Supplemental Material
MVDD: Multi-View Depth Diffusion Models

Zhen Wang[1,2†], Qiangeng Xu[1], Feitong Tan[1], Menglei Chai[1], Shichen Liu[1], Rohit Pandey[1], Sean Fanello[1], Achuta Kadambi[1,2], and Yinda Zhang[1]

[1] Google
[2] University of California, Los Angeles
https://mvdepth.github.io/

1 Content

The supplemental material is organized as follows:

- In Sec. 2, we introduce metrics used in the main paper;
- In Sec. 3, we introduce the detailed camera embeddings and setup;
- In Sec. 4, we provide more details on the network architectures;
- In Sec. 5, we present more quantitative and qualitative results in addition to those in the main paper;
- In Sec. 6, we list used codebases in this paper.

2 Metrics

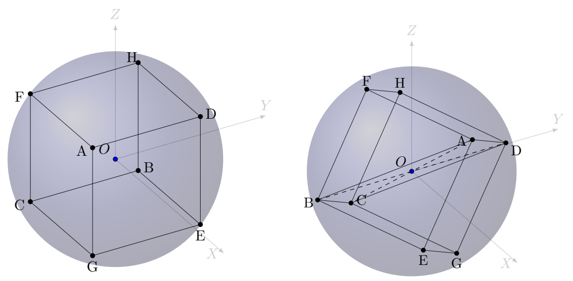
In line with previous research, we employ Chamfer distance (CD) and earth mover's distance (EMD) to quantify the similarity among point clouds. Specifically, their formal definitions are as follows, in accordance with established methods:

$$\begin{aligned} \mathrm{CD}(X,Y) &= \sum_{x\in X}\min_{y\in Y}\|x-y\|_2^2 + \sum_{y\in Y}\min_{x\in X}\|x-y\|_2^2, \\ \mathrm{EMD}(X,Y) &= \min_{\gamma:X\to Y}\sum_{x\in X}\|x-\gamma(x)\|_2. \end{aligned} \tag{1}$$

Given two point clouds, X and Y, with an equal number of points and γ representing a bijection between them. Consider $\mathcal{S}_g$ as the collection of generated point clouds and $\mathcal{S}_r$ as the reference set of point clouds, both having an equal cardinality with $|\mathcal{S}_r| = |\mathcal{S}_r|$.

Coverage (COV) calculates the proportion of point clouds within the reference set that find at least one corresponding match within the generated set. Every point cloud within the generated set is paired with its closest neighbor within the reference set, considering it a match.

$$\mathrm{COV}\left(\mathcal{S}_g, \mathcal{S}_r\right) = \frac{\left|\left\{\arg\min_{Y\in\mathcal{S}_r} D(X,Y) \mid X \in \mathcal{S}_g\right\}\right|}{|\mathcal{S}_r|}. \tag{2}$$



(a) Fixed camera **(b)** Dynamic camera

Fig. 1: Camera setup demonstration.

When considering $D(\cdot,\cdot)$, which can be CD or EMD, it's important to note that while coverage can identify mode collapse, it does not assess the quality of generated point clouds. Surprisingly, it is feasible to attain a flawless coverage score even when the distances between generated and reference point clouds are excessively vast.

As a complementary to COV, Minimum matching distance (MMD) involves calculating, for every point cloud within the reference set, the average distance to its closest neighbor in the generated set:

$$\mathrm{MMD}\left(\mathcal{S}_g, \mathcal{S}_r\right) = \frac{1}{|\mathcal{S}_r|} \sum_{Y \in \mathcal{S}_r} \min_{X \in \mathcal{S}_g} D(X, Y), \tag{3}$$

where $D(\cdot,\cdot)$ is chosen as either CD or EMD. The concept behind MMD involves computing the mean distance between point clouds within the reference set and their nearest counterparts in the generated set, thus serving as a metric to assess quality.

Lopez-Paz and Oquab [5] introduced this method for conducting two-sample tests, determining the similarity between two distributions. Additionally, it has been investigated as a metric for assessing GANs [13]. When considering $\mathcal{S}_{-X} = \mathcal{S}_r \cup \mathcal{S}_g - X$ and N_X representing the nearest neighbor of X in $\mathcal{S}_{-X}$, 1-NNA denotes the leave-one-out accuracy of the 1-nearest neighbor classifier.

$$\begin{aligned} &1 - \mathrm{NNA}\left(\mathcal{S}_g, \mathcal{S}_r\right) \\ &\qquad = \frac{\sum_{X \in \mathcal{S}_g} \mathbb{I}\left[N_X \in \mathcal{S}_g\right] + \sum_{Y \in \mathcal{S}_r} \mathbb{I}\left[N_Y \in \mathcal{S}_r\right]}{|\mathcal{S}_g| + |\mathcal{S}_r|}, \end{aligned} \tag{4}$$

where the function $\mathbb{I}[\cdot]$ is the indicator. With every instance, the 1-NN classifier distinguishes it as belonging to either $\mathcal{S}_r$ or $\mathcal{S}_g$, depending on the closest sample's label. If $\mathcal{S}_r$ and $\mathcal{S}_g$ are drawn from the same source, this classifier's accuracy should eventually stabilize at 50% with a sufficient number of samples. The closer this accuracy gets to 50%, the more similarity exists between $\mathcal{S}_g$ and

$\mathcal{S}_r$, indicating the model's proficiency in modeling the target distribution. In our scenario, computing the nearest neighbor can be done using CD or EMD. Unlike COV and MMD, 1-NNA directly gauges the similarity in distributions, considering both diversity and quality simultaneously.

3 Cameras

3.1 Camera embeddings

In each step of the diffusion process, we also condition a set of extrinsic camera parameters $c \in \mathbb{R}^{F\times 16}$ into the model. According to [11], we found that embedding camera parameters with a 2-layer MLP leads to the most satisfying image quality with distinguishable view differences and we also add camera embeddings to time embeddings as residuals.

$$\mathbf{z} = \mathrm{MLP}\left(\mathrm{PosEnc}\left(\sqrt{\bar{\alpha}_t}\right)\right) + \mathrm{MLP}(\mathrm{Embedding}(c)), \tag{5}$$

where PosEnc is the sinusoidal positional encoding used in Transformer [12] as follows:

$$\begin{aligned}\mathrm{PosEnc}_{t,i}^{(1)} &= \sin\left(\frac{t}{10000^{\frac{i}{d-1}}}\right),\\ \mathrm{PosEnc}_{t,i}^{(2)} &= \cos\left(\frac{t}{10000^{\frac{i}{d-1}}}\right).\end{aligned} \tag{6}$$

Embedding is the embedding layer. We use adaptive group normalization layers (AdaGN) following [4] to incorporate embedding vector $\mathbf{z}$ into every residual block. AdaGN realize group normalization through channel-wise scaling and shifting w.r.t. normalized feature maps in each block of U-Net.

3.2 Camera placement

For shape generation task as we describe in Sec. 4.1 in the main paper, since we have the control over the location of the camera, we place the eight cameras on the eight vertices (A, B, C, D, E, F, G and H) of a cuboid as shown in Fig. 1(a) to make sure it covers the whole object. All vertices are on a sphere of radius $r = \sqrt{3}$. Specifically, vertices A, F, H, D are placed with elevation angle 30° and azimuth angle 45°, 135°, 225° and 315°, respectively. Vertices G, C, B, E are placed with elevation angle −10° and azimuth angle 45°, 135°, 225° and 315°, respectively.

For depth completion task as we describe in Sec. 4.2 in the main paper, as the input depth image might come with any random pose, therefore, we place the first camera (denoted as node A in Fig. 1(b)) in a random location on the sphere. After that, we want to uniquely obtain the locations of the remaining cameras assuming that all the cameras are on a cube. Denoting the coordinate of the first freely placed camera A as (X, Y, Z), we assume that the plane $ABCD$

Table 1: Hyper-parameters for diffusion models.

Parameters	Values
Learning rate	$2e^{-4}$
Number of levels	4
U-Net base channels	64
U-Net channel multiplier	1, 2, 4, 8
U-Net residual block groups	8
U-Net attention head	4
U-Net attention head channels	32
U-Net norm layer type	GroupNorm
Diffusion steps	1000
Noise schedule	Cosine
Clip input range	$[-1, 1]$

always coincides with the plane $x/X = z/Z$. In this way, the coordinates of the remaining vertices can be obtained as:

$$\begin{aligned}
B &= \text{Rot}\left(n_{ABCD}, 2\arctan(\sqrt{2})\right) \cdot A, \\
C &= \text{Rot}\left(n_{ABCD}, 2\arctan(\sqrt{2}) + 2\arctan(1/\sqrt{2})\right) \cdot A, \\
D &= \text{Rot}\left(n_{ABCD}, 4\arctan(\sqrt{2}) + 2\arctan(1/\sqrt{2})\right) \cdot A, \\
E &= \frac{A+B}{2} + \frac{AB \times AD}{||AB \times AD||} \cdot \frac{||AB||}{2}, \\
F &= \frac{A+B}{2} - \frac{AB \times AD}{||AB \times AD||} \cdot \frac{||AB||}{2}, \\
G &= \frac{C+D}{2} + \frac{AB \times AD}{||AB \times AD||} \cdot \frac{||AB||}{2}, \\
H &= \frac{C+D}{2} - \frac{AB \times AD}{||AB \times AD||} \cdot \frac{||AB||}{2},
\end{aligned} \tag{7}$$

where n_{ABCD} is the normal vector of the plane $ABCD$ and $\text{Rot}(\cdot,\cdot)$ represents the rotation matrix with the first term as the axis to be rotated along and the second term as the angle to rotate along the axis. To be concrete, following Euler–Rodrigues formula [3], we have:

$$\text{Rot}(\text{n}, \theta) = \begin{bmatrix} a^2+b^2-c^2-d^2 & 2(bc-ad) & 2(bd+ac) \\ 2(bc+ad) & a^2+c^2-b^2-d^2 & 2(cd-ab) \\ 2(bd-ac) & 2(cd+ab) & a^2+d^2-b^2-c^2 \end{bmatrix}, \tag{8}$$

where $a = \cos(\theta/2)$, $b, c, d = -\frac{n}{||n||}\sin(\theta/2)$.

Table 2: Unconditional generation on ShapeNet categories. MMD (CD) is multiplied by 10^3. 🟡 represents the best result and ⚪ represents the second-best result.

		Vox-diff [15]	DPM [6]	3D-LDM [7]	IM-GAN [2]	PVD [15]	LION [14]	MVDD (Ours)
Airplane	MMD (CD)	1.322	1.300	4.000	0.800 🟡	1.300	1.100	0.900 ⚪
	COV(CD)	11.82	37.95	46.20	47.90	43.97	50.74 🟡	48.25 ⚪
	1-NNA (CD)	99.75	88.56	74.00	67.74	69.22	64.36 🟡	66.75 ⚪
Car	MMD (CD)	5.646	1.200	-	1.103	1.200	1.100 ⚪	1.000 🟡
	COV(CD)	6.530	30.84	-	48.60 🟡	31.11	36.72	47.93 ⚪
	1-NNA (CD)	99.56	93.31	-	66.74 ⚪	80.81	72.95	65.15 🟡
Chair	MMD (CD)	5.840	3.400	16.80	3.700	4.100	2.826 🟡	3.200 ⚪
	COV(CD)	17.52	48.27	42.60	48.29	50.07 ⚪	47.42	50.52 🟡
	1-NNA (CD)	97.12	64.77	58.90 ⚪	60.19	62.70	60.34	57.90 🟡

Algorithm 1 Sampling for unconditional generation (Sec. 4.1)

```
Require: Diffusion model $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$, $\alpha_t$, $\beta_t$
1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: for $t = T, \ldots, 1$ do
3:     $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\epsilon = \mathbf{0}$
4:     $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sqrt{\beta_t}\epsilon$
5: end for
6: return $\mathbf{x}_0$
```

4 Network Architectures

We summarize the parameters of diffusion model in Tab. 1. To build the encoder of the U-Net, we utilize downsampling residual blocks and it consists of various levels, each with a distinct number of channels: 64, 128, 256, 512. The process begins with an input resolution of 128 and progressively reduces to 64, 32, 16, 8 through downsampling. For the decoder, we employ upsampling residual blocks. The number of output channels for each level in the decoder is set to 512, 256, 128, 64. For activation of each layer, a SiLU (Sigmoid Linear Unit) is used after convolution layer. The input depth map has a resolution of 128×128. We use cosine noise scheduling defined as follows [8]:

$$\bar{\alpha}_t = \frac{f(t)}{f(0)}, \quad f(t) = \cos\left(\frac{t/T + s}{1 + s} \cdot \frac{\pi}{2}\right)^2 \tag{9}$$

The details of the inference process for unconditional generation and depth completion are summarized in Algorithm 1 and Algorithm 2, respectively.

5 Additional Results

We demonstrate the generated point cloud in more angles in Fig. 2, Fig. 3 and Fig. 4. More qualitative results can be found in the accompanying supplementary webpage `index.html`. We provide more quantitative results in Tab. 4 in addition to Tab. 1 in the main paper. The generated point cloud, derived from our depth maps, showcases robust 3D coherence and validates efficacy of

Algorithm 2 Sampling for conditional generation (Sec. 4.2)

```
Require: Diffusion model $\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)$, $\alpha_t$, $\beta_t$, input depth map $\mathbf{x}_0^{\text{in}}$
1: $\mathbf{x}_T^{\text{other}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: for $t = T, \ldots, 1$ do
3:     $\mathbf{x}_{t-1}^{\text{in}} \sim \mathcal{N}\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0^{\text{in}}, (1-\bar{\alpha}_t)\,\mathbf{I}\right)$
4:     $\hat{\mathbf{x}}_{t-1}^{\text{other}} \sim \mathcal{N}(\sqrt{1-\beta_t}\ \mu_\theta(\mathbf{x}_t^{r_1:r_R}, t), \beta_t\mathbf{I})$    ▷ First pass
5:     $\mathbf{x}_{t-1}^{\text{other}} \sim \mathcal{N}(\mu_\theta(\hat{\mathbf{x}}_{t-1}^{\text{other}}, \mathbf{x}_0^{in}, t), \beta_t\mathbf{I})$    ▷ Second pass
6: end for
7: return $\mathbf{x}_0^{\text{other}}$
```

Table 3: Auto-regressive generation v.s. our simultaneous generation on the ShapeNet [1] chair category.

	1NN-A	
	CD	EMD
Auto-regressive	70.23	68.13
Simultaneous	57.90	54.51

our suggested attention mechanism focusing on epipolar "line segments" and the depth fusion module. Conversely, existing point cloud-based diffusion models such as those in [6, 14, 15] are constrained by their capacity to generate only 2048 points. This limitation hampers their ability to accurately capture intricate details within 3D shapes.

Auto-regressive v.s. our simultaneous generation. We also compare with the baseline where each view of depth map is generated sequentially, i.e. the first view is generated first, the second view is conditioned on the first view and then the third view is generated conditioned on the first two views etc. As the results show in Tab. 3, our simultaneous strategy yields better results on unconditional generation task in ShapeNet [1] chair category.

We train our model on 13 classes simultaneously and report the results in Tab. 4. Since there is no significant performance drop for training on 13 classes together vs training on each class separately, this demonstrates MVDD's scalability.

6 Used codebases

Our diffusion model is based off of the github repo. The following codebases of the baselines are used:

- PVD [15]: https://github.com/alexzhou907/PVD
- LION [14]: https://github.com/nv-tlabs/LION

Table 4: Generation results on ShapeNet Car.

		Ours (each class trained separately)	Ours (trained on ShapeNet 13 classes)
	MMD (EMD)	0.620	0.640
Car	COV (EMD)	49.53	48.57
	1-NNA (EMD)	56.80	57.93

– DPM [6]: https://github.com/luost26/diffusion-point-cloud

We use other codebases for visualization and evaluation purposes:

– We use the MitSuba renderer for visualizations [9]: and the code to generate the scene discription files for MitSuba: https://github.com/zekunhao1995/PointFlowRenderer.
– We utilize SAP [10] for mesh generation with the code at https://github.com/autonomousvision/shape_as_points.
– For calculating the evaluation metrics, we use the implementation for CD at https: //github.com/ThibaultGROUEIX/ChamferDistancePytorch and for EMD at https://github.com/daerduoCarey/PyTorchEMD.

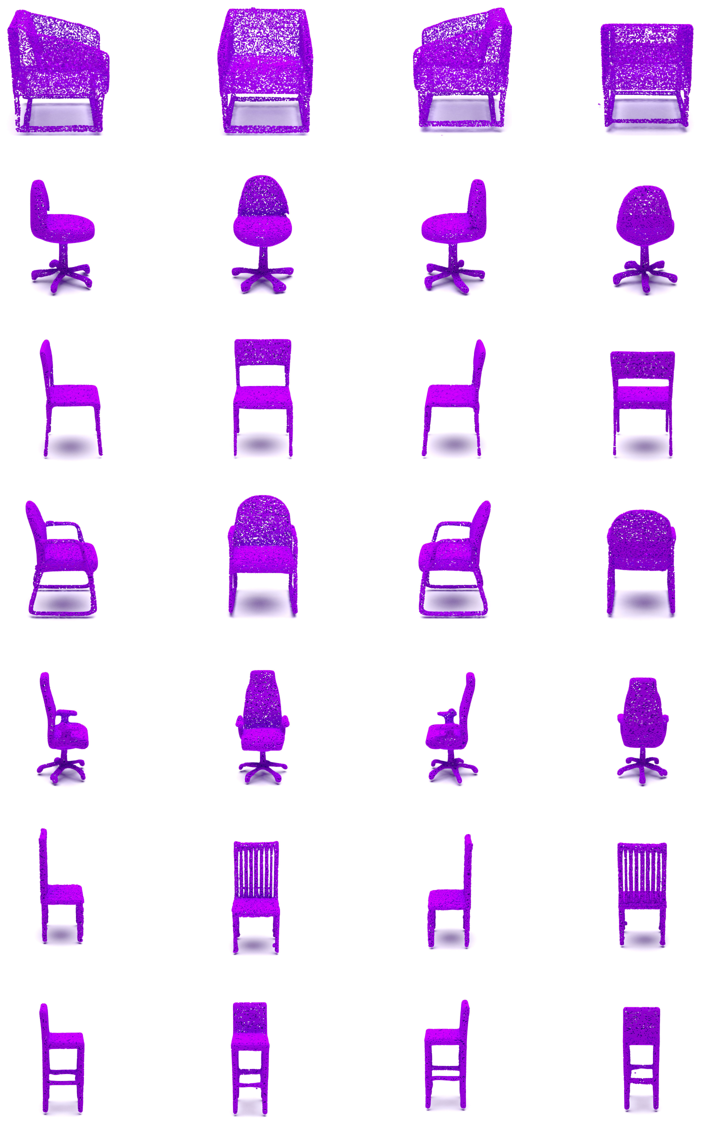


Fig. 2: Generated chairs from more angles. Please see accompanying supplemental webpage `index.html` for more results.

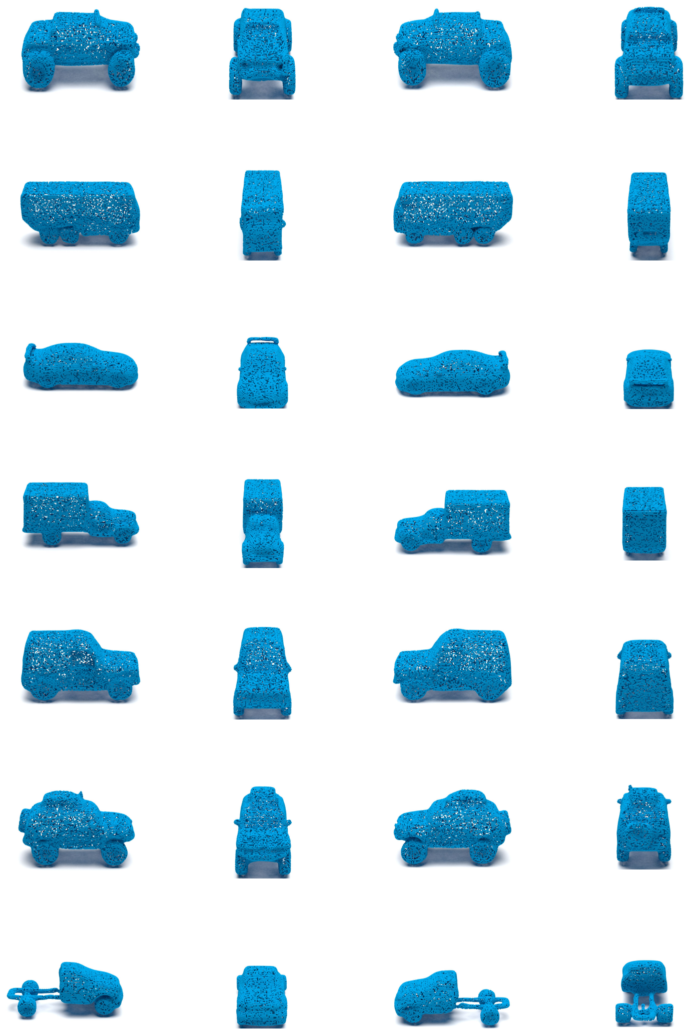


Fig. 3: Generated cars from more angles. Please see accompanying supplemental webpage `index.html` for more results.

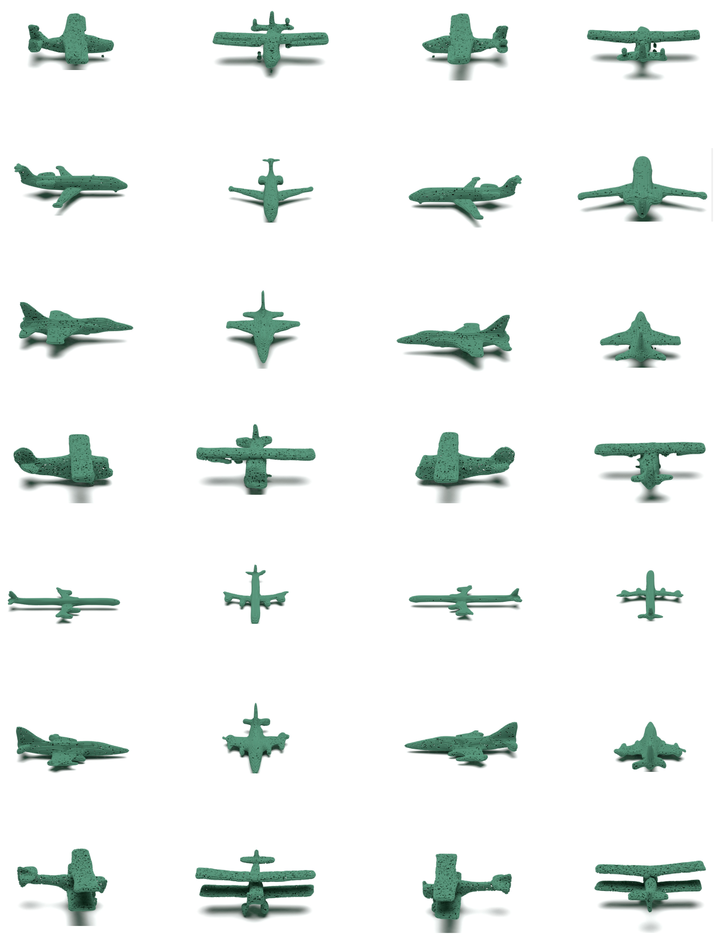


Fig. 4: Generated airplanes from more angles. Please see accompanying supplemental webpage `index.html` for more results.

References

1. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015) 6
2. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019) 5
3. Dai, J.S.: Euler–rodrigues formula variations, quaternion conjugation and intrinsic connections. Mechanism and Machine Theory **92**, 144–152 (2015) 4
4. Dhariwal, P., Nichol, A.: Diffusion models beat gans on image synthesis. Advances in neural information processing systems **34**, 8780–8794 (2021) 3
5. Lopez-Paz, D., Oquab, M.: Revisiting classifier two-sample tests. arXiv preprint arXiv:1610.06545 (2016) 2
6. Luo, S., Hu, W.: Diffusion probabilistic models for 3d point cloud generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2837–2845 (2021) 5, 6, 7
7. Nam, G., Khlifi, M., Rodriguez, A., Tono, A., Zhou, L., Guerrero, P.: 3d-ldm: Neural implicit 3d shape generation with latent diffusion models. arXiv preprint arXiv:2212.00842 (2022) 5
8. Nichol, A.Q., Dhariwal, P.: Improved denoising diffusion probabilistic models. In: International Conference on Machine Learning. pp. 8162–8171. PMLR (2021) 5
9. Nimier-David, M., Vicini, D., Zeltner, T., Jakob, W.: Mitsuba 2: A retargetable forward and inverse renderer. ACM Transactions on Graphics (TOG) **38**(6), 1–17 (2019) 7
10. Peng, S., Jiang, C., Liao, Y., Niemeyer, M., Pollefeys, M., Geiger, A.: Shape as points: A differentiable poisson solver. Advances in Neural Information Processing Systems **34**, 13032–13044 (2021) 7
11. Shi, Y., Wang, P., Ye, J., Long, M., Li, K., Yang, X.: Mvdream: Multi-view diffusion for 3d generation. arXiv preprint arXiv:2308.16512 (2023) 3
12. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017) 3
13. Xu, Q., Huang, G., Yuan, Y., Guo, C., Sun, Y., Wu, F., Weinberger, K.: An empirical study on evaluation metrics of generative adversarial networks. arXiv preprint arXiv:1806.07755 (2018) 2
14. Zeng, X., Vahdat, A., Williams, F., Gojcic, Z., Litany, O., Fidler, S., Kreis, K.: Lion: Latent point diffusion models for 3d shape generation. arXiv preprint arXiv:2210.06978 (2022) 5, 6
15. Zhou, L., Du, Y., Wu, J.: 3d shape generation and completion through point-voxel diffusion. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5826–5835 (2021) 5, 6