A ADDITIONAL RESULTS

- More text-to-video generation result comparisons: `https://t1-diffusion-model.github.io/visual-comparisons-celebvt.html`
- More non-face text-to-video generation results: `https://t1-diffusion-model.github.io/visual-comparisons-webvid.html`
- More 3D viewpoints generation results: `https://t1-diffusion-model.github.io/visual-comparisons-shapenet_3dviewpoints.html`
- More 3D voxel[1] generation result comparisons: `https://t1-diffusion-model.github.io/visual-comparisons-shapenet_voxels.html`

B ADDITIONAL SETTINGS

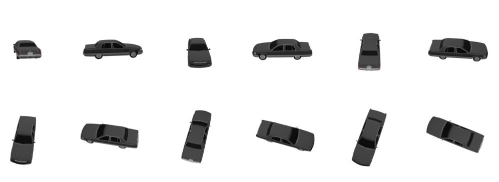
Model Details.

- In the interest of maintaining simplicity, we adhere to the methodology outlined by Dhariwal et al. (Dhariwal & Nichol, 2021) and utilize a 256-dimensional frequency embedding to encapsulate input denoising timesteps. This embedding is then refined through a two-layer Multilayer Perceptron (MLP) with Swish (SiLU) activation functions.
- Our model aligns with the size configuration of DiT-XL (Peebles & Xie, 2022b), which includes retaining the number of transformer blocks (*i.e.* 28), the hidden dimension size of each transformer block (*i.e.*, 1152), and the number of attention heads (*i.e.*, 16).
- Our model derives text embeddings employing T5-XXL (Raffel et al., 2020), culminating in a fixed length token sequence (*i.e.*, 256) which matches the length of the noisy tokens. To further process each text embedding token, our model compresses them via a single layer MLP, which has a hidden dimension size identical to that of the transformer block.

Diffusion Process Details. Our model uses classifier-free guidance in the backward process with a fixed scale of 8.5. To keep consistency with DiT-XL (Peebles & Xie, 2022a), we only applied guidance to the first three channels of each denoised token.

3D Geometry Rendering Settings. Following the settings of pixelNeRF (Yu et al., 2021), we render each car voxel into 128 random views for training models and testing. However, the original setting puts camera far away from the objects and hence results in two many blank areas in the rendered views. We empirically find that these blank areas hurts the diffusion model performance since the noise becomes obvious in blank area and can be easily inferred by diffusion models, which degrades the distribution modeling capability of diffusion models.

To overcome this, we first randomly fill the blank area with Gaussian noise $\mathcal{N}(0, 0.1)$ without overlapping the 3D geometry. We then move the camera in the z-axis from 4.0 into 3.0, which is closer to the object than the previous one. During testing, we use the same settings as pixelNeRF and remove the noise according to the mask. For straightforward understand their difference, we visualized their rendered results in Fig. 7.


(a) pixelNeRF Yu et al. (2021) rendering

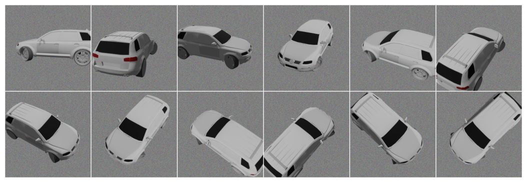

(b) ours rendering

Figure 7: Visual comparisons of different 3D geometry rendering settings.

[1]Note that the comparisons are conducted by rendering their generated voxel objects into 3D viewpoints and then compare those results with ours.

Hyper-parameter	CIFAR10 (Krizhevsky et al., 2009)	CelebV-Text (Yu et al., 2023b)	ShapeNet (Chang et al., 2015)
train res.	64×64	256×256×128	128×128×128 256×256×128 (upsampled)
eval res.	64×64	256×256×128	128×128×251 256×256×251
# dim coordinates	2	3	6
# dim signal	3	3	3
# freq pos. embed	10	10	10
# freq pos. embed t	64	64	64
#blocks	28	28	28
#block latents	1152	1152	1152
#self attention heads	16	16	16
batch size	128	128	128
lr	$1e-4$	$1e-4$	$1e-4$
epochs	400	400	1200

Table 3: Hyperparameters and settings on different datasets.

C Additional Dataset Details

In the subsequent sections, we present the datasets utilized for conducting our experiments. We empirically change the size settings of our model as shown in Tab 3.

- **CelebV-Text** (Yu et al., 2023b) Due to the unavailability of some videos in the released dataset, we utilize the first 60,000 downloadable videos for training our model. For videos that contain more than 128 frames, we uniformly select 128 frames. Conversely, for videos with fewer than 128 frames, we move to the next video, following the order of their names, until we identify a video that meets the required length of 128 frames.
- **ShapeNet (Chang et al., 2015).** The conventional methods in DPF (Zhuang et al., 2023) and GEM (Du et al., 2021) generally involve training models on the ShapeNet dataset, wherein each object is depicted as a voxel grid at a resolution of 64^3. However, our model distinguishes itself by relying on view-level pairs, thereby adopting strategies utilized by innovative view synthesis methods like pixelNeRF (Yu et al., 2021) and GeNVS (Chan et al., 2023). To specify, we conduct training on the car classes of ShapeNet, which encompasses 2,458 cars, each demonstrated with 128 renderings randomly scattered across the surface of a sphere.

  Moreover, it's worth noting that our model refrains from directly leveraging the text descriptions of the car images. Instead, it conditions on the CLIP embedding (Radford et al., 2021) of car images for linguistic guidance. This approach circumvents the potential accumulation of errors that might occur during the text-to-image transformation process.

D Additional Experimental Details

Video Generation Metrics Settings. In video generation, we use FVD (Unterthiner et al., 2018)[2] to evaluate the video spatial-temporal coherency, FID (Heusel et al., 2017)[3] to evaluate the frame quality, and CLIPSIM (Radford et al., 2021)[4] to evaluate relevance between the generated video and input text. As all metrics are sensitive to data scale during testing, we randomly select 2,048 videos from the test data and generate results as the "real" and "fake" part in our metric experiments. For FID, we uniformly sample 4 frames from each video and use a total of 8,192 images. For CLIPSIM, we calculate the average score across all frames. We use the "openai/clip-vit-large-patch14" model for extracting features in CLIPSIM calculation.

[2]FVD is implemented in `https://github.com/sihyun-yu/DIGAN`
[3]FID is implemented in `https://github.com/toshas/torch-fidelity`
[4]CLIPSIM is implemented in `https://github.com/Lightning-AI/torchmetrics`

E VISUALIZATION

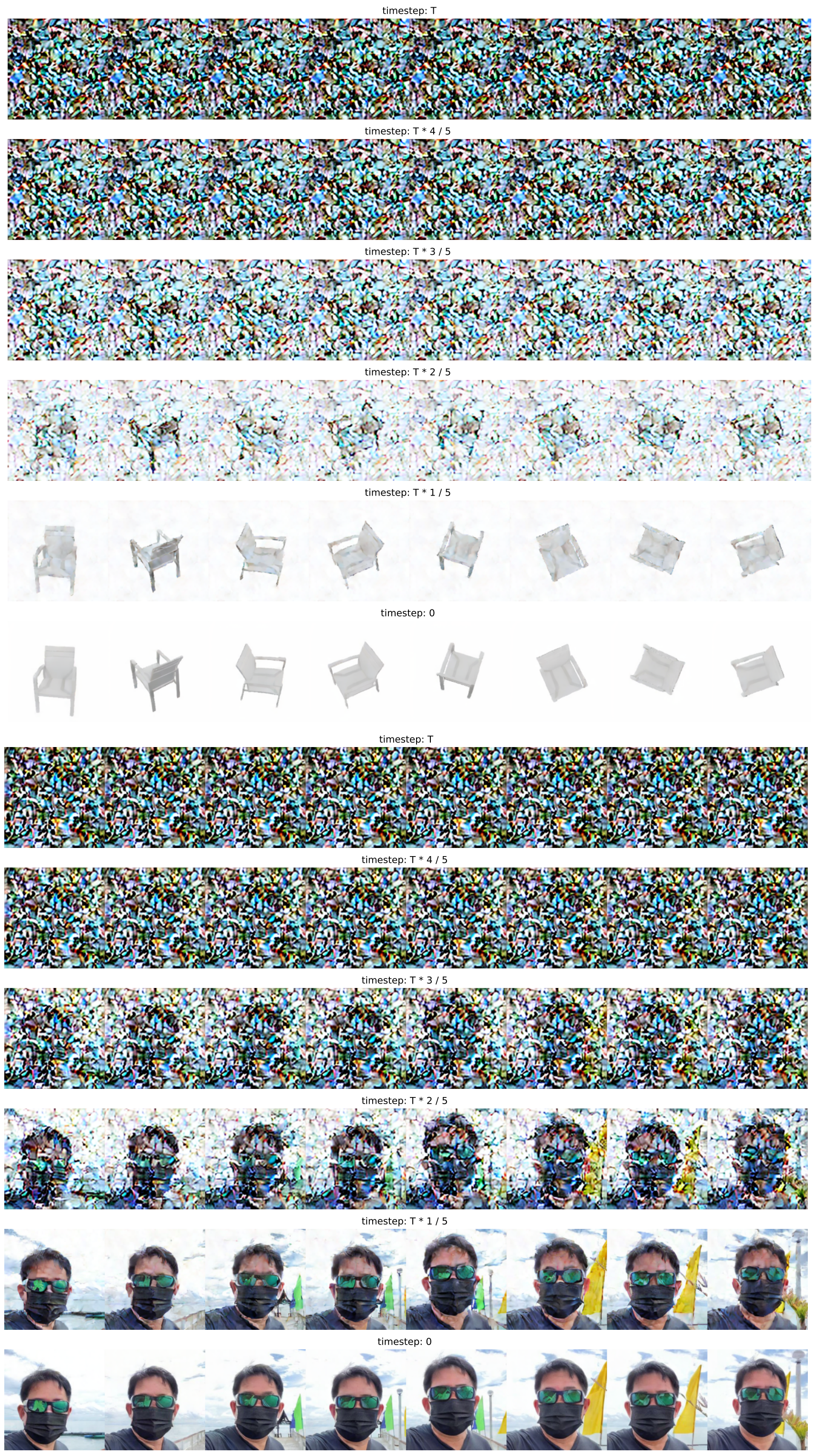


Figure 8: Intermediate results visualization of our multi-view generation. Here we show the predicted clean image at different timesteps for highlighting the cross-view consistency of our model.