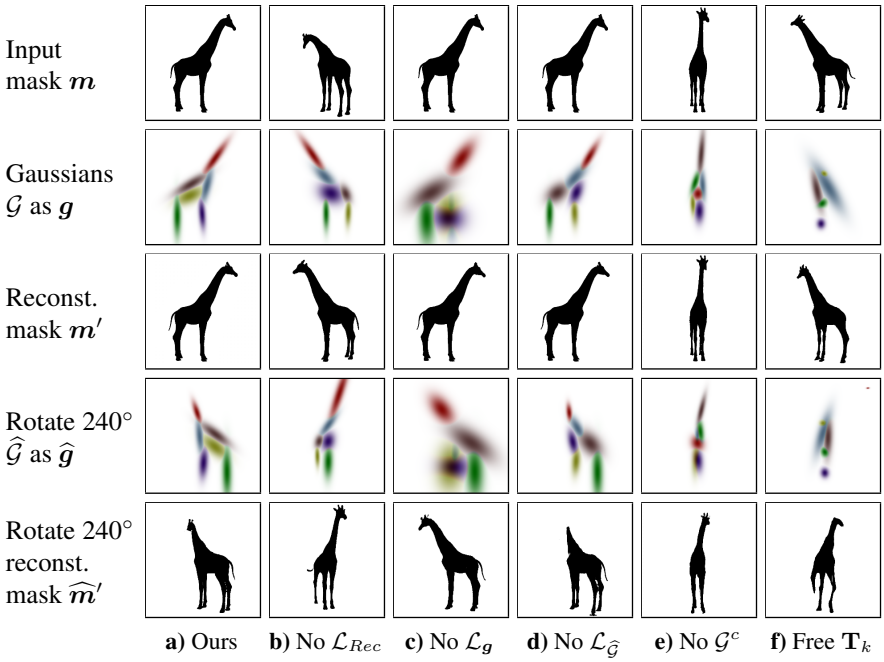

Figure 3. Ablations for *giraffe*. We use different input masks per column as certain effects are only visible at particular angles. Gaussians are colored differently across columns. **(a)** Our full loss model. **(b)** Without a reconstruction loss on m', the Gaussians only approximately correspond to the input mask. **(c)** Without a density loss on g, the Gaussians do not well represent the input mask, yet G_m still produces the correct mask from these less 'coherent' Gaussians. **(d)** Not 'closing the loop' in the self-supervised loss hurts self occlusion cases or when the 2D Gaussian layouts are not sufficient to recover 3D information. **(e)** Not using a canonical representation at all fails to rotate Gaussians recovered for thin front/back views. **(f)** Not bounding the per-instance transforms to reasonable values allows nonsense canonicals.

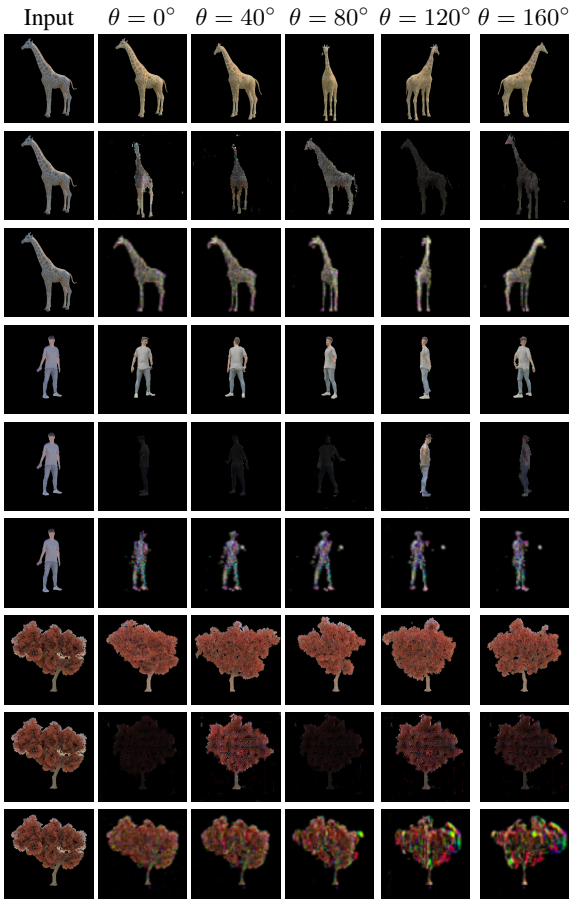

Figure 4. Generation across recovered angles, with any texture latents fixed. For each object, the row ordering is as follows: Ours, HoloGAN [13], PlatonicGAN [3]

when trained on just mask data (Fig. 1), with both part errors (incorrect leg placement) and spurious mask region artifacts. When trained on masked foreground i_f images, which include lighting variation, HoloGAN struggles to generate high-quality masks or texture, and the resulting 3D space mixes variation of both kinds (Fig. 4, *giraffe, manuel*) or fails to rotate the image at all (Fig. 4, *maple*).

We also compare to PlatonicGAN [3], which generates voxel spaces per instance to handle class variation. However, this is not constrained to a canonical model and provides too much freedom to the method. We train it on masked foreground i_f images, and the method outputs masks as part of its process. PlatonicGAN is partly successful at generating consistent voxels (Fig. 1), but introduces geometry errors (misplaced legs, spurious content). The method also sometimes learns to produce spaces with 'double object' impressions. Training on just masks fares similarly. For texture, PlatonicGAN predicts a voxel coloring, which often fails to produce the correct output (Fig. 4).

Conclusion As we move toward 'in the wild' settings, we need intermediate structures for arbitrary objects and training losses that can produce meaningful 3D spaces. We take a step in this direction by implicitly reconstructing a coarse Gaussian representation of object 3D shape and pose, and show a potential use of our approach by conditioning 2D texture generators in a setting where baselines struggle.

Acknowledgements Thank you to Numair Khan for the dataset generator, and for engaging discussions with Helge Rhodin and Srinath Sridhar. Kwang In Kim was supported by the National Research Foundation of Korea (NRF) grant NRF-2021R1A2C2012195, and we thank an Adobe gift.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3D point clouds. In *ICML*, pages 40–49. PMLR, 2018. 1

[2] Kyle Genova, Forrester Cole, Avneesh Sud, Aaron Sarna, and Thomas Funkhouser. Local deep implicit functions for 3D shape. In *CVPR*, pages 4857–4866, 2020. 2

[3] Philipp Henzler, Niloy J Mitra, and Tobias Ritschel. Escaping Plato's cave: 3D shape from adversarial rendering. In *ICCV*, pages 9984–9993, 2019. 1, 2, 3, 4

[4] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, pages 371–386, 2018. 1

[5] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *CVPR*, 2019. 2

[6] Markus Knoche, István Sárándi, and Bastian Leibe. Reposing humans by warping 3D features. In *CVPR Workshop on Towards Human-Centric Image/Video Synthesis*, 2020. 2

[7] Nilesh Kulkarni, Abhinav Gupta, David F Fouhey, and Shubham Tulsiani. Articulation-aware canonical surface mapping. In *CVPR*, pages 452–461, 2020. 2

[8] Jiahui Lei, Srinath Sridhar, Paul Guerrero, Minhyuk Sung, Niloy Mitra, and Leonidas J. Guibas. Pix2Surf: Learning parametric 3D surface models of objects from images. In *ECCV*, 2020. 2

[9] Yiyi Liao, Katja Schwarz, Lars Mescheder, and Andreas Geiger. Towards unsupervised learning of generative models for 3D controllable image synthesis. In *CVPR*, 2020. 2

[10] Jae Hyun Lim and Jong Chul Ye. Geometric GAN. *arXiv preprint arXiv:1705.02894*, 2017. 3

[11] 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. 2

[12] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *ICLR*, 2018. 3

[13] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. HoloGAN: Unsupervised learning of 3D representations from natural images. In *ICCV*, pages 7588–7597, 2019. 1, 2, 3, 4

[14] Tim Saliman, Ian Goodfellow, Wojciech Zaremba, and Vicki Cheung. Improved techniques for training GANs. In *NeurIPS*, 2016. 3

[15] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. GRAF: Generative radiance fields for 3D-aware image synthesis. In *NeurIPS*, 2020. 1, 2

[16] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3D feature embeddings. In *CVPR*, pages 2437–2446, 2019. 2

[17] Srinath Sridhar, Helge Rhodin, Hans-Peter Seidel, Antti Oulasvirta, and Christian Theobalt. Real-time hand tracking using a sum of anisotropic gaussians model. In *3DV*, pages 319–326, 2014. 2

[18] Dustin Tran, Rajesh Ranganath, and David M. Blei. Deep and hierarchical implicit models. *arXiv preprint arXiv:1702.08896*, 7, 2017. 3

[19] Kohei Yamashita, Shohei Nobuhara, and Ko Nishino. 3D-GMNet: Single-view 3D shape recovery as a gaussian mixture. *BMVC*, 2020. 2

[20] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018. 3

[21] Silvia Zuffi, Angjoo Kanazawa, Tanya Berger-Wolf, and Michael J. Black. Three-D safari: Learning to estimate zebra pose, shape, and texture from images "in the wild". In *ICCV*, pages 5358–5367, 2019. 2