SurfGen: Adversarial 3D Shape Synthesis with Explicit Surface Discriminators

Andrew Luo[1] Tianqin Li[1] Wen-Hao Zhang[2] Tai Sing Lee[1]
[1]Carnegie Mellon University [2]University of Chicago

Abstract

Recent advances in deep generative models have led to immense progress in 3D shape synthesis. While existing models are able to synthesize shapes represented as voxels, point-clouds, or implicit functions, these methods only indirectly enforce the plausibility of the final 3D shape surface. Here we present a 3D shape synthesis framework (SurfGen) that directly applies adversarial training to the object surface. Our approach uses a differentiable spherical projection layer to capture and represent the explicit zero isosurface of an implicit 3D generator as functions defined on the unit sphere. By processing the spherical representation of 3D object surfaces with a spherical CNN in an adversarial setting, our generator can better learn the statistics of natural shape surfaces. We evaluate our model on large-scale shape datasets, and demonstrate that the end-to-end trained model is capable of generating high fidelity 3D shapes with diverse topology.

1. Introduction

Shape generation is primarily concerned with the synthesis of diverse, realistic, and novel shapes. High fidelity models of 3D shapes are key to creating immersive virtual worlds, and are important to many disciplines, including architecture, visual effects, and training robots in simulated navigation.

With the introduction of large scale 3D object datasets such as ShapeNet [6] and ModelNet [62], there has been significant progress towards building generative models of 3D shapes. The predominant approach is to perform training using representations that are straightforward for neural networks to work with, such as voxel grids, point clouds, or implicit functions. This is in contrast to the majority of graphics & simulation tasks, which require an triangle mesh representation of a shape. When learned in an adversarial setting, the discriminator in these frameworks only indirectly ensure the realism of the object surface.

A major barrier in training a neural network on object surfaces is the inherent irregularity and discrete nature of 3D surfaces. Different objects within the same object class can have vastly different topologies, and can be characterized by different internal connectivity when represented as a triangle mesh. Our key insight is that a discriminator applied to the surface should focus on the geometric properties of the surface (curvature, orientation, topology, etc.), while ignoring properties that are irrelevant to the shape. Here we propose to transform object surfaces to spherical maps computed with a differentiable function $f_{\mathcal{M}\to S}$. A spherical map is a surface representation defined on discrete samples from the surface of a sphere. The values at each pixel represents a minimal distance of the object surface along a ray, as well as the surface occupancy along the ray. The spherical map is appealing because it is a singular representation that captures the surface geometry of a shape. Given a spherical map, we then utilize a network with spherical convolution layers to complete our discriminator.

Figure 1. Examples of chairs with complex structure generated by our model trained on ShapeNet [6] chair class.

We propose SurfGen, an end-to-end generative model of 3D shapes that is trained with a discriminator which operates on the explicit zero isosurface of an implicit shape function. We demonstrate that applying adversarial training on the surface of an object leads to generated highly realistic shapes. This results in an approach that can generate high quality shapes with arbitrary topology and resolution. Examples of shapes generated by our model are shown in Figure 1.

In summary our contributions are three fold:

- We introduce a spherical projection operator that takes as input an explicit triangle mesh, and is fully differentiable w.r.t. the vertices.
- We propose SurfGen, an end-to-end differentiable 3D shape synthesis framework which applies an adversarial objective on the zero isosurface of the generator.
- We demonstrate our model can synthesize realistic, high quality shapes that have diverse topology.

2. Related Work

Our method is related to prior work on learning statistical models for 3D shape analysis and generation. In this section we will discuss deep learning based models for 3D shapes, spherical projections, and differentiable rendering.

Deep learning for 3D shapes. Recent advances in deep learning have enabled models capable of image-guided shape reconstruction and novel shape synthesis. 3D-R2N2 [9] proposed learning a recurrent model for voxel based reconstruction with multiple views, [16, 19, 60, 69] further improved image based voxel reconstruction. Due to memory constraints, these methods are usually limited in terms of their resolution. Methods have also been proposed to generate point clouds from images [14, 56], however the unordered nature of point clouds limits the fidelity of the final 3D reconstructions. Some approaches have been proposed to deform a mesh template [23, 57, 49], or a set of surface patches [18]. These techniques are typically limited to modeling shapes of fixed topology, or produce meshes that require post-processing to become usable. It has become possible recently to regress *implicit functions* in the form of binary occupancy or signed distances [63, 8, 38, 41]. Similarly, other forms of learned 3D priors have been applied to convert or refine existing shape representations [59, 4, 13].

Generative models for 3D shapes. Modern generative models for 3D shapes generally utilize an adversarial constraint, take a variational approach, or use flow-based models. Of note, [62] design a deep belief network to synthesize novel shapes when trained on a large-scale dataset. [61] proposed using a generative adversarial network (GAN) for voxels. [1, 68] proposes learning latent-GANs [37] for point clouds, while [28] proposes an autoencoding objective to stabilize the training of point cloud generative models. Similarly [15, 53, 46] advocate for modified generators to improve point cloud generation. Implicit generative models based on latent-GANs [8], and point cloud discriminators [27] have also been proposed. While these methods are capable of generating high resolution shapes, they still struggle with thin structures and implausible objects.

Spherical representations. A spherical representation captures information about a shape on the surface of a sphere. For the purpose of shape retrieval, [2, 25, 54] represent shapes as spherical distance functions, [66] captures the number of surface intersections. Spherical representations have also been used in modern deep learning systems for object recognition [5]. [69] uses a non-differentiable spherical representation computed from voxel grids to facilitate 3D shape reconstruction in the projected spherical space, demonstrating the effectiveness of spherical projections for shape synthesis. In our work, we propose a fully differentiable spherical projection to capture the statistics of surfaces, allowing for the end-to-end training of our generator.

Differentiable rendering. A spherical projection can be viewed as a rendering operation using a non-linear projection operator. Because the rendering operation is normally discrete, it does not provide usable error gradients for optimization. A variety of mesh [10, 33, 29, 24, 31, 8, 43], point-cloud, and implicit [32, 40, 39, 48] based differentiable renderers have been proposed. [24] developed an approximation of gradient for rasterization. [50] proposes learning high frequency faces details using a differentiable renderer. [35] also demonstrated the feasibility of differentiable rendering in 3D scene optimization. [64] proposed a differentiable volume sampling method which can approximate the ray tracing algorithm. Other works such as [22, 21] also implemented differentiable operation for single image 3D shape reconstruction tasks. Our work enables gradient based optimization in a spherical projection layer, allowing a generator to be optimized by error signals defined in the spherical domain.

3. Approach

We propose **SurfGen**, an end-to-end fully differentiable framework for the generation of 3D shapes. Key to our model, is a differentiable spherical projection operator which allows surfaces to be represented as a spherical projection map, where an adversarial loss can be naturally applied. Our model is illustrated in Figure 2.

3.1. 3D Shape Generator

We adopt DeepSDF [41] as the generator in our model. In this generator, each shape is represented by an implicit signed distance function (SDF). For each point $\mathbf{p} = (p_x, p_y, p_z)$ and a given shape s, the SDF encodes the distance of the point to the nearest surface: $\text{SDF}_s(\mathbf{p}) = d, d \in \mathcal{R}$. The sign of d encodes if the point is inside (negative) or outside (outside) a given shape. Our generator g_ψ is trained to map a randomly sampled latent code z and a position p to a corresponding SDF value:

$$g_\psi(z, \mathbf{p}) \approx \text{SDF}_s(\mathbf{p}) \tag{1}$$

The surface of a shape is implicitly represented by the zero isosurface of the SDF.

3.2. Differentiable triangle mesh extraction

In order to apply a discriminator to the surface of an object represented as an implicit SDF, we need to first find the zero isosurface. We choose to utilize marching cubes [34]

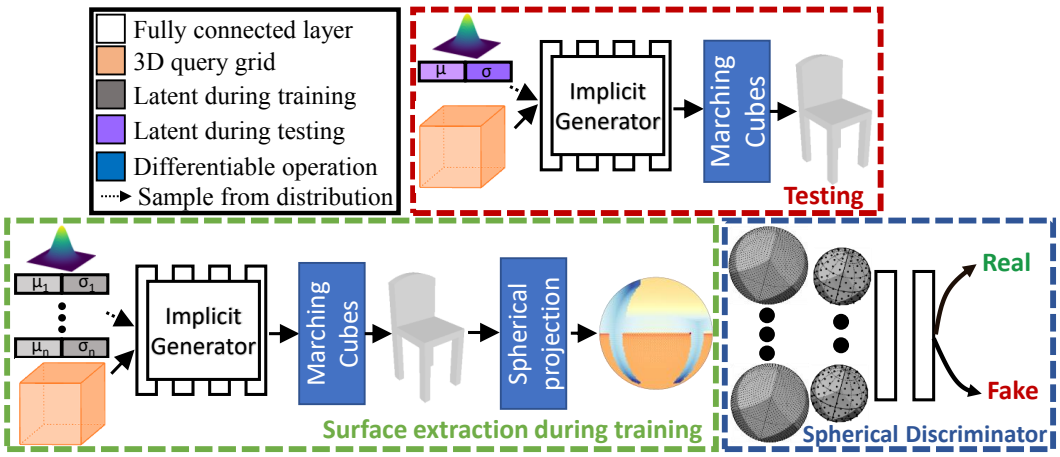

Figure 2. Components of our model. **Top**: During test time, we sample a random code and extract the triangle mesh; **Bottom**: During training, we extract the surface of the shape represented as a triangle mesh, and compute the differentiable spherical projection of the surface. The spherical surface map is fed to a spherical discriminator. Note that we do not utilize an encoder during training.

to extract an explicit triangle mesh from the signed distance function. While sphere tracing can also be utilized to extract the zero isosurface from a signed distance function, we found the required speed-accuracy trade-off unsuitable for use in training.

Instead, we use the MeshSDF method proposed in [44] for differentiable isosurface extraction. During training, we evaluate our signed distance function generator g_ψ on an euclidean grid of size 128^3 in the range of $[-1, 1]$, and use marching cubes (MC) to extract the surface as a triangle mesh $\mathcal{M} = (V, F)$, where $V = \{v_j\}_{j=1}^{M}$ is the set of mesh vertices represented in $\mathbb{R}^3$, and F represents the set of triangle faces enclosed by the edges.

We use the loss from a discriminator D_ϕ to compute a gradient with respect to the vertices V from the zero isosurface. For a discriminator loss $\mathcal{L}$ differentiable w.r.t. vertices V, the gradient with respect to generator weights ψ can be computed by evaluating:

$$n(v) = \nabla g_\psi(v, z) \text{ for } v \in V \tag{2}$$

$$\frac{\partial \mathcal{L}}{\partial \psi} = -\frac{\partial \mathcal{L}}{\partial v} \cdot n(v) \text{ for } v \in V \tag{3}$$

This approach to differentiable surface extraction allows discriminator gradients present on the vertices V to modify surface shape and topology by changing the underlying signed distance function. We refer readers to [44] for a proof and discussion of this method.

3.3. Differentiable spherical surface projection

In this section, we present our diffentiable spherical projection layer. Consider a 3D object parameterized as a triangle mesh $\mathcal{M} = (V, F)$. Because surfaces themselves can vary in topology, and meshes can themselves vary in internal connectivity, it is non-trivial to train a neural network on the surface on an object.

Our spherical projection layer $f_{\mathcal{M}\to S}$ transforms an irregular 3D mesh into a regular spherical domain. The support of the spherical representation is defined as discrete samples on the unit sphere S^2 with $\theta \in [0, 2\pi]$ and $\phi \in [0, \pi]$. There are two major challenges to adopting such a framework. First, the pixel coordinates lie on the surface of a sphere parameterized by (θ, ϕ), which prevents the use of a projection matrix to express the vertex transformations. Second, the rasterization and z-buffer algorithm are discrete operations, which causes discontinuities in the back-propagated gradients as triangles change in depth or move laterally. We use a similar approach to those presented in [31, 7] to enable a differentiable spherical surface projection function.

Because of the non-linear projection, we utilize ray-casting to find intersections of the object surface with rays that originate from the unit sphere. Each ray R_i is defined as a six tuple representing origin and direction in euclidean coordinates:

$$\overrightarrow{R_i} = (O_x, O_y, O_z, D_x, D_y, D_z)_i \tag{4}$$

We modify the ray-intersection kernel to output direct intersections, as well as "near misses" where the ray is within distance r of a triangle. The ray-intersection kernel is repeated such that the k nearest hits along each ray are retrieved:

$$(\overrightarrow{p_{j^i}}, F_j)_{j=1}^{k} = \texttt{rayintersect}_{k,r}(\mathcal{M}, R_i) \tag{5}$$

At an intersection location $\overrightarrow{p_{j^i}}$ inside triangle j for R_i, we compute a pixel attribute u_j^i as an barycentric interpolation of vertex attributes

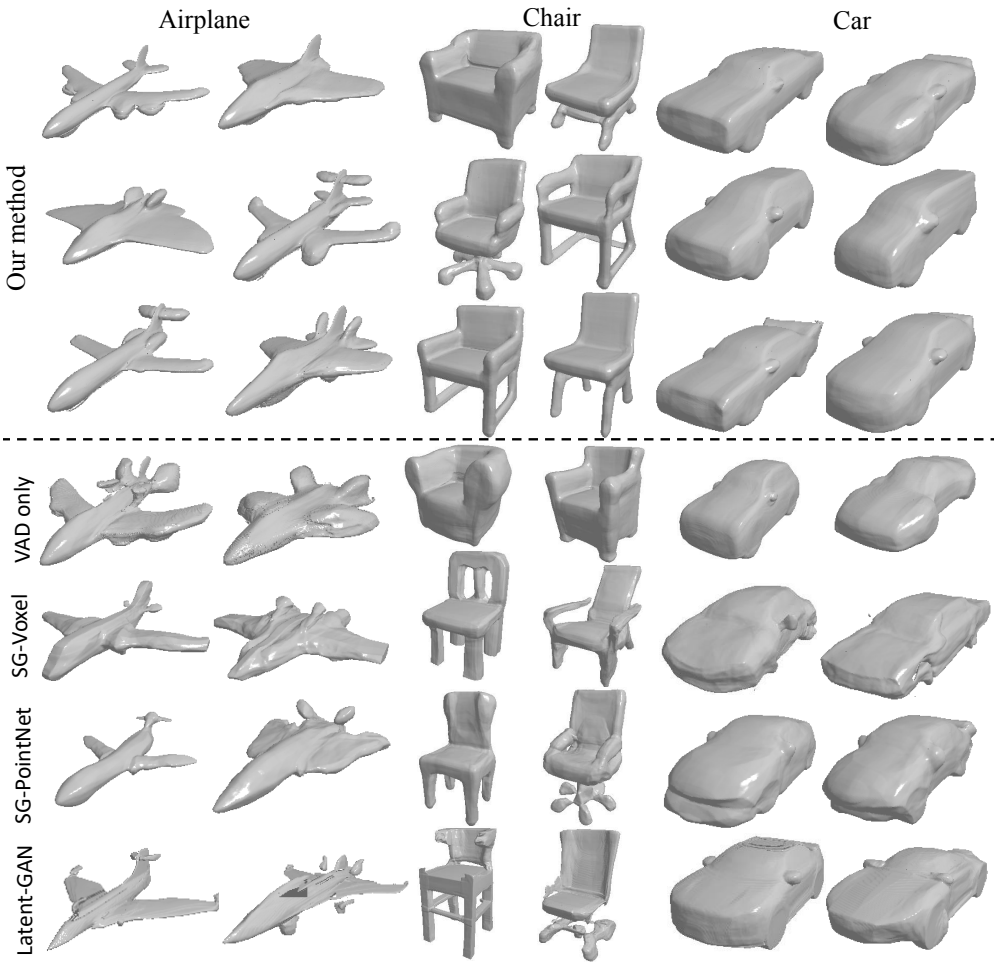

Figure 3. Qualitative results for shapes generated by our SurfGen model; In comparison with samples from other works evaluated on the same testing set: VAD loss only [67], ShapeGAN w/ voxel [27] discriminator, ShapeGAN w/ PointNet discriminator [27], and DeepSDF with a latent generator [1, 8].

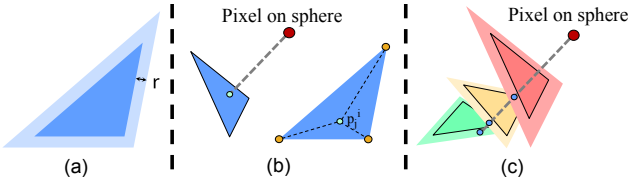

Figure 4. Differentiable spherical projection from the perspective of a ray. **(a)** The search region of each ray includes the triangle (solid) and a small "radius" r neighborhood (transparent). **(b)** Direct hits inside a triangle allows us to interpolate depth. **(c)** "Near miss" hits in the neighborhood of a triangle allows for occupancy to be computed.

$$u_j^i = w_{j,0}^i \cdot u_{j,0} + w_{j,1}^i \cdot u_{j,1} + w_{j,2}^i \cdot u_{j,2} \tag{6}$$

$$w_{j,k}^i = \Omega_k(\overrightarrow{p_j^i}, \overrightarrow{v_{j,0}}, \overrightarrow{v_{j,1}}, \overrightarrow{v_{j,2}}); k = \{0, 1, 2\} \tag{7}$$

where the barycentric weights are computed using the differentiable function Ω. When computing a spherical depth projection, we set the attributes to be the (x, y, z) coordinate of each vertex in triangle j:

$$(u_{j,0}, u_{j,1}, u_{j,1}) = (v_{j,0}, v_{j,1}, v_{j,2}) \tag{8}$$

Beyond the spherical depth map, the spherical silhouette of a shape can also be computed such that it is differentiable. We utilize "near miss" rays that do not intersect inside a triangle, but has a point on that ray $p_{j^i}^{'}$ that is within distance r outside of a triangle. We compute the squared euclidean distance between the ray and the closest point on the triangle:

$$d(p_{j^i}^{'}, F_j) = \min_{p \in F_j} \left\| p - p_{j^i}^{'} \right\|_2^2 \tag{9}$$

$$\alpha_j^i = \exp \frac{-d(p_{j^i}^{'}, F_j)}{\delta} \tag{10}$$

Where δ is a hyper-parameter that controls how fast the influence decays as a function of distance. We use a product based function to aggregate all the near miss collisions for

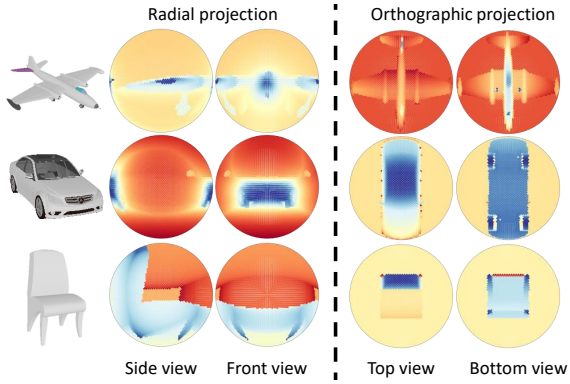

Figure 5. Visualization of radial and orthographic projections. These depth projections reflect the surface distance from the origin of the ray.

a given ray:

$$\alpha^i = 1 - \prod_j (1 - \alpha_j^i) \tag{11}$$

A ray is illustrated in Figure 4. This allows for the efficient computation of spherical depth and occupancy maps in a way that is fully differentiable with respect to the vertex attributes.

3.4. Choice of sampling and projection

Sampling on a Sphere. A discretization of the sphere needs to be selected prior to spherical projection. Common sampling schemes includes Driscoll and Healy [12], icosahedral [58], and HEALPix [17]; which use rectangular, icosahedral, and equal area sampling respectively. Due to our efficient ray-casting based implementation, the differentiable spherical projection layer can work with any discretization of the sphere. In practice, the HEALPix based sampling with $12,288$ points is used due to the balanced area assigned to each point on the sphere.

Choice of Projection. Our spherical projection layer can model arbitrary projections. The most simple projection operator is the *radial spherical projection*, which gives a ray direction to be $(D_x, D_y, D_z) = -(O_x, O_y, O_z)$, where O is discrete sample on the sphere. It was shown in [69], that a radial projection may have poor coverage of a shape due to self-occlusions. They address this by combining the spherical representation with an additional non-radial projection. Taking inspiration from orthographic map projections, we propose *orthographic spherical projection* where $(D_x, D_y, D_z) = -(0, 0, O_z)$, where z is assumed to be the gravity aligned (top-down) axis. Our full spherical projection combines the spherical depth map from radial & orthographic spherical projections, and use the silhouette map from the orthographic projection. The radial silhouette map is omitted since most radial rays have direct intersections within a triangle face, and do not provide a useful training signal. We show how the projections compare in Figure 5.

3.5. Discriminator Implementation

Network Architecture. As our spherical maps are discrete samples on a unit sphere and non-euclidean, we cannot use a regular 2D or 3D convolution. We implement our discriminator using the graph-based spherical convolution layers proposed by [11]. Our discriminator takes as input the 3 spherical maps described in section 3.4. The network consists of 5 residual spherical blocks that each perform average pooling to reduce the total number of spherical samples by a factor of 4, as well as a single self-attention layer in the discriminator. We average pool the final 12 pixels, 256 channel spherical maps, and use fully connected layers to produce the scalar discriminator output. We utilize Instance Normalization [52] and Leaky ReLU [36] in the discriminator.

3.6. Overall model

The straightfoward approach to training an implicit generator with a surface discriminator would be to update the implicit generator and surface discriminator in an alternate fashion. However, unlike the SDF which is defined for all $p \in \mathbb{R}^3$, the surface only exists at select locations. We need to constrain the generator such that the location of the isosurface is within the spherical projection layer, and ideally approximate to the desired shape. Methods using geometric initializations [3], meta-learning [47], and variational methods [20] have been proposed to stabilize training and accelerate convergence. We choose to regularize our generator with a variational autodecoder (VAD) [20] loss alongside our adversarial criterion. This VAD-GAN setup preserves the generator+discriminator structure of a generative adversarial network (GAN) during training, only requiring the addition of a light-weight embedding layer to the model, while significantly outperforming the original VAD only objective.

During training, we sample a latent code z from the approximate posterior for a given shape x_j modeled as a multivariate Gaussian with diagonal covariance:

$$q(z|x_j) := \mathcal{N}(z; \mu_j, \sigma_j^2 \cdot \mathbb{I}) \tag{12}$$

We use the reparameterization trick [26] to allow for direct optimization of latent parameters μ_j, σ_j, and set $z_j = \mu_j + \sigma_j \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, \mathbb{I})$. During training, the generator and latent parameters are trained to minimize the the adversarial objective $\mathcal{L}_{GAN}$ and maximize the evidence lower bound of the marginal likelihood (**ELBO**):

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{z \sim q(z|x)} \Big[\log p(x|z) \Big] - D_{KL}(q(z|x)||p(z))$$

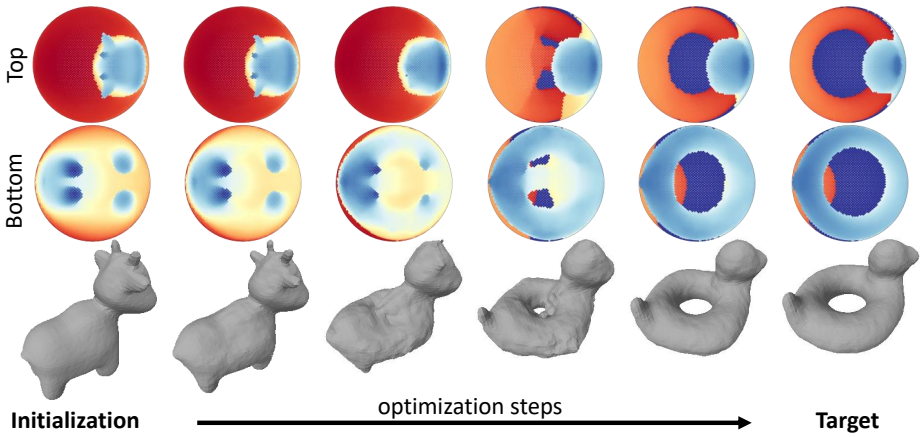

Figure 6. We optimize a MSE loss on the spherical representation between the target and initial shape. **From top to bottom:** the top view of the radial projection, the bottom view of the radial projection, rendered SDF output. This shows that our spherical surface projection can induce topology and shape changes.

$$p(x|z) = -\mathop{\mathbb{E}}_{\mathbf{p}}\left[\mathcal{L}_{SDF}\Big(g_\psi(z,\mathbf{p}), \text{SDF}_s(\mathbf{p})\Big)\right] \tag{13}$$

Equation 13 could be approximated by sampling from 3D space following a certain distribution defined in [41]. For the final results, we use clamped version of L1 loss for $\mathcal{L}_{SDF}(\cdot)$. To stabilize GAN training, we employ standard hinge GAN loss $\mathcal{L}_{\text{hinge}}$ [30, 51] with a small amount of feature matching loss $\mathcal{L}_{\text{feat}}$ [45] applied to the first four discriminator feature maps.

In summary, the overall objective is summarized in Equation 15 where we use $\alpha = 1$, $\beta = 1$ and $\gamma = 1e^{-5}$, $\lambda = 0.5$.

$$\mathcal{L}_{GAN} = \mathcal{L}_{\text{hinge}} + \lambda\mathcal{L}_{\text{feat}} \tag{14}$$

$$\mathcal{L}_{\text{total}} = \alpha\mathcal{L}_{GAN} + \beta\mathcal{L}_{SDF} + \gamma D_{KL} \tag{15}$$

In order to facilitate stable training, we zero the gradients in the generated depth maps where there is no surface occupancy in the corresponding target shape:

$$\text{Depth}_{\text{grad}}[O_{\text{target}} < 1.0] = 0 \tag{16}$$

To ensure that surface gradients w.r.t. the SDF generator are within the same magnitude as the other losses, we scale the gradient from Equation 3 with a constant $\omega = 1e^{-4}$.

Our model is implemented using Pytorch, with the ray casting implemented in embree [55, 42], and marching cubes implemented in CUDA [65].

4. Experiments

4.1. Validating end-to-end optimization

We first verify the differentiability of our spherical projection layer. Using the data from [44], we train a generator network g_ψ to approximate the signed distance function of two shapes: a genus zero cow, and a genus one rubber duck.

	Chamfer			EMD		
Iterations	5	10	30	5	10	30
Radial Depth	0.0486	**0.0327**	0.0032	0.0389	0.0244	0.0029
Ortho Depth	0.0496	0.0335	0.0024	0.0384	0.0234	0.0026
Radial silhouette	0.0584	0.0590	0.0587	0.0446	0.0443	0.0433
Ortho silhouette	0.0556	0.0528	0.0460	0.0443	0.0373	0.0322
Combined	**0.0478**	0.0329	**0.0022**	**0.0356**	**0.0219**	**0.0010**

Table 1. We compare the effectiveness of four spherical features in terms of Chamfer and earth movers distance (EMD) during the optimization process. *Combined* indicates the radial depth, ortho depth, and ortho silhouette are used together. **Bold** indicates best method at each iteration.

The generator learns to associate a latent code with a corresponding implicit distance field. The loss during optimization is the pixel-wise mean squared error (MSE) between the current and target spherical projections:

$$\frac{1}{N}(f_{\mathcal{M}\to S}(g_\psi(z)) - S_{\text{target}})^2 \tag{17}$$

The gradients are backpropagated and are used to optimize the latent code z. As shown in Figure 6, our spherical projection layer can modify the underlying implicit representation, and can change both the shape and topology of an object.

We quantitatively compare the effect of using radial depth, radial silhouette, orthographic depth, orthographic silhouette, as well as combined (radial depth + orthographic depth + orthographic silhouette) as features for our optimization in Table 1. Because few rays in the radial projection experience a "near miss" for the shape, the radial silhouette does not provide useful gradients for the optimization process. The loss that uses the three combined features provides the fastest convergence when measured by chamfer distance or earth movers distance. The final projection used in our shape generation experiment uses the combined features.

Category	Model	Discriminator	JSD (↓)	MMD (↓)		COV (%, ↑)	
				CD	EMD	CD	EMD
Airplane	ShapeGAN-Voxel	Voxel	0.2848	0.0193	0.1818	0.0794	0.0918
	ShapeGAN-PN	PointNet	0.2393	0.0119	0.1670	0.1066	0.1191
	Latent-GAN	Latent	0.3643	0.0138	0.1871	0.0843	0.0893
	VAD-SDF	None	0.2221	0.0103	0.1500	0.1141	0.1190
	SurfGen (ours)	Surface	**0.1586**	**0.0074**	**0.1371**	**0.1191**	**0.1215**
Chair	ShapeGAN-Voxel	Voxel	0.0307	0.0231	0.2042	0.3408	0.3537
	ShapeGAN-PN	PointNet	0.0304	0.0152	0.1711	0.3641	**0.3868**
	Latent-GAN	Latent	0.0394	0.0125	0.1660	0.3742	0.3667
	VAD-SDF	None	0.0846	0.0142	0.1662	0.2051	0.2180
	SurfGen (ours)	Surface	**0.0287**	**0.0095**	**0.1440**	**0.3812**	0.3586
Car	ShapeGAN-Voxel	Voxel	0.0336	0.0056	0.1221	0.3181	0.2869
	ShapeGAN-PN	PointNet	**0.0259**	0.0051	0.1061	**0.3409**	**0.3579**
	Latent-GAN	Latent	0.0649	0.0061	0.1292	0.2784	0.2556
	VAD-SDF	None	0.0568	0.0048	0.1063	0.2414	0.2585
	SurfGen (ours)	Surface	0.0463	**0.0038**	**0.0982**	0.2755	0.3267

Table 2. Generation results across ShapeNet classes; ↓ indicates that a lower value is better, ↑ indicates a higher value is better.

4.2. Shape Generation

Data preparation. All models are trained on one of three categories from ShapeNet.v2 [6]: *airplane*, *car*, and *chair*. To ensure comparability, we use the official training split.

Each shape is centered and normalized to the unit sphere. We utilize the improved signed distance generation method proposed by the authors of ShapeGAN. Each shape is rendered from 50 equidistance views, the depth buffer is projected into object space to compute the surface point cloud. A point is considered to be outside the shape if it is seen by any camera. Shapes are discarded if fewer than 0.5% of the points are inside. We apply a small negative offset ($2e^{-3}$) to SDF values during training to facilitate surface extraction. This results in 2788, 2452, 4550 shapes in the training set for the *airplane*, *car*, and *chair* categories respectively. We use the same sampling scheme used in DeepSDF. For the ground truth test set, we uniformly query points in a unit sphere, and randomly select 2048 points thresholded to lie near the surface.

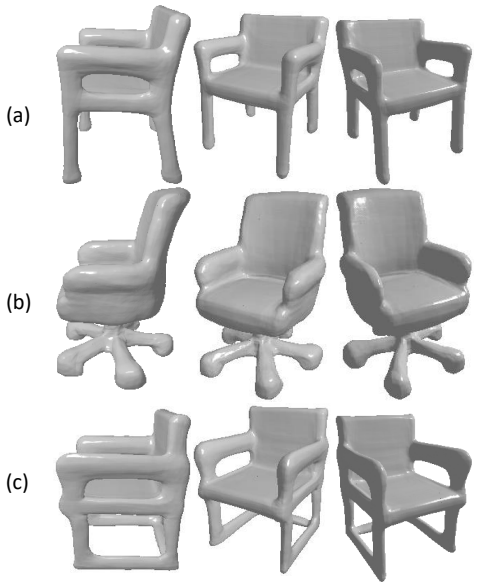

Figure 7. Multiple views of synthesized shapes by our model.

Baselines. We compare against four alternative models for implicit shape generation: two variants of ShapeGAN [27] which use a voxel and PointNet as discriminator, a latent-GAN trained on DeepSDF embeddings, and a VAD based SDF model which does not utilize our surface discriminator. We use the VAD layer implementation provided in [20]. For the latent-GAN, VAD, and our SurfGen model, we use the DeepSDF network as backbone. For ShapeGAN the author provided code and hyperparameters are used.

Evaluation and metrics. Every mesh is extracted using marching cubes at 256^3 resolution, and 2048 points are randomly sampled from each mesh. The results are compared using the metrics introduced by [1]. Minimum matching distance (MMD) measures the distance of a shape from the test set to its nearest neighbor, and can be understood to be a proxy for *fidelity*. Coverage (COV) measures the fraction of the test set that are the nearest neighbor to a sample in the generated set, and can be understood to be a proxy for diversity. For each measure, the distance metric can either be chamfer distance (CD) or earth movers distance (EMD).

Results. Qualitative results for shapes generated by each method are shown in Figure 3 and Figure 7, while quantitative results are shown in Table 2. Shapes generated by our model have high-quality surfaces that are largely free from the high frequency artifacts present in Latent-GAN, and the block like artifacts from the two ShapeGAN variants. We also observe many thin shell like artifacts on VAD synthesized airplanes. For the chair class, the VAD only method

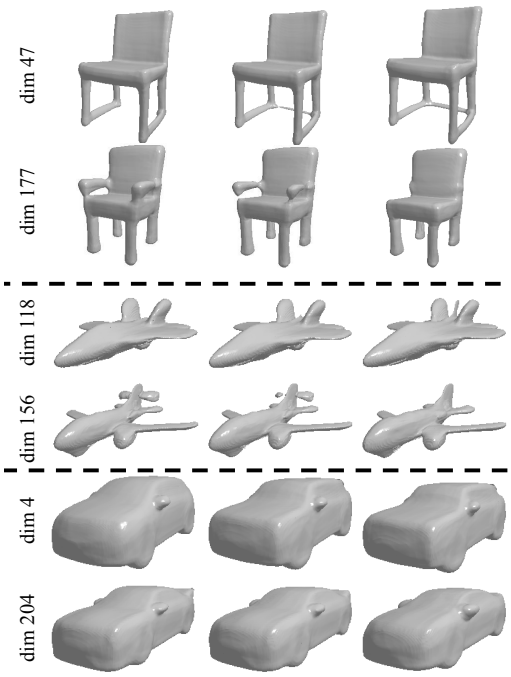

Figure 8. The effect of manipulating individual dimensions of the latent vector

struggles to capture shapes with complicated geometry, and generally synthesizes chairs with no thin parts.

SurfGen has the lowest MMD value across all evaluated classes for both CD & EMD, this indicates that our methods synthesizes shapes are closely matched to samples from the test set. SurfGen produces high coverage for the airplane class, while it is tied for coverage in the chair class. Our method is less competitive in the car class. A possible reason is the lack of high frequency geometry normally present on the outside surface of cars. This causes cars to be difficult to learn using a surface based adversarial loss.

4.3. Latent Analysis

Individual latent dimensions. We explore if individual dimensions in our latent space have semantic meaning. Given a randomly sampled latent code, we select a dimension and increase its value in small increments. Our results are shown in Figure 8. We observe that changing certain dimensions can induce structural changes in a shape.

Interpolating between latents. In Figure 9, we demonstrate that shapes can smoothly change as we linearly interpolate between two randomly sampled latent vectors, even between shapes that have different topologies.

Computational efficiency. For a batch of 8 shapes, on a dual `Nvidia 3090` system the grid query step at 128^3 resolution takes approximately $2,500$ ms, while marching cubes takes 90 ms. The spherical projection takes around 600 ms on a 12 core CPU. For G and D combined, the forward and backwards pass for a single batch takes around 5 seconds total.

Figure 9. Visualization of the shape as we linearly interpolate between two randomly sampled latent vectors.

Figure 10. Two failure cases that exhibit ring like artifacts near the wing.

4.4. Failure Cases

Certain shapes do not have well behaved spherical projections. In the most common case, we observe that this is mostly due to a shape having protrusions along the gravity aligned axis that significantly affect the centering of a shape into a unit sphere. The issue is most prominent in *airplanes*, while this is to a large degree mitigated by using the orthographic projection, we observe rare ring-like artifacts in airplanes as shown in Figure 10 that seem to be induced by the spherical discriminator. As part of future work, it may make sense to investigate the integration of the surface discriminator with discriminators directly applied to the implicit field.

5. Conclusion

In this paper, we have introduced a novel shape synthesis method that allows a discriminator to focus directly on the surface on a generated shape. Our method is capable of generating diverse high quality shapes with complex geometry. Our model has diverse downstream applications as part of a larger 3D synthesis pipeline. We hope our work will inspire future research in 3D shape synthesis.

6. Acknowledgment

This work was supported by an NSF grant CISE RI 1816568 and an interdisciplinary training fellowship in computational neuroscience at Carnegie Mellon sponsored by NIH NIDA grant 5T90 DA023426.

References

[1] Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning representations and generative models for 3d point clouds. In *International conference on machine learning*, pages 40–49. PMLR, 2018. 2, 4, 7

[2] Mihael Ankerst, Gabi Kastenmüller, Hans-Peter Kriegel, and Thomas Seidl. 3d shape histograms for similarity search and classification in spatial databases. In *International symposium on spatial databases*, pages 207–226. Springer, 1999. 2

[3] Matan Atzmon and Yaron Lipman. Sal: Sign agnostic learning of shapes from raw data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2565–2574, 2020. 5

[4] Abhishek Badki, Orazio Gallo, Jan Kautz, and Pradeep Sen. Meshlet priors for 3d mesh reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2849–2858, 2020. 2

[5] Zhangjie Cao, Qixing Huang, and Ramani Karthik. 3d object classification via spherical projections. In *2017 International Conference on 3D Vision (3DV)*, pages 566–574. IEEE, 2017. 2

[6] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 1, 7

[7] Wenzheng Chen, Jun Gao, Huan Ling, Edward Smith, Jaakko Lehtinen, Alec Jacobson, and Sanja Fidler. Learning to predict 3d objects with an interpolation-based differentiable renderer. In *Advances In Neural Information Processing Systems*, 2019. 3

[8] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 2, 4

[9] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 2

[10] Martin de La Gorce, David J Fleet, and Nikos Paragios. Model-based 3d hand pose estimation from monocular video. *IEEE transactions on pattern analysis and machine intelligence*, 33(9):1793–1805, 2011. 2

[11] Michaël Defferrard, Martino Milani, Frédérick Gusset, and Nathanaël Perraudin. Deepsphere: a graph-based spherical cnn. *arXiv preprint arXiv:2012.15000*, 2020. 5

[12] James R Driscoll and Dennis M Healy. Computing fourier transforms and convolutions on the 2-sphere. *Advances in applied mathematics*, 15(2):202–250, 1994. 5

[13] Philipp Erler, Paul Guerrero, Stefan Ohrhallinger, Niloy J Mitra, and Michael Wimmer. Points2surf learning implicit surfaces from point clouds. In *European Conference on Computer Vision*, pages 108–124. Springer, 2020. 2

[14] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 2

[15] Matheus Gadelha, Rui Wang, and Subhransu Maji. Multiresolution tree networks for 3d point cloud processing. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 103–118, 2018. 2

[16] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016. 2

[17] Krzysztof M Gorski, Eric Hivon, Anthony J Banday, Benjamin D Wandelt, Frode K Hansen, Mstvos Reinecke, and Matthia Bartelmann. Healpix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere. *The Astrophysical Journal*, 622(2):759, 2005. 5

[18] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 216–224, 2018. 2

[19] Christian Häne, Shubham Tulsiani, and Jitendra Malik. Hierarchical surface prediction for 3d object reconstruction. In *2017 International Conference on 3D Vision (3DV)*, pages 412–420. IEEE, 2017. 2

[20] Zekun Hao, Hadar Averbuch-Elor, Noah Snavely, and Serge Belongie. Dualsdf: Semantic shape manipulation using a two-level representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7631–7641, 2020. 5, 7

[21] Paul Henderson and Vittorio Ferrari. Learning to generate and reconstruct 3d meshes with only 2d supervision. *arXiv preprint arXiv:1807.09259*, 2018. 2

[22] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018. 2

[23] Abhishek Kar, Shubham Tulsiani, Joao Carreira, and Jitendra Malik. Category-specific object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1966–1974, 2015. 2

[24] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, 2018. 2

[25] Michael Kazhdan, Thomas Funkhouser, and Szymon Rusinkiewicz. Rotation invariant spherical harmonic representation of 3 d shape descriptors. In *Symposium on geometry processing*, volume 6, pages 156–164, 2003. 2

[26] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 5

[27] Marian Kleineberg, Matthias Fey, and Frank Weichert. Adversarial generation of continuous implicit shape representations. *arXiv preprint arXiv:2002.00349*, 2020. 2, 4, 7

[28] Chun-Liang Li, Manzil Zaheer, Yang Zhang, Barnabas Poczos, and Ruslan Salakhutdinov. Point cloud gan. *arXiv preprint arXiv:1810.05795*, 2018. 2

[29] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Transactions on Graphics (TOG)*, 37(6):1–11, 2018. 2

[30] Jae Hyun Lim and Jong Chul Ye. Geometric gan. *arXiv preprint arXiv:1705.02894*, 2017. 6

[31] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7708–7717, 2019. 2, 3

[32] Shaohui Liu, Yinda Zhang, Songyou Peng, Boxin Shi, Marc Pollefeys, and Zhaopeng Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2019–2028, 2020. 2

[33] Matthew M Loper and Michael J Black. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pages 154–169. Springer, 2014. 2

[34] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *ACM siggraph computer graphics*, 21(4):163–169, 1987. 2

[35] Andrew Luo, Zhoutong Zhang, Jiajun Wu, and Joshua B Tenenbaum. End-to-end optimization of scene layout. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3754–3763, 2020. 2

[36] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3. Citeseer, 2013. 5

[37] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015. 2

[38] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4460–4470, 2019. 2

[39] 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*, pages 405–421. Springer, 2020. 2

[40] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3504–3515, 2020. 2

[41] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 2, 6

[42] Samuel F. Potter. sampotter/python-embree:, Mar. 2021. 6

[43] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv preprint arXiv:2007.08501*, 2020. 2

[44] Edoardo Remelli, Artem Lukoianov, Stephan R Richter, Benoît Guillard, Timur Bagautdinov, Pierre Baque, and Pascal Fua. Meshsdf: Differentiable iso-surface extraction. *arXiv preprint arXiv:2006.03997*, 2020. 3, 6

[45] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29:2234–2242, 2016. 6

[46] Dong Wook Shu, Sung Woo Park, and Junseok Kwon. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3859–3868, 2019. 2

[47] Vincent Sitzmann, Eric R Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *arXiv preprint arXiv:2006.09662*, 2020. 5

[48] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *arXiv preprint arXiv:1906.01618*, 2019. 2

[49] Edward J Smith, Scott Fujimoto, Adriana Romero, and David Meger. Geometrics: Exploiting geometric structure for graph-encoded objects. *arXiv preprint arXiv:1901.11461*, 2019. 2

[50] Ayush Tewari, Michael Zollhöfer, Pablo Garrido, Florian Bernard, Hyeongwoo Kim, Patrick Pérez, and Christian Theobalt. Self-supervised multi-level face model learning for monocular reconstruction at over 250 hz. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2549–2559, 2018. 2

[51] Dustin Tran, Rajesh Ranganath, and David M Blei. Deep and hierarchical implicit models. *arXiv preprint arXiv:1702.08896*, 7(3):13, 2017. 6

[52] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 5

[53] Diego Valsesia, Giulia Fracastoro, and Enrico Magli. Learning localized generative models for 3d point clouds via graph convolution. In *International conference on learning representations*, 2018. 2

[54] Dejan V Vranic, Dietmar Saupe, and Jörg Richter. Tools for 3d-object retrieval: Karhunen-loeve transform and spherical harmonics. In *2001 IEEE Fourth Workshop on Multimedia Signal Processing (Cat. No. 01TH8564)*, pages 293–298. IEEE, 2001. 2

[55] Ingo Wald, Sven Woop, Carsten Benthin, Gregory S Johnson, and Manfred Ernst. Embree: a kernel framework for efficient cpu ray tracing. *ACM Transactions on Graphics (TOG)*, 33(4):1–8, 2014. 6

[56] Jinglu Wang, Bo Sun, and Yan Lu. Mvpnet: Multi-view point regression networks for 3d object reconstruction from a single image. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8949–8956, 2019. 2

[57] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh

models from single rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 52–67, 2018. 2

[58] Magnus J Wenninger. *Spherical models*, volume 3. Courier Corporation, 1999. 5

[59] Francis Williams, Teseo Schneider, Claudio Silva, Denis Zorin, Joan Bruna, and Daniele Panozzo. Deep geometric prior for surface reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10130–10139, 2019. 2

[60] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. In *Advances in neural information processing systems*, pages 540–550, 2017. 2

[61] Jiajun Wu, Chengkai Zhang, Tianfan Xue, Bill Freeman, and Josh Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *Advances in neural information processing systems*, 29:82–90, 2016. 2

[62] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015. 1, 2

[63] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In *Advances in Neural Information Processing Systems*, pages 492–502, 2019. 2

[64] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Advances in neural information processing systems*, pages 1696–1704, 2016. 2

[65] Tatsuya Yatagawa. mcubes_pytorch: Pytorch implementation for marching cubes. 6

[66] Meng Yu, Indriyati Atmosukarto, Wee Kheng Leow, Zhiyong Huang, and Rong Xu. 3d model retrieval with morphing-based geometric and topological feature maps. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–656. IEEE, 2003. 2

[67] Amir Zadeh, Yao-Chong Lim, Paul Pu Liang, and Louis-Philippe Morency. Variational auto-decoder. *arXiv preprint arXiv:1903.00840*, 2019. 4

[68] Maciej Zamorski, Maciej Zieba, Piotr Klukowski, Rafał Nowak, Karol Kurach, Wojciech Stokowiec, and Tomasz Trzciński. Adversarial autoencoders for compact representations of 3d point clouds. *arXiv preprint arXiv:1811.07605*, 2018. 2

[69] Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Josh Tenenbaum, Bill Freeman, and Jiajun Wu. Learning to reconstruct shapes from unseen classes. *Advances in neural information processing systems*, 31:2257–2268, 2018. 2, 5