Morpheus: Text-Driven 3D Gaussian Splat Shape and Color Stylization

Jamie Wynn*[1] Zawar Qureshi*[1] Jakub Powierza[1] Jamie Watson[1,2] Mohamed Sayed[1]
[1]Niantic [2]UCL
https://nianticlabs.github.io/morpheus/

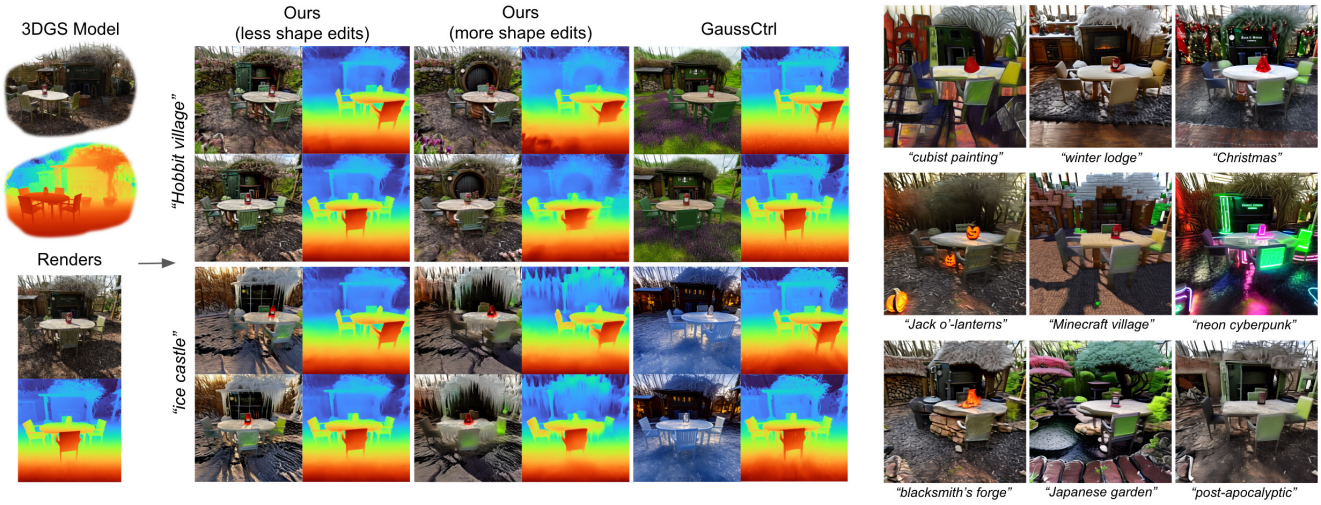

Figure 1. We introduce a new method for stylization of novel views using text prompts. The output of our method is a stylized 3D Gaussian Splatting model of which we show renders here. Our method allows stylization control of both appearance and shape. Using the same prompt, our method can produce different stylizations with variable shape alteration, allowing for more striking shape and color stylization compared to GaussCtrl [87]. We show multiple stylizations of the same scene. Code online.

Abstract

Exploring real-world spaces using novel-view synthesis is fun, and reimagining those worlds in a different style adds another layer of excitement. Stylized worlds can also be used for downstream tasks where there is limited training data and a need to expand a model's training distribution. Most current novel-view synthesis stylization techniques lack the ability to convincingly change geometry. This is because editing geometry while maintaining multi-view consistency is very challenging. In this work, we propose a new autoregressive 3D Gaussian Splatting stylization method. As part of this method, we contribute a new RGBD diffusion model that allows for strength control over appearance and shape stylization. To ensure consistency across stylized frames, we use a combination of novel depth-guided cross attention, feature injection, and a ControlNet conditioned on composite frames for guiding the stylization of new frames. We validate our method via extensive qualitative results, quantitative experiments, and a user study.

1. Introduction

As humans, we want to explore worlds and stories beyond what we experience in our daily lives for entertainment or education. Our history of art reflects this; we started with paintings on stone walls, moved on to pigment on canvas, mastered the art and technicalities of photography, and then invented moving pictures. When this was not enough, we created better ways of experiencing these worlds by either adding a sense of depth as with stereoscopes or Sensorama [29, 83] or by allowing freedom and control over viewpoint [7]. More recently, these experiences have been realized by simulating a world using computer generated graphics. Realism is achieved with work by artists to make textures and models, and then work by graphics researchers to allow real-time renders of these complex and detailed

*denotes equal contribution.

worlds for them to come alive. This also enables the creation of new kinds of worlds that are no longer necessarily 'real' but still detailed and immersive enough to convince viewers and players that they are real.

However, this all comes at a cost. While building 3D representations of structures and objects by hand – often in the form of meshes – allows for full artistic control, it requires expensive artistic skill and time. Recent work aims to cut down the effort needed to bring real objects into the virtual world. This could involve reconstructing a 3D model of an object from one [63] or a collection of images [75, 89]. A more convenient approach is to render novel views of a scene using image-based rendering [32, 43, 54]. The only requirement for these methods is a dense collection of images with camera poses obtained via either SLAM [94, 96] or SfM [74]. The most recent of these methods, 3D Gaussian Splatting [43] (3DGS), uses primitives that can be rendered in real-time using conventional rasterization techniques, allowing almost-seamless integration into existing rendering pipelines. While this gives us realism with little effort, we are limited by what we can capture in the real world. The next challenge is to change these captures to allow for the exploration of, and immersion in, stylized versions of those worlds.

There are many works on altering captures of the real world. The simplest problem is editing or stylizing 2D images, where many recent methods excel [40, 48, 69]; there, stylization is often controlled using language prompts and example images. A more challenging setting is stylizing 3D representations. Since existing 2D generative and stylization models are powerful and mature, they are often used as a building block for 3D editing. These 2D models often lack explicit understanding of geometry, the ability to output a representation of modified geometry, and strength control over appearance and shape stylization. Because of this, modification is often superficial and limited to texture changes, especially since multi-view consistency is required and is easily broken when stylization requires geometry edits.

To tackle these challenges, we introduce a new method for stylizing 3D Gaussian Splats. Our method is informed by the geometry present in the input 3DGS model and allows strength control over appearance and shape stylization. Our method operates on renders of a 3DGS, producing frame-by-frame stylization for arbitrary camera trajectories. We show that 3DGS models made from those stylized frames are qualitatively and quantitatively superior to existing methods. We highlight our contributions as:

1. an autoregressive pipeline for stylizing 3D Gaussian Splatting models of scenes given text prompts,
2. an RGBD stylization diffusion model conditioned on a text prompt and an RGBD image with separate controls over geometry and appearance,
3. a ControlNet conditioned on warped frame composites for propagating previous frame stylization,
4. and depth-informed feature sharing for consistent frame-to-frame stylization.

2. Related Work

Novel-View Synthesis (NVS) is a popular task in computer graphics where, given an image or a collection of posed images of a scene, an image is output from an arbitrary view. Early approaches construct lumigraphs [28] – volumes that capture the behavior of light as it passes through a scene – that can be used to render novel views. Subsequent approaches aim to reconstruct textured geometry [10, 32, 59] to utilize traditional rendering pipelines, multiplane images for layered rendering [77, 79], and learned networks for combining multiple image fragments [33, 67]. More recent methods render novel views volumetrically using implicit functions that learn a radiance field of a scene via gradient descent [54]. An alternate approach [43] is to optimize a set of 3D Gaussian primitives that can be rendered down to images via splatting in real-time. Subsequent work uses regularization during optimization [11, 14, 17, 62], models raw capture [55], improves rendering time [15, 27, 34, 51, 56], improves training speed and/or quality [2, 56, 66, 92], estimates camera parameters [3, 85], or incorporates semantic understanding [44, 64].

2D Image Stylization Early image stylization approaches first extract low level image features such as gradients, edges, and local segments [47] and then place brush strokes [30, 36, 52], build mosaics [46], apply artistic dithering [58], or place cubist blocks [16]. While these methods are capable of limited styles, follow-up work allows the use of templates or reference images [61, 95]. Given the simple explicit library of edits, local edits throughout the image are globally consistent, and such brush strokes or texture abstractions can be propagated through video [37, 84].

Learned stylization has expanded the library of available styles by conditioning generation on example images [40, 48]. Recent diffusion models [38, 71] have improved fidelity and resolution, allowed for stylization [86], enabled text-based image editing [5], and provided generation conditioned on depth, keypoints, or edges when combined with ControlNets [93]. Latent Diffusion models generate images by progressively denoising a 2D latent map using a U-Net and then decoding the latent into the full image. A ControlNet mirrors the architecture of a diffusion model's U-Net and is trained to influence the denoising process of the diffusion model. It first encodes a control condition and then shares intermediate features across to the diffusion model at every layer during the denoising process.

Consistent Stylization 2D stylization can vary dramatically depending on the input image, and so stylization

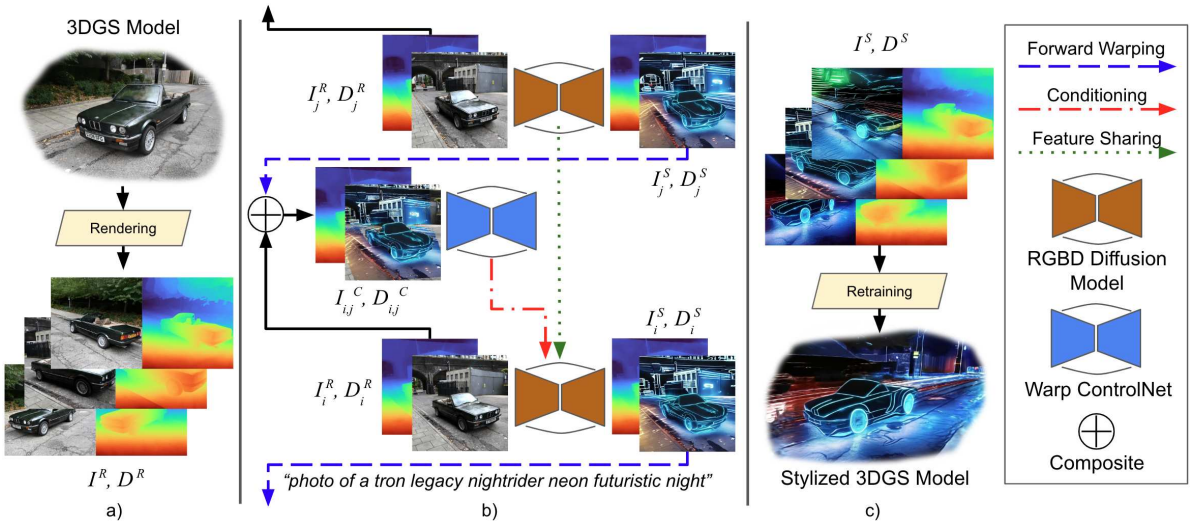

Figure 2. **Method Overview** a) Our pipeline takes as input a novel view synthesis model, in this case a 3D Gaussian Splatting (3DGS) model, and first renders a set of representative images and their depth maps $\{I^R, D^R\}$. b) Our pipeline stylizes rendered images autoregressively. We use a novel RGBD diffusion model (Section 3.1) conditioned on the input RGBD render $\{I^S_i, D^S_i\}$, a stylization prompt, and stylization noise parameters that modulate the strength of appearance and shape stylization. For every subsequent frame, we warp previously stylized frames $\{I^S_j, D^S_j\}$ to the current frame and form a composite $\{I^C_{ij}, D^C_{ij}\}$. We use a Warp ControlNet (Section 3.2) conditioned on the warped composite and a validity mask to guide the RGBD stylization of the current frame $\{I^R_i, D^R_i\}$ to produce $\{I^S_i, D^S_i\}$. During diffusion we use depth-informed feature sharing (Section 3.3) to propagate deep stylization features. c) We then retrain a 3DGS model using newly stylized frames $\{I^S, D^S\}$.

consistency from one frame to another is a challenge. While multiple frames can be generated or edited simultaneously as in video diffusion models [4, 39] or 2D-based 3D scene generation [26], these models are often expensive in terms of training time, inference time, and memory usage. There are also difficulties in acquiring suitable 3D training data. The problem is especially prevalent for NVS stylization, where captions may also be required for the data. Some methods including Instruct-NeRF2NeRF and Instruct-GS2GS [31, 82], SNeRF [57], and VicaNeRF [18] gradually stylize a collection of images by alternating between modifying individual frames and NVS optimization. These methods require lengthy offline processing and produce results with either limited shape alteration or blurry textures. Score Distillation Sampling [60] has been used to generate 3D scenes from 2D models [60, 70, 78], but it often produces hazy results, even when used for stylization as in our early experiments. Other NVS stylization methods achieve multi-view consistency by sharing latent information in intermediate stages through cross-attention as in GaussCtrl [6, 87], direct feature injection as in DGE [13, 35], flow- or depth-based warping [1, 20, 22], or warp-friendly noise representations [9]. These methods may suffer by sharing erroneous information from multiple views with complex geometry. In contrast, we use depth-guided feature sharing that respects the scene's geometry and leads to more consistent stylizations. G3DST [53] train generalizable modules for stylizing NeRFs but exhibits limited stylization beyond texture changes. 3D scene-level stylization is possible via 3D noise representations as in ConsistDreamer [12, 45], Gaussian-embedded features [50], or Gaussian primitive tracing [50]. Notably, ConsistDreamer's stylizations are surface-level with local texture modification. In contrast, our method makes dramatic and controllable geometry changes to the 3DGS scene.

3. Method

Our method takes as input a depth-regularized 3D Gaussian Splatting (3DGS) model of a scene from which we render a set of RGB and depth images using poses from a representative camera trajectory. It also takes in a stylization prompt and two values indicating the strength of both appearance and geometry stylization. The intermediate output of our model is a set of consistently stylized RGBD frames. We use these stylized frames to train our output stylized 3DGS model. In Section 3.1 we describe our 2D RGBD diffusion model that allows for appearance and shape stylization strength control using separate denoising of color and depth. When stylizing every new frame in the sequence, we composite the original render and projections of selected previously stylized frames as input to an RGBD-informed Warp ControlNet (Section 3.2) to guide stylization of new frames. We use a mixture of depth-informed feature-sharing strategies to share feature information across from stylized frames to the current frame (Section 3.3) to encourage consistent stylization. Our method is outlined in Figure 2.

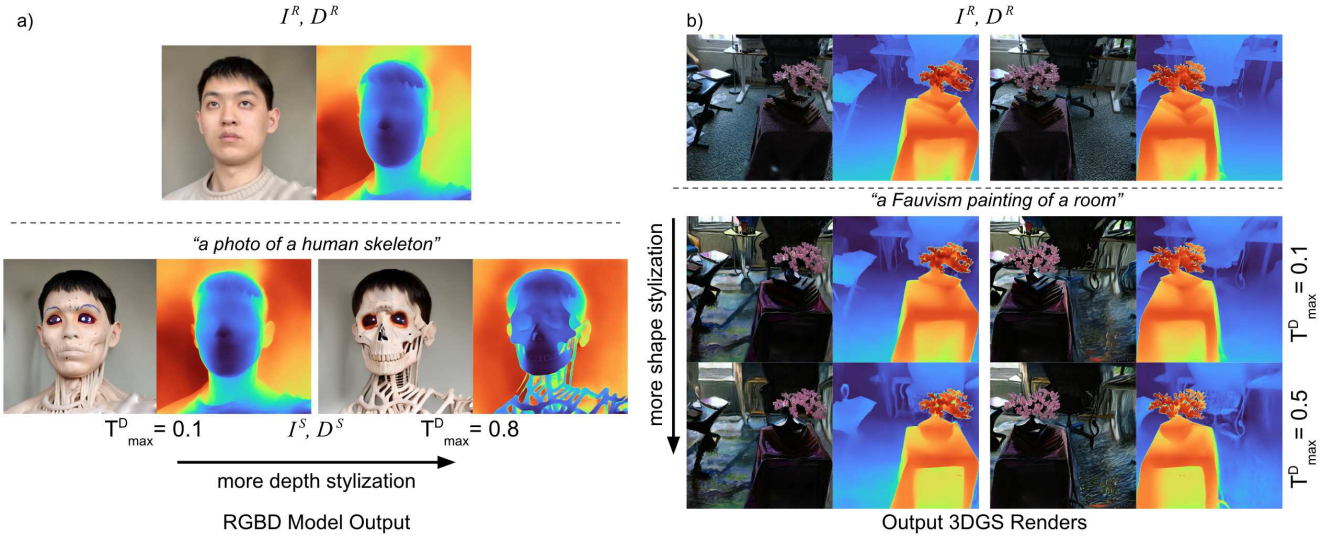

Figure 3. For the same prompt, we vary stylization strength for geometry. a) We show the output of our RGBD model for the same stylization prompt but with varying depth stylization strengths. Note how the depths change when we ask for higher depth stylization but the overall color gamut does not. b) We show the effect of shape stylization in output 3DGS models from our method.

3.1. Geometry and Appearance Stylization

Our RGBD diffusion model takes as input an image render I_i^R, the rendered depth map D_i^R, a prompt, and style strength parameters for both depth and color, and outputs stylized color I_i^S and depth D_i^S. Following previous diffusion stylization methods, during inference we progressively apply noise to our inputs before denoising with prompt conditioning. Starting at diffusion time $t = 0$, we first add Gaussian noise to get to time $t = T_{\text{noise}}$, and then use DDIM inversion to get to noise level T_{max}. This latent is then denoised with the network conditioned on the target prompt. This use of noise rather than inversion for the first few steps allows us to guard against overfitting on the high-frequency components of the input RGBD, since the forward process tends to destroy high-frequency components first.

We wish to control the stylization strength of color and depth separately, while still editing these channels in a manner that keeps them consistent with one another. To do this, we denoise both of them simultaneously and modify the noise schedules for each respective channel by introducing two separate maximum timesteps for adding noise, T^D_{max} and T^I_{max} for depth and color respectively. During denoising, we do not permit the network to change the depth until $t < T^D_{\text{max}}$, nor to change the RGB channels until $t < T^I_{\text{max}}$. Inspired by diffusion inpainting methods [68], we pass in masks M_t^D and M_t^I, consisting of ones if the current noising/denoising timestep satisfies the condition $t \leq T^{I,D}_{\text{max}}$ and 0 otherwise. This allows us to inform the network of whether its changes to the RGB and D channels will be accepted or not on a given denoising step, just as the mask passed into inpainting models informs the network of whether its changes to a given pixel will be accepted. Since we predict scale-invariant depth, we scale the output depth map, D_i^S, using the rendered depth D_i^R. We show examples of variable-depth stylization in Fig. 3.

3.2. Warp ControlNet for Consistent Inpainting

We wish to propagate previous frames' stylization when stylizing new frames. We forward-warp previously stylized frames using their depths D_j^S to the current frame, I_i, and compute warped frames I_{ij}^S, warped depths D_{ij}^S, and validity masks M_{ij}^S where j is the index of a previously stylized frame. For each warped reference frame, we composite it with the unstylized current frame to get I_{ij}^C, D_{ij}^C.

A naive approach to generating a new frame conditional on a previously stylized frame would be to warp the stylized previous frame, and then inpaint missing regions. However, this leaves warping artifacts. Instead, we create a specifically trained custom ControlNet [93] conditioned on the composites I_{ij}^C, D_{ij}^C, the composite mask M_{ij}^S, and the input prompt. This guides the RGBD diffusion model to correct artifacts in the warped region, and inpaint the rest of the image consistently with warped regions. We pass each reference frame's composite through the ControlNet and average guidance features over each reference frame. We elaborate on the model training in Section 4.2.

3.3. Depth-Informed Information Sharing

Forward warping of RGB and depth pixels does not preserve fine textures, and does not capture deep features used in previous frames, which our ControlNet does not have access to. To that end, we use feature injection [80] and cross-attention [6, 87]. These mechanisms help inform layers of the network of how reference frames were stylized. How-

ever, cross-attending and injecting features across all image patches might lead to undesirable effects such as duplicated or misplaced aesthetic features; see Figure 5. While previous work uses epipolar lines to guide this process [13], we use depth information to more precisely transfer feature information from reference frames to the current frame. We start by building 4D heatmaps L_{ij} computed by forward-warping the reference frame depth D_j^S to the current frame. We use a forward warp to guide the transfer of features from reference to source frames by (a) increasing the strength of cross-attention where a reference frame pixel forward-warps to a target-frame pixel, and (b) directly injecting features from reference-frame pixels to corresponding target-frame pixels. We show this visually in Figure 4.

We modify the diffusion model's self-attention layers to allow the image to attend to both itself (as in the unmodified self-attention) and to the reference images, by concatenating together the keys from the original image and the reference images. Our cross-attention then becomes:

$$\text{softmax}\left(\frac{Q[K_{\text{self}}, K_{\text{ref}}]^T}{\sqrt{d_k}} + \log \Delta\right)[V_{\text{self}}, V_{\text{ref}}] \quad (1)$$

where Δ is a mask which we use to control the amount of attention between each key-query pair. When the key is in K_{self}, Δ is a constant λ_{self} (which we set to 0.5 everywhere). Where the key is in K_{ref}, it is equal to L_{ij}, allowing us to modulate the strength of the cross-attention based on our knowledge of the geometry.

During denoising, it is desirable to use a reference image at the same noise level as the image being generated. For this reason, we cache intermediate latents for all frames. At each denoising timestep, we then retrieve the cached reference frame latents from the corresponding timestep and use them for feature-sharing.

For feature injection, we use an argmax across reference features on L_{ij} to select which reference feature to inject into the network. We do not perform feature injection in the first two and last three layers since we only want higher order semantic information and to prevent texture artifacts.

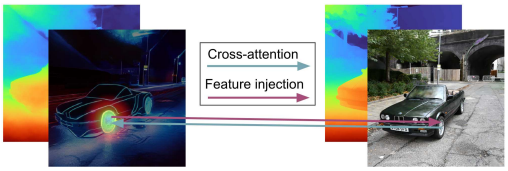

Figure 4. **Feature information sharing** We show a slice through the heatmap L for a single pixel in the target frame.

4. Implementation Details

4.1. Frame Selection, Warping, and Resolution

Our pipeline takes as input and produces output images at a resolution of 512×512. We select a smooth representative trajectory through the 3DGS, so that pose changes from each frame to the next are not too extreme.

We obtain the first frame by running the RGBD model without the ControlNet. For each reference frame, we warp it to the next frame to be generated and composite it with the unstylized new frame, forming the input to our ControlNet. To warp, we construct a mesh by backprojecting stylized depth maps to create vertices, forming mesh edges between neighboring pixels, and clipping edges using a normals check w.r.t the camera look-at. We then render this mesh to the current view using PyTorch3D [65]. All results are generated using warps of and feature sharing from the first and last stylized frames.

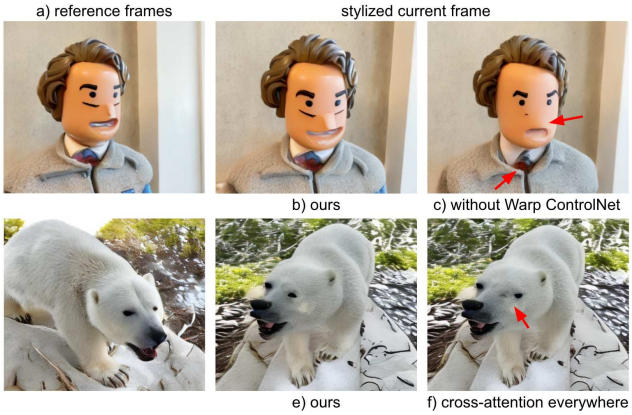

Figure 5. **Qualitative Ablations** We show a) two reference frames, the results of style propagation using ours in b) and e), and then ablations in c) and f). c) without our Warp ControlNet, the geometry and texture on the face and tie are not propagated correctly. f) by cross attending everywhere across the entirety of the frame without depth-informed feature sharing, patches like the bear's eye may be misplaced or repeated leading to inconsistency.

4.2. Architecture and Training

For our RGBD latent model, we initialize with Stable Diffusion 2.1 [68] as a base. We encode both RGB and depth images separately. Following [41], we encode the depth map by stacking it three times along the channel dimension and using the image encoder. For our denoising U-Net [68], the input channels are four encoded feature channels and mask channels M_t^D and M_t^I. We train our RGBD model on 500k images from the aesthetic subset of the Re-LAION-2B dataset [76]. For supervising our model's depth output we generate depth maps for every training-set image using a combination of three off-the-shelf depth models to prevent overfitting to the characteristics of any particular depth model: Depth Anything V2 [88], Marigold [42], and GeoWizard [24]. We apply Gaussian blur to these depth maps 75% of the time. We use noise scheduler strategies from [49].

Our ControlNet model is based on the original implementation [93]. We change the hint input channel size to

	CLIP		Image Consistency		Time
	CLIP Direction Similarity ↑	CLIP Direction Consistency ↑	RMSE ↓	LPIPS ↓	
Instruct NeRF2NeRF [31]	.098	.531	.0463	.0540	∼ hours
Instruct GS2GS [81]	.097	.519	.0501	*.0403*	∼ 30 minutes
GaussCtrl [87]	*.123*	*.590*	.0471	.0438	∼ 10 minutes
DGE [13]	.113	.565	*.0384*	.0407	∼ 10 minutes
Ours	**.175**	**.606**	**.0370**	**.0378**	∼ 10 minutes

Table 1. **Quantitative Evaluation** We compute metrics for 53 stylizations on a range of published and new scenes. CLIP Direction Similarity measures how well the stylization implied by the prompt is respected. CLIP Direction Consistency measures how consistent this stylization is across frames. We also compute image consistency scores from [23]. Ours outperforms other novel-view stylization methods.

accommodate the 5 channels of our composited RGBD image and its mask. We also expand the channel sizes of the hint network from $(16, 32, 96, 256)$ to $(48, 96, 192, 384)$ to allow it to better understand the stylization and compositing problem in our input. To train this ControlNet, we generate pairs of training data by predicting depth maps for RGB images, stylizing those frames using our RGBD model (which we train before the ControlNet), warping those stylized frames to an arbitrary camera, and warping back. For our training set, we generate 250k pairs from our RGBD model using 10k prompts. We sample 6 random camera transforms for each. See the supplemental for further details.

4.3. 3DGS Optimization

We train a 3D Gaussian Splatting (3DGS) model for every scene before rendering trajectories for use in our method. To produce 3DGS models with good depth renders, we use depth and normal regularization. We first run a monocular depth model, Metric3D [91], on each training view, render a depth map from the 3DGS at that view, and median-scale the rendered depth map using the Metric3D depth prediction. We compute a depth loss using a scale-invariant loss [19]. We compute dot-product and cosine-similarity losses on normal maps from both depth maps made using cross products on local image gradients [73].

The final stage of our pipeline is training a new 3DGS model using stylized output color and depth maps from our method. In this instance, we regularize using our method's depth predictions, as well as normals derived from them. We also use a Total Variation L1 (TVL1) [8] loss on rendered normals from depth as regularization to reduce floating artifacts. See the supplemental for details.

5. Experiments

We evaluate our method both quantitatively (Table 1), qualitatively (Figure 7), and with a user study. We evaluate on scenes from Instruct-NeRF2NeRF [31], GaussCtrl [87], ScanNet++ [90], Mip-NeRF360 [2], and our new scenes. We compare against a video editing model, CCEdit [21], in the supplemental. For all baselines, we use official code.

Evaluation Frame Selection Our method uses a smooth camera trajectory through the splat, whereas our baselines take an unordered set of sparse frames (for which we use the original training views of the splat). In order to fairly evaluate our method against our baselines, we find nearest neighbor pairs of poses between our trajectory and the original training views used by our baselines. We interpolate halfway between every such pair of views. We use these views for evaluating quantitatively and for the user study.

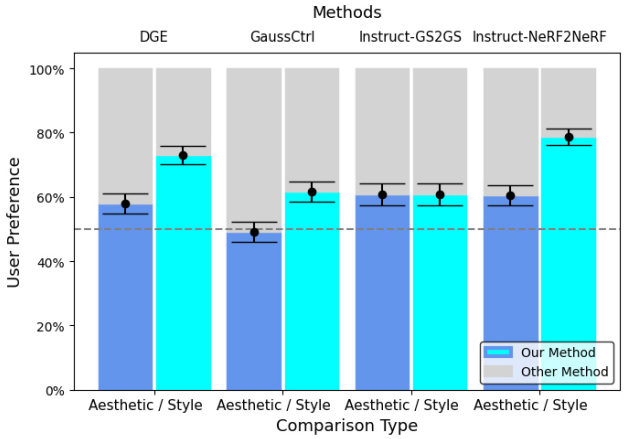

Figure 6. **A/B User Study** 31 participants consistently preferred our method's adherence to **style** prompts and found it to either match or exceed other methods in **aesthetic** quality.

Metrics We compute CLIP metrics defined in [5, 25, 31]. Specifically, we first compute the vector direction between the negative prompt and the stylization prompt. We also compute the vector direction between the unstylized and stylized images. Both vector directions should agree if the prompt is adhered to; this metric is labeled CLIP Direction Similarity. We also measure the consistency in stylization by computing the change in image vector directions across frames, CLIP Direction Consistency.

Qualitative Evaluation We show extensive qualitative results as CLIP metrics alone are not very reliable at judging stylization quality [31, 87]. We experiment with varying the strength of our geometry stylization on both the intermediate RGBD diffusion model and the final 3DGS in Fig. 3. Notably, in the final 3DGS stylizations, our method is capable of leaving texture information consistent while changing geometry with varying T^D_{max}. Our method stylizes the scene beyond simple local texture and color edits compared to ConsistDreamer [12] in Fig. 8. We show a side-by-side comparison of our model and baselines in Fig. 7; our model

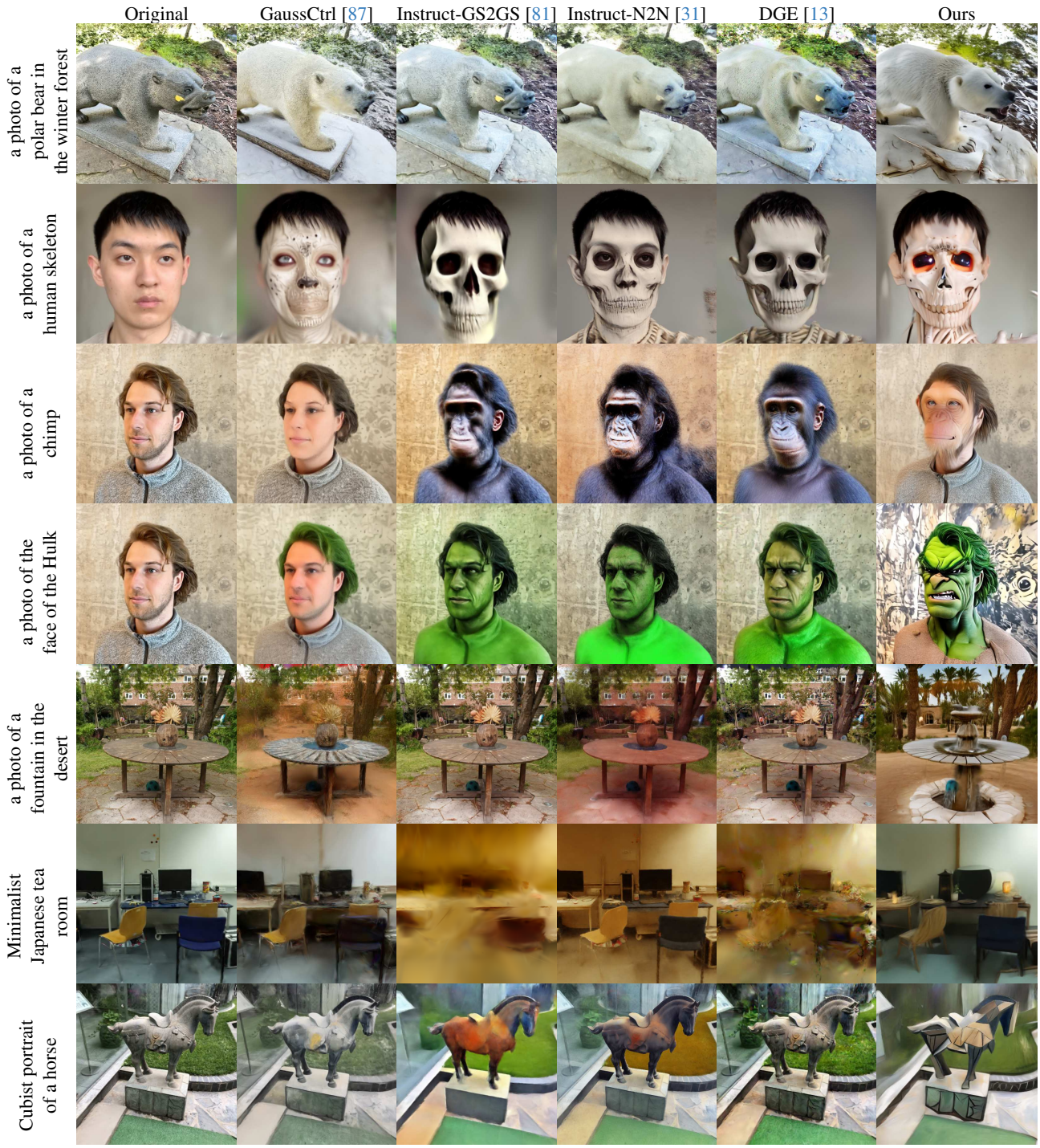

Figure 7. **Qualitative Comparison from Novel Views** Our method's ability to change the scene's shape allow its stylizations to be more aesthetically pleasing and exhibit more adherence to the style prompt.

consistently outperforms previous methods in both shape editing and overall quality.

User Study We report an A/B user study with 31 participants in Figure 6. We use 49 prompt/scene combinations and render A/B comparison videos from a circular wiggle at randomly selected evaluation views of the unstylized splat, our method, and one of each of the baselines from Table 1. We randomly select 8 A/B videos for each baseline for a total of 32 videos. We ask users the following questions for each video: *Of the two videos, which one most closely*

follows the **style** *description?* and *Of the two videos, which one is most* **aesthetically** *pleasing?*. We note that the videos are not cherry picked and are rendered from difficult evaluation views using challenging stylization prompts.

Figure 8. Qualitative comparison of our method with ConsistDreamer [12] since code is not available. Our method alters the scene beyond local texture and color augmentations.

Ablations We ablate our core contributions one by one in Table 2 and show qualitative comparisons in Figure 5. Inconsistent pipeline output would lead to blurry 3DGS models whose blurry renders exhibit deceptively low RMSE and high consistency. Therefore, we define the metrics *Sequential RMSE* and *Sequential LPIPS* as the RMSE and LPIPS respectively computed between successive pairs of stylized frames from our pipeline (prior to 3DGS retraining), where we warp one to the other. This allows us to quantify how good 3D consistency is between successive views from our pipeline. In Table 2, (1) is naive stylization with nothing to enforce consistency between frames; (2) uses an RGB inpainting model with a ControlNet conditioned on depth; and (3) adds depth prediction on stylized frames for warping and compositing. (6) Without our Warp ControlNet, stylization is incorrectly propagated and artifacts are left behind, resulting in worse sequential RMSE and LPIPS. Not sharing features (4) or performing cross-attention equally across the whole of the reference images (5) without depth guidance have a similar effect on metrics.

	Similarity ↑	Consistency ↑	Seq. RMSE ↓	Seq. LPIPS ↓
(1) Single-Frame Independent Stylization with Sec 3.1	0.161	0.592	.1170	.0941
(2) Warp + RGB Inpaint + Depth ControlNet	0.110	0.581	**.0585**	.0959
(3) Warp + RGB Inpaint + Depth ControlNet + DAv2 [88]	0.104	**0.629**	.0776	.0975
(4) w\o feature sharing (3.3)	*0.178*	*0.611*	.0817	.0931
(5) w full x-attn, no feat. injection (3.3)	**0.180**	0.610	.0730	*.0914*
(6) w\o Warp ControlNet (3.2) with inpainting	0.170	0.604	.0834	.0917
(7) Ours	0.175	0.606	*.0702*	**.0911**

Table 2. **Quantitative Ablations** We ablate our core contributions and report their effect on scores.

6. Limitations

Our method's stylization quality is dependent on trajectory selection. Since our method is reliant on images and depths rendered from the original NVS model, we occasionally inherit errors and insert them into stylized output, see Figure 9. We used 3DGS for a balance of speed and quality; since our method is agnostic to the underlying NVS representation, this component could be swapped in for other ones. Our method takes ∼10mins to run on average. While this is on par or faster than baselines, further improvements could be made via faster diffusion models [72].

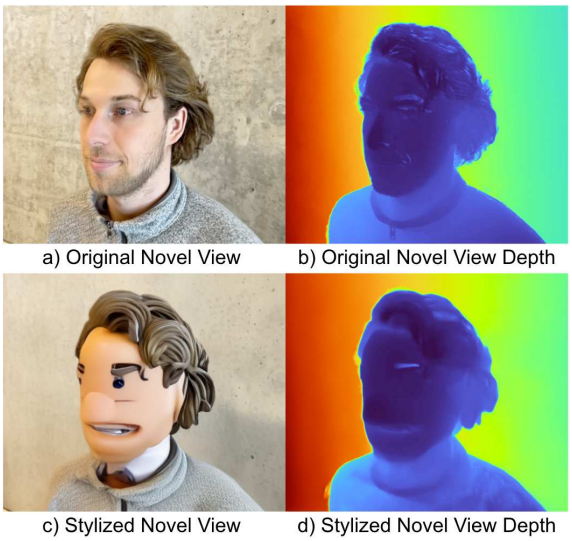

Figure 9. **Limitations** Given errors in the original 3D Gaussian Splat model (see the fuzziness on the right shoulder in a) and the incorrect depth on the eyebrow in b), our method will sometimes inherit these errors in the stylization.

7. Conclusion

We have presented a new method for 3D Gaussian Splatting stylization. Our new method utilizes a novel RGBD model for stylization strength control over shape and appearance, a Warp ControlNet for consistently propagating stylizations, and depth-guided feature injection and cross attention. We validated our contributions and the superiority of our method on a user study, a quantitative benchmark, and through qualitative results.

8. Acknowledgements

We are grateful to the following colleagues for their support and helpful discussions: Sara Vicente, Saki Shinoda, Stanimir Vichev, Michael Firman, and Gabriel Brostow.

References

[1] Yuxiang Bao, Di Qiu, Guoliang Kang, Baochang Zhang, Bo Jin, Kaiye Wang, and Pengfei Yan. Latentwarp: Consistent diffusion latents for zero-shot video-to-video translation. *arXiv preprint arXiv:2311.00353*, 2023. 3

[2] 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. 2, 6

[3] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4160–4169, 2023. 2

[4] 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. 3

[5] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023. 2, 6

[6] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22560–22570, 2023. 3, 4

[7] Marcus Carter and Ben Egliston. Picturing early virtual reality. *Published February 2024 by the Critical Augmented and Virtual Reality Researchers Network (https://cavrn. org/).© Kate Clark, Marcus Carter, Ben Egliston &*, page 18. 1

[8] Tony F Chan and Selim Esedoglu. Aspects of total variation regularized l 1 function approximation. *SIAM Journal on Applied Mathematics*, 65(5):1817–1837, 2005. 6

[9] Pascal Chang, Jingwei Tang, Markus Gross, and Vinicius C Azevedo. How i warped your noise: a temporally-correlated noise prior for diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[10] Gaurav Chaurasia, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis. Depth synthesis and local warps for plausible image-based navigation. *ACM transactions on graphics (TOG)*, 32(3):1–12, 2013. 2

[11] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14124–14133, 2021. 2

[12] Jun-Kun Chen, Samuel Rota Bulò, Norman Müller, Lorenzo Porzi, Peter Kontschieder, and Yu-Xiong Wang. Consistdreamer: 3d-consistent 2d diffusion for high-fidelity scene editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21071–21080, 2024. 3, 6, 8

[13] Minghao Chen, Iro Laina, and Andrea Vedaldi. Dge: Direct gaussian 3d editing by consistent multi-view editing. 2024. 3, 5, 6

[14] Yuedong Chen, Haofei Xu, Chuanxia Zheng, Bohan Zhuang, Marc Pollefeys, Andreas Geiger, Tat-Jen Cham, and Jianfei Cai. Mvsplat: Efficient 3d gaussian splatting from sparse multi-view images. *arXiv preprint arXiv:2403.14627*, 2024. 2

[15] Zhiqin Chen, Thomas Funkhouser, Peter Hedman, and Andrea Tagliasacchi. Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16569–16578, 2023. 2

[16] John P Collomosse and Peter M Hall. Cubist style rendering from photographs. *IEEE Transactions on Visualization and Computer Graphics*, 9(4):443–453, 2003. 2

[17] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022. 2

[18] Jiahua Dong and Yu-Xiong Wang. Vica-nerf: View-consistency-aware 3d editing of neural radiance fields. *Advances in Neural Information Processing Systems*, 36, 2024. 3

[19] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. *Advances in neural information processing systems*, 27, 2014. 6

[20] Xiang Fan, Anand Bhattad, and Ranjay Krishna. Videoshop: Localized semantic video editing with noise-extrapolated diffusion inversion. *arXiv preprint arXiv:2403.14617*, 2024. 3

[21] Ruoyu Feng, Wenming Weng, Yanhui Wang, Yuhui Yuan, Jianmin Bao, Chong Luo, Zhibo Chen, and Baining Guo. Ccedit: Creative and controllable video editing via diffusion models, 2024. 6

[22] Yutang Feng, Sicheng Gao, Yuxiang Bao, Xiaodi Wang, Shumin Han, Juan Zhang, Baochang Zhang, and Angela Yao. Wave: Warping ddim inversion features for zero-shot text-to-video editing. ECCV, 2024. 3

[23] Yutang Feng, Sicheng Gao, Yuxiang Bao, Xiaodi Wang, Shumin Han, Juan Zhang, Baochang Zhang, and Angela Yao. Wave: Warping ddim inversion features for zero-shot text-to-video editing. In *European Conference on Computer Vision*, pages 38–55. Springer, 2025. 6

[24] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. Geowiz-

ard: Unleashing the diffusion priors for 3d geometry estimation from a single image. In *European Conference on Computer Vision*, pages 241–258. Springer, 2025. 5

[25] Rinon Gal, Or Patashnik, Haggai Maron, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators. *ACM Transactions on Graphics (TOG)*, 41(4):1–13, 2022. 6

[26] Ruiqi Gao, Aleksander Holynski, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul Srinivasan, Jonathan T Barron, and Ben Poole. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv preprint arXiv:2405.10314*, 2024. 3

[27] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14346–14355, 2021. 2

[28] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 453–464. 2023. 2

[29] Nicholaus Gutierrez. The ballad of morton heilig: on vr's mythic past. *JCMS: Journal of Cinema and Media Studies*, 62(3):86–106, 2023. 1

[30] Paul Haeberli. Paint by numbers: Abstract image representations. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, pages 207–214, 1990. 2

[31] Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19740–19750, 2023. 3, 6

[32] Peter Hedman, Tobias Ritschel, George Drettakis, and Gabriel Brostow. Scalable Inside-Out Image-Based Rendering. 35(6):231:1–231:11, 2016. 2

[33] Peter Hedman, Julien Philip, True Price, Jan-Michael Frahm, George Drettakis, and Gabriel Brostow. Deep blending for free-viewpoint image-based rendering. 37(6):257:1–257:15, 2018. 2

[34] Peter Hedman, Pratul P Srinivasan, Ben Mildenhall, Jonathan T Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5875–5884, 2021. 2

[35] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control.(2022). *URL https://arxiv. org/abs/2208.01626*, 2022. 3

[36] Aaron Hertzmann. Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 453–460, 1998. 2

[37] Aaron Hertzmann and Ken Perlin. Painterly rendering for video and interaction. In *Proceedings of the 1st international symposium on Non-photorealistic animation and rendering*, pages 7–12, 2000. 2

[38] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2

[39] 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. 3

[40] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017. 2

[41] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 5

[42] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9492–9502, 2024. 5

[43] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2

[44] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19729–19739, 2023. 2

[45] Umar Khalid, Hasan Iqbal, Nazmul Karim, Jing Hua, and Chen Chen. Latenteditor: Text driven local editing of 3d scenes. *arXiv preprint arXiv:2312.09313*, 2023. 3

[46] Junhwan Kim, Fabio Pellacini, et al. Jigsaw image mosaics. *ACM Transactions on Graphics*, 21(3):657–664, 2002. 2

[47] Jan Eric Kyprianidis, John Collomosse, Tinghuai Wang, and Tobias Isenberg. State of the" art": A taxonomy of artistic stylization techniques for images and video. *IEEE transactions on visualization and computer graphics*, 19(5):866–885, 2012. 2

[48] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 702–716. Springer, 2016. 2

[49] Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. Common diffusion noise schedules and sample steps are flawed. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 5404–5411, 2024. 5

[50] Kunhao Liu, Fangneng Zhan, Muyu Xu, Christian Theobalt, Ling Shao, and Shijian Lu. Stylegaussian: Instant 3d style transfer with gaussian splatting. *arXiv preprint arXiv:2403.07807*, 2024. 3

[51] Tianqi Liu, Guangcong Wang, Shoukang Hu, Liao Shen, Xinyi Ye, Yuhang Zang, Zhiguo Cao, Wei Li, and Ziwei

Liu. Fast generalizable gaussian splatting reconstruction from multi-view stereo. *arXiv preprint arXiv:2405.12218*, 2024. 2

[52] Jingwan Lu, Pedro V Sander, and Adam Finkelstein. Interactive painterly stylization of images, videos and 3d animations. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, pages 127–134, 2010. 2

[53] Adil Meric, Umut Kocasari, Matthias Nießner, and Barbara Roessle. G3dst: Generalizing 3d style transfer with neural radiance fields across scenes and styles. *arXiv preprint arXiv:2408.13508*, 2024. 3

[54] 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. 2

[55] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P Srinivasan, and Jonathan T Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16190–16199, 2022. 2

[56] 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. 2

[57] Thu Nguyen-Phuoc, Feng Liu, and Lei Xiao. Snerf: stylized neural implicit representations for 3d scenes. *arXiv preprint arXiv:2207.02363*, 2022. 3

[58] Victor Ostromoukhov and Roger D Hersch. Multi-color and artistic dithering. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 425–432, 1999. 2

[59] Eric Penner and Li Zhang. Soft 3d reconstruction for view synthesis. *ACM Transactions on Graphics (TOG)*, 36(6):1–11, 2017. 2

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

[61] Tania Pouli and Erik Reinhard. Progressive color transfer for images of arbitrary dynamic range. *Computers & Graphics*, 35(1):67–80, 2011. 2

[62] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2

[63] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, et al. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. *arXiv preprint arXiv:2306.17843*, 2023. 2

[64] Minghan Qin, Wanhua Li, Jiawei Zhou, Haoqian Wang, and Hanspeter Pfister. Langsplat: 3d language gaussian splatting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20051–20060, 2024. 2

[65] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 5

[66] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14335–14345, 2021. 2

[67] Gernot Riegler and Vladlen Koltun. Free view synthesis. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIX 16*, pages 623–640. Springer, 2020. 2

[68] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 4, 5

[69] 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. 2

[70] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 22500–22510, 2023. 3

[71] 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. 2

[72] Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. Adversarial diffusion distillation. In *European Conference on Computer Vision*, pages 87–103. Springer, 2024. 8

[73] Mohamed Sayed, John Gibson, Jamie Watson, Victor Prisacariu, Michael Firman, and Clément Godard. Simplerecon: 3d reconstruction without 3d convolutions. In *European Conference on Computer Vision*, pages 1–19. Springer, 2022. 6

[74] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[75] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 2

[76] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022. 5

[77] Richard Szeliski and Polina Golland. Stereo matching with transparency and matting. *International Journal of Computer Vision*, 32(1):45–61, 1999. 2

[78] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023. 3

[79] Richard Tucker and Noah Snavely. Single-view view synthesis with multiplane images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 551–560, 2020. 2

[80] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1921–1930, 2023. 4

[81] Cyrus Vachha and Ayaan Haque. Instruct-gs2gs: Editing 3d gaussian splats with instructions (2024). *URL https://instruct-gs2gs. github. io*. 6

[82] Cyrus Vachha and Ayaan Haque. Instruct-gs2gs: Editing 3d gaussian splats with instructions, 2024. 3

[83] Nicholas J Wade. Charles wheatstone (1802–1875), 2002. 1

[84] Jue Wang, Yingqing Xu, Heung-Yeung Shum, and Michael F Cohen. Video tooning. In *ACM SIGGRAPH 2004 Papers*, pages 574–583. 2004. 2

[85] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf–: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 2

[86] Zhizhong Wang, Lei Zhao, and Wei Xing. Stylediffusion: Controllable disentangled style transfer via diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7677–7689, 2023. 2

[87] Jing Wu, Jia-Wang Bian, Xinghui Li, Guangrun Wang, Ian Reid, Philip Torr, and Victor Adrian Prisacariu. Gaussctrl: multi-view consistent text-driven 3d gaussian splatting editing. *arXiv preprint arXiv:2403.08733*, 2024. 1, 3, 4, 6

[88] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*, 2024. 5, 8

[89] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. *European Conference on Computer Vision (ECCV)*, 2018. 2

[90] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023. 6

[91] Wei Yin, Chi Zhang, Hao Chen, Zhipeng Cai, Gang Yu, Kaixuan Wang, Xiaozhi Chen, and Chunhua Shen. Metric3d: Towards zero-shot metric 3d prediction from a single image. 2023. 6

[92] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv preprint arXiv:2010.07492*, 2020. 2

[93] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 2, 4, 5

[94] Youmin Zhang, Fabio Tosi, Stefano Mattoccia, and Matteo Poggi. GO-SLAM: Global optimization for consistent 3D instant reconstruction. In *ICCV*, 2023. 2

[95] Mingtian Zhao and Song-Chun Zhu. Portrait painting using active templates. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Non-Photorealistic Animation and Rendering*, pages 117–124, 2011. 2

[96] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R. Oswald, and Marc Pollefeys. NICE-SLAM: Neural implicit scalable encoding for SLAM. In *CVPR*, 2022. 2