SC3D: Self-conditioned Generative Gaussian Model with 3D-aware Feedback

Anonymous Author(s)
Affiliation
Address
email

Abstract

Existing single image-to-3D creation methods typically involve a two-stage process, first generating multi-view images, and then using these images for 3D reconstruction. However, training these two stages separately leads to significant data bias in the inference phase, thus affecting the quality of reconstructed results. We introduce a unified 3D generation framework, named SC3D, which integrates diffusion-based multi-view image generation and 3D reconstruction through a self-conditioning mechanism. In our framework, these two modules are established as a cyclic relationship so that they adapt to the distribution of each other. During the denoising process of multi-view generation, we feed rendered color images and maps by SC3D itself to the multi-view generation module. This self-conditioned method with 3D aware feedback unites the entire process and improves geometric consistency. Experiments show that our approach enhances sampling quality, and improves the efficiency and output quality of the generation process.

1 Introduction

3D content creation from a single image have improved rapidly in recent years with the adoption of large 3D datasets [1, 2, 3] and diffusion models [4, 5, 6]. A body of research [7, 8, 9, 10, 11, 12, 13, 14] has focused on multi-view diffusion models, fine-tuning pretrained image or video diffusion models on 3D datasets to enable consistent multi-view synthesis. These methods demonstrate generalizability and produce promising results. Another group of works [15, 16, 17, 18, 19] propose generalizable reconstruction models, generating 3D representation from one or few views in a feed-forward process. Theses reconstruction models built upon convolutional network or transformer backbone, have led to efficient image-to-3D creation.

Since single-view reconstruction models [15] trained on 3D datasets [1, 20] lack generalizability and often produce blurring at unseen viewpoints, several works [21, 16, 18, 19] extend models to sparse-view input, boosting the reconstruction quality. As shown in Fig. 1, these methods split 3D generation into two stages: multi-view synthesis and 3D reconstruction. By combining generalizable multi-view diffusion models and robust sparse-view reconstruction models, such pipelines achieve high-quality image to 3D generation. However, combining the two independently designed models introduces a significant "data bias" to the reconstruction model. The data bias is mainly reflected in two aspects: **(1) Multi-view bias.** Multi-view diffusion models learn consistency at the image level, struggle to ensure geometric consistency. When it comes to reconstruction, multi-view images that lack geometric consistency affect the subsequent stage. **(2) Limited data for reconstruction model.** Unlike multi-view diffusion models, reconstruction models which are trained from scratch on limited 3D dataset, lacks the generalization ability.

Recent works like IM-3D [22] and VideoMV [23] have attempted to aggregate the rendered views of the reconstructed 3D model into previous-step multi-view synthesis, thus improving the capability

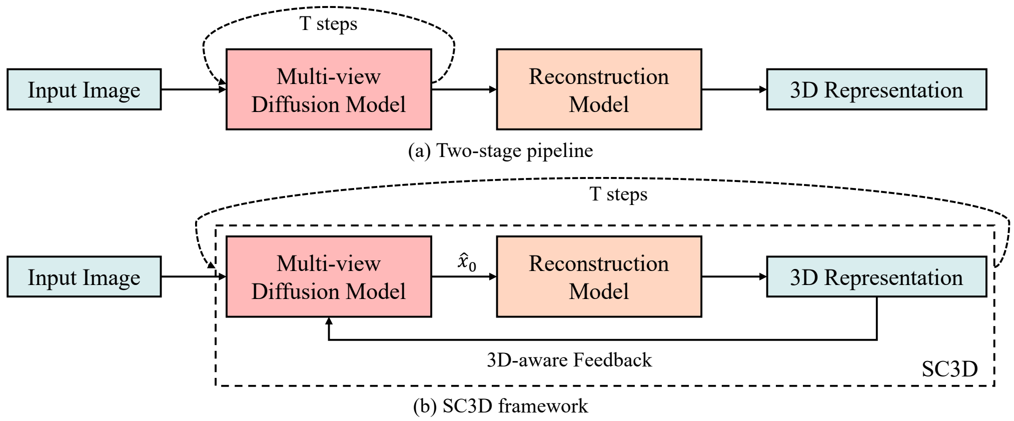

Figure 1: **Concept comparison** between SC3D and previous two-stage methods. Instead of directly combining multi-view diffusion model and reconstruction model, our self-conditioned framework involves joint training of these two models and establish them as a cyclic association. During the denoising process, rendered 3D-aware maps are fed to the multi-view generation module.

and consistency of the generated multi-view images. These methods integrate the aforementioned two stages at the inference phase. But the models at both stages still lack joint training, which prevents the reconstruction model from enhancing its robustness to the generated poor multiviews. Moreover, these test-time aggregating methods cannot directly utilize geometric information such as depth maps, normal maps, or position maps that can also be obtained from the reconstructed 3D. Notably, these explicit 3D aware maps can better guide the multi-view generation.

To address these challenges, we propose a unified single image-to-3D creation framework, named SC3D, which integrates multi-view generation and 3D reconstruction through a self-conditioning mechanism. Our framework involves jointly training the multi-view diffusion model and the reconstruction model. In SC3D, these two modules are established as a cyclic relationship so that they adapt to the characteristics of each other, enabling robust generation at inference. Specifically, during the denoising process, we feed rendered 3D-aware maps from the reconstructed 3D to the multi-view generation module. By leveraging the color maps and spatial canonical coordinates maps from the reconstruction 3D representation as condition, our multi-view diffusion model synthesizes multi-view images that better conform to the actual 3D structure. This self-conditioned framework with 3D aware feedback unites the 3D generation process and enhances the robustness for unseen complex scenes. Experiments on the GSO dataset [24] validate that our SC3D reduces data bias between training and inference, and enhances the overall efficiency and output quality.

Our key contributions are as follows:

- We introduce SC3D, which unifies multi-view generation and 3D reconstruction in a single framework and involves jointly training these two modules, enabling adaption to each other.
- SC3D employs a self-conditioning mechanism with 3D-aware feedback, using rendered 3D-aware maps to guide the multi-view generation, ensuring better geometric consistency and robustness.
- Experiments show that SC3D significantly reduces data bias, improves the quality of 3D reconstruction, and enhances overall efficiency in creating 3D content from a single image.

2 Related Work

Image/Video Diffusion for Multi-view Generation Diffusion models [25, 26, 27, 28, 29, 30, 31, 32, 33, 34] have demonstrated their powerful generative capabilities in image and video generation fields. Current research [7, 8, 9, 10, 11, 12, 13, 14, 35] fine-tunes pretrained image/video diffusion models on 3D datasets like Objaverse [1] and MVImageNet [20]. Zero123 [7] introduces relative view condition to image diffusion models, enabling novel view synthesis from a single image and preserving generalizability. Based on it, methods like SyncDreamer [9], ConsistNet [36] and EpiDiff [11] design attention modules to generate consistent multi-view images. These methods finetuned from image diffusion models produce generally promising results. By considering multi-view images as consecutive frames of a video (e.g., orbiting camera views), it naturally leads to the idea of applying video generation models to 3D generation [13]. However, since the diffusion model is not explicitly modeled in 3D space, the generated multi-view images often struggle to achieve consistent and robust details.

Image to 3D Reconstruction Recently, the task of reconstructing 3D objects has evolved from traditional multi-view reconstruction methods [37, 38, 39, 40] to feed-forward reconstruction models [15, 41, 42, 16, 17, 18, 19]. Ultilizing one or few shot as input, these highly generalizable reconstruction models synthesize 3D representation, enabling the rapid generation of 3D objects. LRM [15] proposes a transformer-based model to effectively map image tokens to 3D triplanes. Instant3D [21] further extends LRM to sparse-view input, significantly boosting the reconstruction quality. LGM [16] and GRM [17] replace the triplane representation with 3D Gaussians [40] to enjoy its superior rendering efficiency. CRM [18] and InstantMesh [19] optimize on the mesh representation for high-quality geometry and texture modeling. These reconstrucion models built upon convolutional network architecture or transformer backbone, have led to efficient image-to-3D creation.

Pipelines of 3D Generation Early works propose to distill knowledge of image prior to create 3D models via Score Distillation Sampling (SDS) [43, 44, 45], limited by the low speed of per-scene optimization. Several works [9, 11, 14, 22] fine-tune image diffusion models to generate multi-view images, which are then utilized for 3D shape and appearance recovery with traditional reconstruction methods [46, 40]. More recently, several works [21, 16, 18, 19, 23] involve both multi-view diffusion models and feed-forward reconstruction models in the generation process. Such pipelines attempt to combine the processes into a cohesive two-stage approach, thus achieving highly generalizable and high-quality single-image to 3D generation. However, due to the lack of explicit 3D modeling, the results generated by the multi-view diffusion model cannot guarantee strong consistency, which will lead to data deviation for the reconstructed model between the testing phase and the training phase. Compared to them, we propose a unified pipeline, integrating the two stages through a self-conditioning mechanism at the training stage, with 3D aware feedback for high consistency.

3 Method

Given a single image, SC3D aims to generate multiview-consistent images with a reconstructed 3D Gaussion model. To reduce the data bias and improve robustness of the generation, we propose SC3D, a unified 3D generation framework which integrates multi-view synthesis and 3D reconstruction through a self-conditioning mechanism. As illustrated in Fig. 2, the proposed framework involves a video diffusion model (SVD [32]) as multi-view generator (refer to Section 3.1) and a feed-forward reconstruction model to recover a 3D Gaussian Splatting (refer to Section 3.2. Moreover, we introduce a self-conditioning mechanism, feeding the 3D-aware information obtained from the reconstruction module back to the multi-view generation process (refer to Section 3.3). The 3D-aware denoising sampling strategy iteratively refines the multi-view images and the 3d model, thus enhancing the final production.

3.1 Video Diffusion Model as Multiview Generator

Recent video diffusion models such as those in [13, 34] have demonstrated a remarkable capability to generate 3D-aware videos by scaling up both the model and dataset. Our research employs the well-known Stable Video Diffusion (SVD) Model, which generates videos from image input. Formally, given an image $I \in \mathbb{R}^{3\times h\times w}$, the model is designed to generate a video $V \in \mathbb{R}^{f\times 3\times h\times w}$. Further details about SVD can be found in Appendix A.1.

We enhance the video diffusion model with camera control c to generate images from different viewpoints. Traditional methods encode camera positions at the frame level, which results in all pixels within one view sharing the same positional encoding [47, 13]. Building on the innovations of previous work [11, 35], we integrate the camera condition c into the denoising network by parameterizing the rays $\mathbf{r} = (o, o \times d)$. Specifically, we use two-layered MLP to inject Plücker ray embeddings for each latent pixel, enabling precise positional encoding at the pixel level. This approach allows for more detailed and accurate 3D rendering, as pixel-specific embedding enhances the model's ability to handle complex variations in depth and perspective across the video frames.

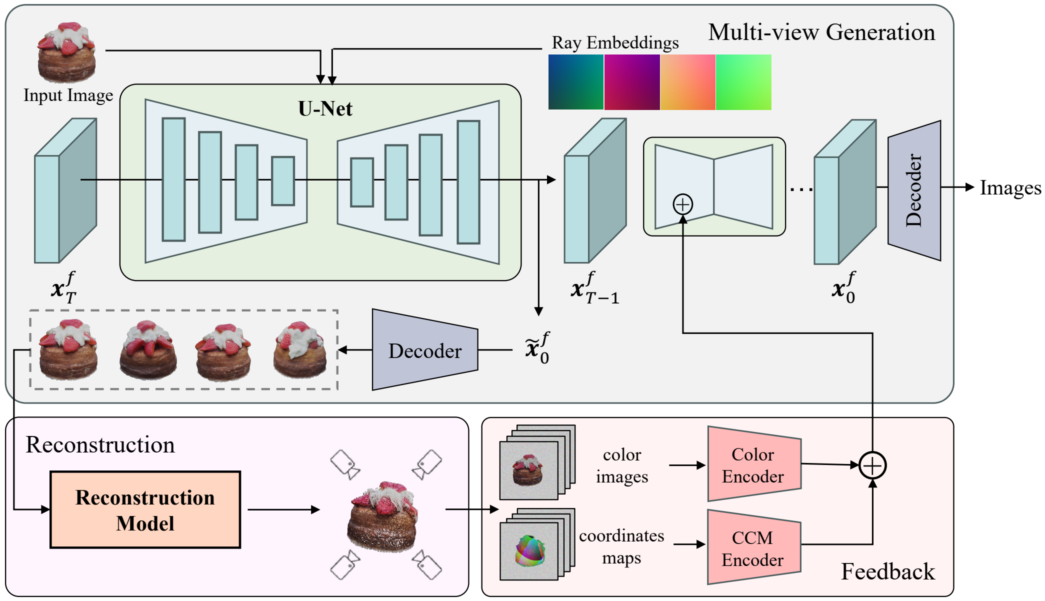

Figure 2: **Overview of SC3D.** We adopt a video diffusion model as the multi-view generator by incorporating the input image and relative camera poses. In the denoising sampling loop, we decode the predicted $\widetilde{\mathbf{x}}_0^f$ to noise-corrupted images, which are then used to recover 3D representation by a feed-forward reconstruction model. Then the rendered color images and coordinates maps are encoded and fed into the next denoising step. At inference, the 3D-aware denoising sampling strategy iteratively refines the images by incorporating feedback from the reconstructed 3D into the denoising loop, enhancing multi-view consistency and image quality.

In our framework, unlike existing two-stage methods, our multi-view diffusion model does not complete multiple denoising steps independently. In contrast, in the denoising sampling loop, we obtain the straightly predicted $\widetilde{\mathbf{x}}_0^f$ at the current timestep, which will be used for subsequent 3D reconstruction. Then we use rendered 3d-aware view maps as conditions to guide the next denoising step. Therefore, at each sampling step, we do the reparameterization of the output from the denoising network F_θ to convert it into $\widetilde{\mathbf{x}}_0^f$. Taking a single view as an example, we processes the denoised image $c_{\text{in}}(\sigma)\mathbf{x}$ and the associated noise level $c_{\text{noise}}(\sigma)$, which σ indicates the standard deviation of the noise. The reparameterization is formulated as follows:

$$\tilde{\mathbf{x}}_0 = c_{\text{skip}}(\sigma)\mathbf{x} + c_{\text{out}}(\sigma)F_\theta(c_{\text{in}}(\sigma)\mathbf{x}; c_{\text{noise}}(\sigma)). \tag{1}$$

The above operation process adjusts the output of F_θ to $\widetilde{\mathbf{x}}_0^f$, which will be decoded into images and passed to the subsequent 3D reconstruction module.

3.2 Feed-Forward Reconstruction Model

In the SC3D framework, the feed-forward reconstruction model is designed to recover 3D models from pre-generated multi-view images, which can be images decoded from straightly predicted $\widetilde{\mathbf{x}}_0^f$, or completely denoised images. We utilize Large Multi-View Gaussian Model (LGM) [16] $\mathcal{G}$ as our reconstruction module due to its real-time rendering capabilities that benefit from 3D representation of Gaussian Splatting. This method integrates seamlessly with our jointly training framework, allowing for quick adaptation and efficient processing.

We pass four specific views from the reparameterized output $\widetilde{\mathbf{x}}_0^f$ to the Large Gaussian Model (LGM) for 3D Gaussian Splatting reconstruction. To enhance the performance of LGM, particularly its sensitivity to different noise levels $c_{\text{noise}}(\sigma)$ and image details, we introduce a zero-initialized time embedding layer within the original U-Net structure of the LGM. This innovative modification enables the LGM to dynamically adapt to the diverse outputs that arise at different stages of the denoising process, thereby substantially improving its capacity to accurately reconstruct 3D content from images that have undergone partial denoising.

The loss function employed for the fine-tuning of the LGM is articulated as follows:

$$\mathcal{L}_{\mathcal{G}} = \mathcal{L}_{\text{rgb}}(\mathbf{x}_0, \mathcal{G}(\tilde{\mathbf{x}}_0, c_{\text{noise}}(\sigma))) + \lambda \mathcal{L}_{\text{LPIPS}}(\mathbf{x}_0, \mathcal{G}(\tilde{\mathbf{x}}_0, c_{\text{noise}}(\sigma))). \tag{2}$$

where we have utilized the mean square error loss $\mathcal{L}_{\text{rgb}}$ for the color channel and a VGG-based perceptual loss $\mathcal{L}_{\text{LPIPS}}$[43] for the LPIPS term. In practical applications, the weighting factor λ is conventionally set to 1.

Additionally, to maintain the model's reconstruction capability for normal images, we also input the model without adding noise and calculate the corresponding loss. In this case, we set $c_{\text{noise}}(\sigma)$ to 0.

3.3 3D-Aware Feedback Mechanism

As shown in Fig. 2, we adopt a 3D-aware feedback mechanism that involves the rendered color images and geometric maps produced by our reconstruction module in a denoising loop to further improve the multi-view consistency of the resulting images and facilitate cyclic adaptation of the two stages. Instead of integrating multi-view generation and 3D reconstruction at the inference stage using re-sampling strategy [22, 23], we propose to train these two modules jointly to support more informative feedback. Specifically, in addition to the rendered color images, our flexible framework is able to derive additional geometric features to guide the generation process, which brings guidance of more explicit 3D information to multi-view generation.

In practice, we obtain color images and canonical coordinates maps [48] from the reconstructed 3D model, and utilize them as condition to guide the next denoising step of multi-view generation. We use position maps instead of depth maps or normal maps as the representative of geometric maps because canonical coordinate maps record the vertex coordinate values after normalization of the overall 3D model, rather than the normalization of the relative self-view (such as depth maps). This operation enables the rendered maps to be characterized as cross-view alignment, providing the strong guidance of more explicit cross-view geometry relationship. The details of canonical coordinates map can be found in Appendix A.2.

We adopt a 3D-aware self-conditioning [49] training and inference strategy that leverages reconstruction stage results to enhance multi-view consistency and the quality of generated images. During training, the original denoising network $F_\theta(\mathbf{x}; \sigma)$ is augmented with a 3D-aware feedback denoising network $F_\theta(\mathcal{G}(\tilde{\mathbf{x}}_0); \sigma)$, where $\mathcal{G}(\tilde{\mathbf{x}}_0)$ is the output of the LGM reconstruction.

To encode color images and coordinates maps into the denoising network of multi-view generation module, we design two simple and lightweight encoders for color images and coordinates maps using a series of convolutional neural networks, like T2I-Adapter [50]. The encoders are composed of four feature extraction blocks and three downsample blocks to change the feature resolution, so that the dimension of the encoded features is the same as the intermediate feature in the encoder of U-Net denoiser. The extracted features from the two conditional modalities are then added to the U-Net encoder at each scale.

Training Strategy As illustrated in Algorithm 1, to train a 3D-aware multi-view generation network, we use the rendered maps by the 3D reconstruction module as the self-conditioning input. In practice, we randomly use this self-conditioning mechanism with a probability of 0.5. When not using the 3D reconstruction result, we set $\mathcal{G}(\tilde{x}_0) = 0$ as the input. This probabilistic approach ensures balanced learning, allowing the model to effectively incorporate 3D information without over-reliance on it.

Algorithm 1 Training SC3D with the self-conditioned strategy.

```
def train_loss(x, cond_image):
    """Returns the loss on a training example x."""
    # Sample sigma from a log-normal distribution
    sigma = log_normal(P_mean, P_std)

    # Reparameterize sigma to obtain conditioning parameters
    c_in, c_out, c_skip, c_noise, lambda_param = reparameterizing(sigma)

    # Add noise to input data
    noise_x = x + sigma * normal(mean=0, std=1)
    input_x = c_in * noise_x

    # Initial prediction without self-conditioning
    self_cond = None
    F_pred = net(input_x, c_noise, cond_image, self_cond)
    pred_x = c_out * F_pred + c_skip * noise_x

    # Update self_cond using the reconstruction model
    self_cond = recon_model(pred_x, c_noise)

    # Use rendered maps as condition and denoise
    if self_cond and np.random.uniform(0, 1) > 0.5:
        F_pred = net(input_x, t, cond_image, self_cond.detach())
        pred_x = c_out * F_pred + c_skip * noise_x

    # Compute loss
    loss = lambda_param * (pred_x - target) ** 2
    recon_loss = recon_loss_fn(self_cond, x)

    return loss.mean() + recon_loss
```

Inference/sampling strategy At the inference stage, as shown in Algorithm 2, the 3D feedback $\mathcal{G}(\tilde{\mathbf{x}}_0)$ is initially set to 0. At each timestep, this feedback is updated with the previous reconstruction result $\mathcal{G}(\tilde{\mathbf{x}}_0)$. This iterative process refines the 3D representation, ensuring each frame benefits from prior reconstructions, leading to higher quality and more consistent 3D-aware images.

Algorithm 2 Sampling algorithm of SC3D.

```
def generate(sigmas, cond_image):
    self_cond = None
    x_T = normal(mean=0, std=1)  # Initialize latent variable with Gaussian noise
    for sigma in sigmas:
        # Reparameterize sigma to obtain conditioning parameters
        c_in, c_out, c_skip, c_noise, lambda_param = reparameterizing(sigma)

        # Add noise to the latent variable
        noise_x = x_T + sigma * normal(mean=0, std=1)
        input_x = c_in * noise_x

        # Generate prediction
        F_pred = net(input_x, t, cond_image, self_cond)
        pred_x = c_out * F_pred + c_skip * noise_x

        # Update self_cond using the reconstruction model
        self_cond = recon_model(pred_x, c_noise)

    return pred_x
```

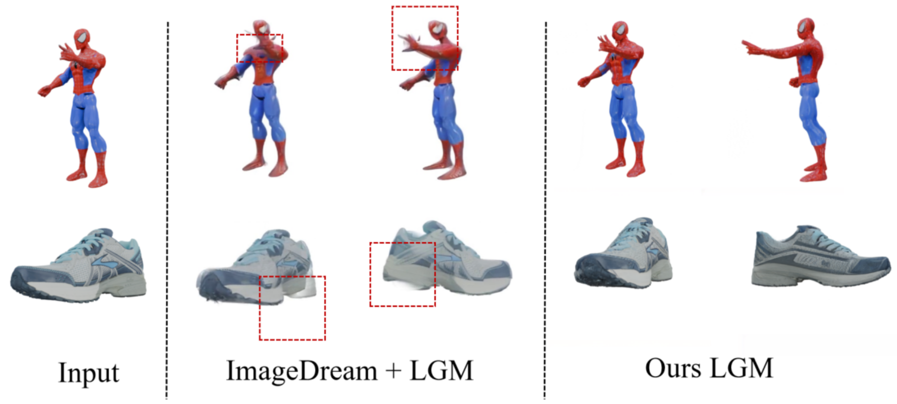

Figure 3: Qualitative comparison with ImageDream-LGM and Our LGM.

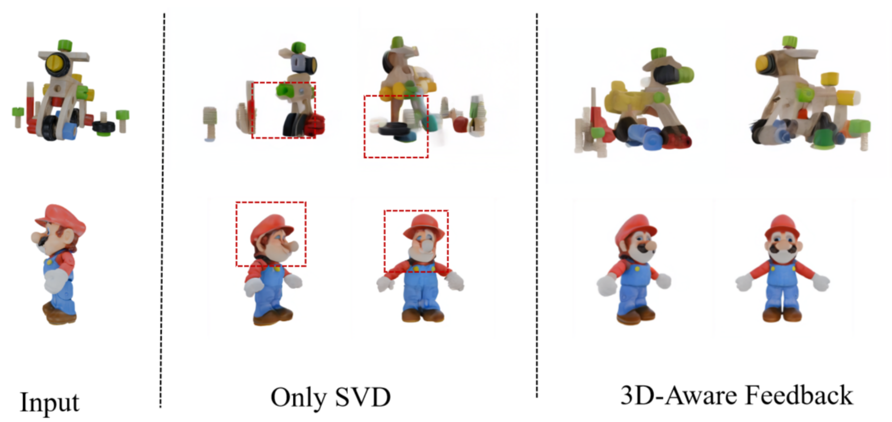

Figure 4: Qualitative comparison with no-feedback and 3d-aware feedback.

4 Experiments

We focus on 3D asset content synthesis, training our model on the G-Objaverse [1, 51] dataset and the LVIS subset of Objaverse, which consists of 300K high-quality 3D objects and is widely used in 3D generation. We evaluate SC3D on the Google Scanned Object (GSO) dataset [24], which consists of approximately 1,000 scanned models, and we randomly select 100 samples for comparison. We adopt TripoSR[42], SyncDreamer[9], SV3D[13], ImageDream [8] combined with LGM [16] as the baseline approach [16] and VideoMV[23] as baseline methods. For each baseline, we report PSNR, SSIM, and LPIPS metrics.

4.1 Comparison results

For LGM, we utilize the official LGM single-image generation pipeline, which employs ImageDream [52] to transition from a single image input to multiple images. However, the conical coordinate system employed by ImageDream complicates the direct evaluation of the output. To address this, we use the official code to test on the GSO dataset, followed by manual calibration to assess the generated quality, as illustrated in Fig. 3. The misalignment between the two stages of ImageDream and LGM often results in generated models with blurred linear edges and geometric ambiguities. Nonetheless, our LGM, enhanced by a feedback mechanism, demonstrates significantly improved geometric and texture quality, producing results that closely approximate reality.

As illustrate in 6, We find that although it can generate very continuous frames, the generated content tends to deviate from the given input image. This results in sub-optimal performance in

Method	Resolution	PSNR↑	SSIM↑	LPIPS↓
TripoSR	256×256	18.481	0.8506	0.1357
SyncDreamer	256×256	20.056	0.8163	0.1596
SV3D	576×576	21.042	0.8497	0.1296
VideoMV(SD)	256×256	17.459	0.806	0.1446
VideoMV(GS)	256×256	17.577	0.807	0.1454
SC3D (SVD)	512×512	21.625	0.9045	0.1011
SC3D (GS)	512×512	**21.761**	**0.9094**	**0.0991**

Table 1: Comparison of performance metrics across different models and configurations.

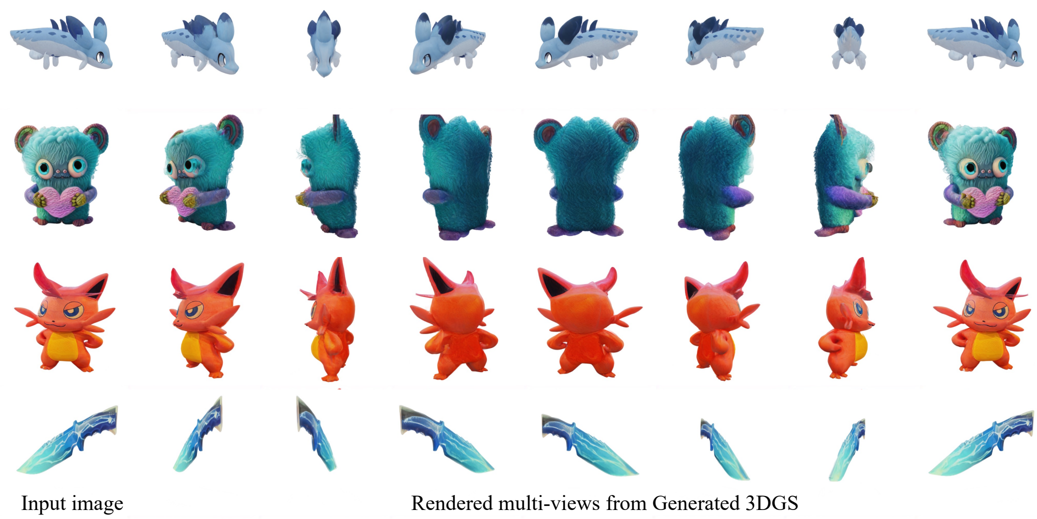

Figure 5: Out of distribution testing results.

the reconstruction metric. Additionally, VideoMV training the LGM separately with noisy images deteriorates, resulting in a visually noticeable reduction in its ability to generate texture details.

4.2 Ablation study

To validate the effectiveness of the proposed SC-3D framework, we conducted a series of ablation studies comparing PSNR, SSIM, and LPIPS metrics for different configurations (Table 2). We start with the base video diffusion model we trained, We then introduced 3D coordinates map feedback and RGB texture feedback from the reconstruction model to the diffusion model, which improved geometric consistency and texture detail across views. Combining both feedback mechanisms in the SVD + 3D-aware Feedback configuration resulted in the best performance, demonstrating significant improvements in the final 3D reconstruction quality by enhancing both geometric consistency and texture detail preservation.

Method	Variant	PSNR ↑	SSIM ↑	LPIPS ↓
SVD	SVD	20.038	0.8745	0.1253
	GS	20.549	0.8651	0.1183
SVD + Coordinates Map Feedback	SVD	21.021	0.8973	0.1110
	GS	21.325	0.8937	0.1092
SVD + 3D-aware Feedback	SVD	21.752	**0.9122**	0.0993
	GS	**21.761**	0.9094	**0.0991**

Table 2: Performance metrics of different feedback mechanisms.

Figure 6: The Generation Example of VideoMV

We also demonstrate the impact of incorporating feedback mechanisms on the two models, as shown in Table 3. It can be observed that when no feedback mechanism is used, there is a significant discrepancy between the two models' modalities, which leads to a degradation in their combined performance.

Method	Delta PSNR	Delta SSIM	Delta LPIPS
SVD	0.511	0.0094	0.0070
SVD + Coordinates Map Feedback	0.304	0.0036	0.0018
SVD + 3D-aware Feedback	0.009	0.0028	0.0002

Table 3: The absolute differences in performance metrics between GS and SVD generation results..

4.3 Limitations

Current models utilize Gaussian splatting as a 3D representation, mapping and rendering coordinates to textures for feedback. Although algorithms for converting Gaussian Splatting to meshe are under development, achieving high quality in converting Gaussian models to general meshes remains challenging. Directly employing a NeRF-based feed-forward model during the training process significantly reduces training speed due to the computational demands of volumetric rendering. Our model currently lacks the ability to generalize to the scene level, a limitation we intend to address in future research.

5 Conclusion

In this paper, we introduce SC3D, a unified framework for 3D generation from a single image that integrates multi-view image generation and 3D reconstruction through a self-conditioning mechanism. By establishing a cyclic relationship between these two stages, our approach effectively mitigates the data bias encountered in traditional methods. The self-conditioned method with 3D-aware feedback enhances geometric consistency throughout the generation process.

Our experiments demonstrate that SC3D not only improves the quality and efficiency of the generation process but also achieves superior geometric consistency and detail in the reconstructed 3D models. By jointly training the multi-view diffusion model and the reconstruction model, SC3D adapts to the inherent biases of each stage, resulting in more robust and accurate outputs.

References

[1] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023.

[2] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36, 2024.

[3] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, et al. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 803–814, 2023.

[4] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR.

[5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[6] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[7] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9298–9309, 2023.

[8] Yichun Shi, Peng Wang, Jianglong Ye, Mai Long, Kejie Li, and Xiao Yang. Mvdream: Multi-view diffusion for 3d generation. *arXiv preprint arXiv:2308.16512*, 2023.

[9] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023.

[10] Jeong-gi Kwak, Erqun Dong, Yuhe Jin, Hanseok Ko, Shweta Mahajan, and Kwang Moo Yi. Vivid-1-to-3: Novel view synthesis with video diffusion models. *arXiv preprint arXiv:2312.01305*, 2023.

[11] Zehuan Huang, Hao Wen, Junting Dong, Yaohui Wang, Yangguang Li, Xinyuan Chen, Yan-Pei Cao, Ding Liang, Yu Qiao, Bo Dai, et al. Epidiff: Enhancing multi-view synthesis via localized epipolar-constrained diffusion. *arXiv preprint arXiv:2312.06725*, 2023.

[12] Shitao Tang, Jiacheng Chen, Dilin Wang, Chengzhou Tang, Fuyang Zhang, Yuchen Fan, Vikas Chandra, Yasutaka Furukawa, and Rakesh Ranjan. Mvdiffusion++: A dense high-resolution multi-view diffusion model for single or sparse-view 3d object reconstruction. *arXiv preprint arXiv:2402.12712*, 2024.

[13] Vikram Voleti, Chun-Han Yao, Mark Boss, Adam Letts, David Pankratz, Dmitry Tochilkin, Christian Laforte, Robin Rombach, and Varun Jampani. Sv3d: Novel multi-view synthesis and 3d generation from a single image using latent video diffusion, 2024.

[14] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang, Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. *arXiv preprint arXiv:2310.15008*, 2023.

[15] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023.

[16] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. *arXiv preprint arXiv:2402.05054*, 2024.

[17] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. Grm: Large gaussian reconstruction model for efficient 3d reconstruction and generation. *arXiv preprint arXiv:2403.14621*, 2024.

[18] Zhengyi Wang, Yikai Wang, Yifei Chen, Chendong Xiang, Shuo Chen, Dajiang Yu, Chongxuan Li, Hang Su, and Jun Zhu. Crm: Single image to 3d textured mesh with convolutional reconstruction model. *arXiv preprint arXiv:2403.05034*, 2024.

[19] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191*, 2024.

[20] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9150–9161, 2023.

[21] Jiahao Li, Hao Tan, Kai Zhang, Zexiang Xu, Fujun Luan, Yinghao Xu, Yicong Hong, Kalyan Sunkavalli, Greg Shakhnarovich, and Sai Bi. Instant3d: Fast text-to-3d with sparse-view generation and large reconstruction model. *arXiv preprint arXiv:2311.06214*, 2023.

[22] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, Natalia Neverova, Andrea Vedaldi, Oran Gafni, and Filippos Kokkinos. Im-3d: Iterative multiview diffusion and reconstruction for high-quality 3d generation. *arXiv preprint arXiv:2402.08682*, 2024.

[23] Qi Zuo, Xiaodong Gu, Lingteng Qiu, Yuan Dong, Zhengyi Zhao, Weihao Yuan, Rui Peng, Siyu Zhu, Zilong Dong, Liefeng Bo, et al. Videomv: Consistent multi-view generation based on large video generative model. *arXiv preprint arXiv:2403.12010*, 2024.

[24] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2553–2560. IEEE, 2022.

[25] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[26] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

[27] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[28] Axel Sauer, Frederic Boesel, Tim Dockhorn, Andreas Blattmann, Patrick Esser, and Robin Rombach. Fast high-resolution image synthesis with latent adversarial diffusion distillation. *arXiv preprint arXiv:2403.12015*, 2024.

[29] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 35:8633–8646, 2022.

[30] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022.

[31] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

[32] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

[33] Xin Ma, Yaohui Wang, Gengyun Jia, Xinyuan Chen, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024.

[34] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024.

[35] Chuanxia Zheng and Andrea Vedaldi. Free3d: Consistent novel view synthesis without 3d representation. *arXiv preprint arXiv:2312.04551*, 2023.

[36] Jiayu Yang, Ziang Cheng, Yunfei Duan, Pan Ji, and Hongdong Li. Consistnet: Enforcing 3d consistency for multi-view images diffusion. *arXiv preprint arXiv:2310.10343*, 2023.

[37] 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. *Communications of the ACM*, 65(1):99–106, 2021.

[38] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021.

[39] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022.

[40] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuehler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42(4):1–14, 2023.

[41] Hanwen Jiang, Zhenyu Jiang, Yue Zhao, and Qixing Huang. Leap: Liberate sparse-view 3d modeling from camera poses. *arXiv preprint arXiv:2310.01410*, 2023.

[42] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. *arXiv preprint arXiv:2312.09147*, 2023.

[43] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

[44] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309, 2023.

[45] Yuan-Chen Guo, Ying-Tian Liu, Ruizhi Shao, Christian Laforte, Vikram Voleti, Guan Luo, Chia-Hao Chen, Zi-Xin Zou, Chen Wang, Yan-Pei Cao, and Song-Hai Zhang. threestudio: A unified framework for 3d content generation. `https://github.com/threestudio-project/threestudio`, 2023.

[46] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021.

[47] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9298–9309, 2023.

[48] Weiyu Li, Rui Chen, Xuelin Chen, and Ping Tan. Sweetdreamer: Aligning geometric priors in 2d diffusion for consistent text-to-3d. *arXiv preprint arXiv:2310.02596*, 2023.

[49] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

[50] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, and Ying Shan. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 4296–4304, 2024.

[51] Lingteng Qiu, Guanying Chen, Xiaodong Gu, Qi zuo, Mutian Xu, Yushuang Wu, Weihao Yuan, Zilong Dong, Liefeng Bo, and Xiaoguang Han. Richdreamer: A generalizable normal-depth diffusion model for detail richness in text-to-3d. *arXiv preprint arXiv:2311.16918*, 2023.

[52] Peng Wang and Yichun Shi. Imagedream: Image-prompt multi-view diffusion for 3d generation. *arXiv preprint arXiv:2312.02201*, 2023.

[53] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022.

[54] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models.

[55] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016.

[56] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[57] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506, 2020.

A Technical Details

A.1 Video model finetuning

Based on the approach outlined in [32], the generation process employs the EDM framework[53]. Let $p_{\text{data}}(\mathbf{x}_0)$ represent the video data distribution, and $p(\mathbf{x};\sigma)$ be the distribution obtained by adding Gaussian noise with variance σ^2 to the data. For sufficiently large σ_{max}, $p(x;\sigma^2_{\text{max}})$ approximates a normal distribution $\mathcal{N}(0,\sigma^2_{\text{max}})$. Diffusion models (DMs) leverage this property and begin with high variance Gaussian noise, $x_M \sim \mathcal{N}(0,\sigma^2_{\text{max}})$, and then iteratively denoise the data until reaching $\sigma_0 = 0$.

In practice, this iterative refinement process can be implemented through the numerical simulation of the Probability Flow ordinary differential equation (ODE):

$$d\mathbf{x} = -\dot{\sigma}(t)\sigma(t)\nabla_{\mathbf{x}} \log p(\mathbf{x};\sigma(t))\, dt \tag{3}$$

where $\nabla_{\mathbf{x}} \log p((\mathbf{x};\sigma)$ is called as score function.

DM training is to learn a model $s_\theta(\mathbf{x};\sigma)$ to approximate the score function $\nabla_{\mathbf{x}} \log p((\mathbf{x};\sigma)$. The model can be parameterized as:

$$\nabla_{\mathbf{x}} \log p((\mathbf{x};\sigma) \approx s_\theta((\mathbf{x};\sigma) = \frac{D_\theta(\mathbf{x};\sigma) - \mathbf{x}}{\sigma^2}, \tag{4}$$

where D_θ is a learnable denoiser that aims to predict ground truth $\mathbf{x}_0$.

The denoiser D_θ is trained via denoising score matching (DSM):

$$\mathbb{E}_{\mathbf{x}_0 \sim p_{\text{data}}(\mathbf{x}_0),(\sigma,n)\sim p(\sigma,n)} \left[\lambda_\sigma \|D_\theta(\mathbf{x}_0 + n;\sigma) - \mathbf{x}_0\|_2^2\right], \tag{5}$$

where $p(\sigma,n) = p(\sigma)\mathcal{N}(n;0,\sigma^2)$, $p(\sigma)$ is a distribution over noise levels σ, λ_σ is a weighting function. The learnable denoiser D_θ is parameterized as:

$$D_\theta(\mathbf{x};\sigma) = c_{\text{skip}}(\sigma)\mathbf{x} + c_{\text{out}}(\sigma)F_\theta(c_{\text{in}}(\sigma)\mathbf{x}; c_{\text{noise}}(\sigma)), \tag{6}$$

where F_θ is the network to be trained.

We sample $\log\sigma \sim \mathcal{N}(P_{\text{mean}}, P^2_{\text{std}})$, with $P_{\text{mean}} = 1.0$ and $P_{\text{std}} = 1.6$. Then we obtain all the parameters as follows:

$$c_{\text{in}} = \frac{1}{\sqrt{\sigma^2+1}} \tag{7}$$

$$c_{\text{out}} = \frac{-\sigma}{\sqrt{\sigma^2+1}} \tag{8}$$

$$c_{\text{skip}}(\sigma) = \frac{1}{\sigma^2+1} \tag{9}$$

$$c_{\text{noise}}(\sigma) = 0.25\log\sigma \tag{10}$$

$$\lambda(\sigma) = \frac{1+\sigma^2}{\sigma^2} \tag{11}$$

We fine-tune the network backbone F_θ on multi-view images of size 512×512. During training, for each instance in the dataset, we uniformly sample 8 views and choose the first view as the input view. view images of size 512×512.

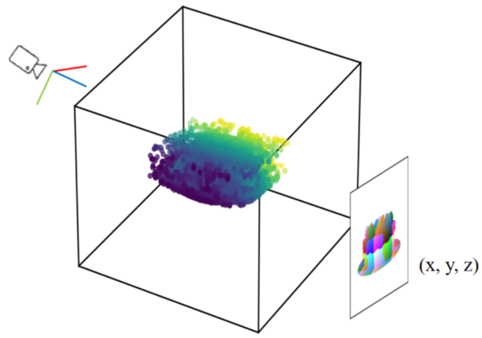

Figure 7: The projection process of coordinates map.

A.2 Coordinates Map

In conditional control models such as ControlNet[54], T2IAdapter, when depth maps are used as input, their range needs to be normalized to the [0, 1] interval, typically using the formula: $(p - p_{mean})/(p_{max} - p_{min})$. However, this normalization process may introduce scale ambiguity, which can affect the multi-view generation performance. To avoid the issues caused by normalization, we use coordinate maps. Coordinate maps transform the depth value d to a common world coordinate system using the camera's intrinsic and extrinsic parameters, represented as (X, Y, Z). The transformation formula is:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = K^{-1} \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \cdot d$$

where (u, v) are the pixel coordinates, d is the corresponding depth value, and K is the camera intrinsic matrix.

A.3 3D Feedback

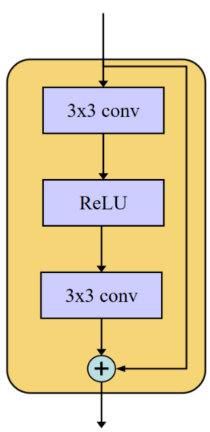

Figure 8: Architecture of the residual block used in feedback stage.

Input	inp $\in \mathbb{R}^{3\times512\times512}$
PixelUnshuffle [55]	$192 \times 64 \times 64$
ResBlock $\times 3$	$320 \times 64 \times 64$
ResBlock $\times 3$	$640 \times 32 \times 32$
ResBlock $\times 3$	$1280 \times 16 \times 16$
ResBlock $\times 3$	$1280 \times 8 \times 8$

Table 4: The detailed structure of all layers in the feedback injection network.

With reference to Section 3.3 in the main paper, Fig. 8 and Table 4 provide a detailed illustration of the feedback injection netwrok. We use two networks to inject the coordinates map and RGB texture map feedback into the score function. Each network consists of four feature extraction blocks and three downsample blocks to adjust the feature resolution. The reconstruction coordinates map and

RGB texture map initially have a resolution of 512×512. We employ the pixel unshuffle operation to downsample these maps to 64×64.

At each scale, three residual blocks[56] are used to extract the multi-scale feedback features, denoted as $F_P = \{F_p^1, F_p^2, F_p^3, F_p^4\}$ and $F_T = \{F_t^1, F_t^2, F_t^3, F_t^4\}$ for the coordinates map and RGB texture map, respectively. These feedback features match the intermediate features $F_{\text{enc}} = \{F_{\text{enc}}^1, F_{\text{enc}}^2, F_{\text{enc}}^3, F_{\text{enc}}^4\}$ in the encoder of the UNet denoiser. The feedback features F_P and F_T are added to the intermediate features F_{enc} at each scale as described by the following equations:

$$\mathbf{F}_p = \mathcal{F}^0(P) \tag{12}$$

$$\mathbf{F}_t = \mathcal{F}^1(T) \tag{13}$$

$$\mathbf{F}_{\text{enc}}^i = \mathbf{F}_{\text{enc}}^i + \mathbf{F}_p^i + \mathbf{F}_t^i, \quad i \in \{1, 2, 3, 4\} \tag{14}$$

where P represents the coordinates map feedback input, and T represents the RGB texture feedback input. $\mathcal{F}^0$ and $\mathcal{F}^1$ denote the functions of the feedback inject network applied to the coordinates map and RGB texture map, respectively.

B Training Details and Experimental Settings

Implementation As illustrate in Table 5, all models are trained for 30,000 iterations using 8 A100 GPUs with a total batch size of 32. We clip the gradient with a maximum norm of 1.0. We use the AdamW optimizer with a learning rate of 1×10^{-5} and employ FP16 mixed precision with DeepSeed[57] with Zero-2 for efficient training. We adjust the cameras in each batch so that the initial input view consistently represents the reference frame, using an identity rotation matrix and a fixed translation for alignment.

The inference settings are shown in Table 6.

Hyperparameter	SVD (1.8 B)	LGM (424M)
Training		
Optimizer	AdamW	AdamW
Learning rate	1e-5	1e-5
Batch size per GPU	4	4
# training steps	40k	40k
# GPUs	8	8
Training time (days)	4	4
Input Resolution	$8 \times 512 \times 512 \times 3$	$4 \times 256 \times 256 \times 3$
Output Resolution	$8 \times 512 \times 512 \times 3$	$- \times 512 \times 512 \times 3$
Diffusion setup		
P_{mean}	1.0	-
P_{std}	1.6	-

Table 5: Hyperparameters for the training stage.

Hyperparameter	SC3D	VideoMV	SV3D	SyncDreamer
Sampling parameters				
Sampler	Euler	DDIM	Euler	DDIM
steps	25	50	50	50
cfg gudiance	$1.0 \sim 3.0$	6.0	6.0	2.0

Table 6: Hyperparameters for the inference stage.

C Additional Visualization Results

Figure 9: Visualization results generated by our SC3D. For each sample (3 rows), the 1st row is ground truth, 2nd row is the generated multi-view images, while 3rd row is the rendered views from reconstructed 3DGS. For each row, the first image is the input image.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly outline the primary contributions of the paper. The claims made are directly supported by the experiments presented in the paper, ensuring an accurate representation of the work's contributions and limitations.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See in Section 4.3.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA] .

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide the GSO generation result and code in the supplemental materials.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We provide the code in the supplemental materials.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: See in Appendix B.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The paper does not provide error bars or any statistical significance measures for the experimental results.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: In the Section 1, we discuss how 3D generation can accelerate various industries by enhancing design processes, improving simulations, and reducing production costs.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper does not involve the release of data or models that have a high risk for misuse.

Guidelines: The paper focuses on foundational research and does not have direct societal implications. It does not address societal impacts.

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The SVD model (https://huggingface.co/stabilityai/stable-video-diffusion-img2vid) is intended for research purposes only. The following assets are used in the paper, and their licenses are properly acknowledged:

- Gobjaverse: https://github.com/modelscope/richdreamer/tree/main/dataset/gobjaverse
- LGM: https://github.com/3DTopia/LGM.git
- Syncdreamer: https://github.com/liuyuan-pal/SyncDreamer.git
- Objaverse: https://huggingface.co/datasets/allenai/objaverse

The use of the Objaverse dataset as a whole is licensed under the ODC-By v1.0 license. Individual objects in Objaverse are licensed under various Creative Commons licenses, including:

- CC-BY 4.0 - 721K objects
- CC-BY-NC 4.0 - 25K objects
- CC-BY-NC-SA 4.0 - 52K objects
- CC-BY-SA 4.0 - 16K objects
- CC0 1.0 - 3.5K objects

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the code and generation results in supplemental materials.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.