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/

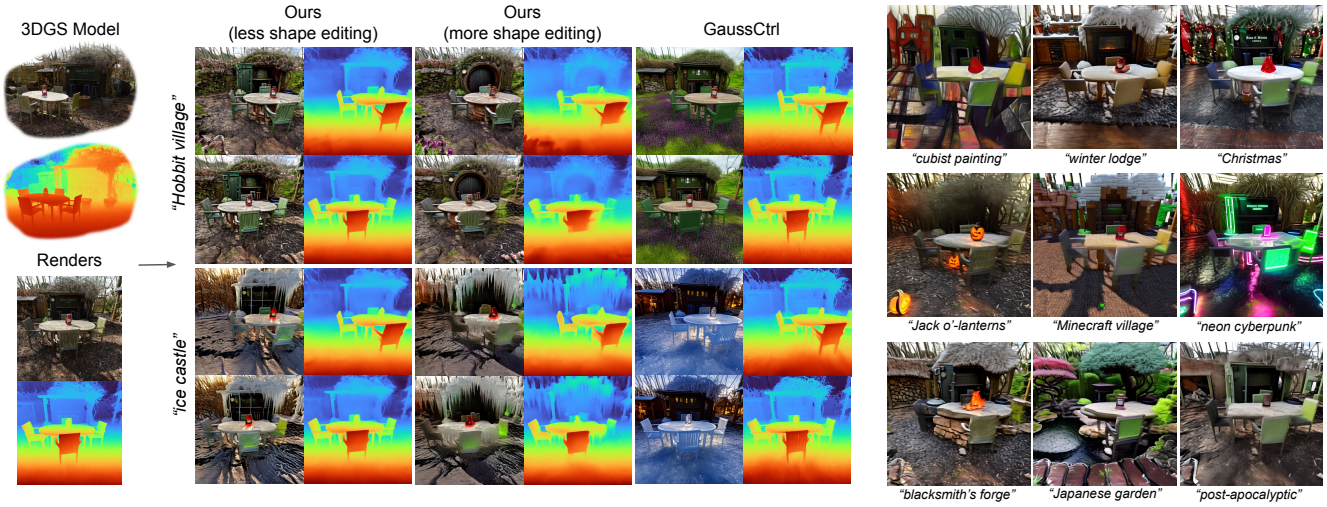

Figure 1. We introduce a new method for novel-view stylization using text prompts. The output of our method is a stylized 3D Gaussian Splatting model, from which we show renders here. Our method allows stylization control of both appearance and shape. Using the same prompt, our method can produce different stylizations with the same overall texture, but variable shape alteration allowing for more striking shape and color stylization compared to GaussCtrl [88]. We show multiple stylizations of the same scene.

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 any geometry change requires increased style strength which is often capped for stylization stability and consistency. 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 Warp 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. Code will be released online.

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 pushed for better ways of experiencing these worlds by either adding a sense of depth as with stereoscopes or Sensorama [30, 84] or by allowing freedom and control over viewpoint [8]. More recently, these experiences have been realized by simulating a world using computer generated graphics. Realism is achieved with painstaking work by artists to make textures and models, and then work

*denotes equal contribution.

by graphics researchers to allow real-time renders of these complex and detailed 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 [65] or a collection of images [76, 90]. A more convenient approach is to render novel views of a scene using image-based rendering [33, 44, 55]. The only requirement for these methods is a dense collection of images with camera poses obtained via either SLAM [95, 97] or SfM [75]. The most recent of these methods, 3D Gaussian Splatting [44] (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 [41, 49, 71]; 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. There, stylization has to be consistent from one view to another, or viewer immersion is broken. 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 with increased stylization strength – especially if geometry is edited.

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 stylized 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 [29] – 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 [11, 33, 61] to utilize traditional rendering pipelines, multiplane images for layered rendering [78, 80], and learned networks for combining multiple image fragments [34, 69]. More recent methods render novel views volumetrically using implicit functions that learn a radiance field of a scene via gradient descent [55]. An alternate approach [44] 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 [12, 15, 18, 64], models raw capture [56], improves rendering time [16, 28, 35, 52, 58], improves training speed and/or quality [3, 58, 68, 93], estimates camera parameters [4, 86], or incorporates semantic understanding [45, 66].

2D Image Stylization Early image stylization approaches first extract low level image features such as gradients, edges, and local segments [48] and then place brush strokes [31, 37, 53], build mosaics [47], apply artistic dithering [60], or place cubist blocks [17]. While these methods are capable of limited styles, follow-up work allows the use of templates or reference images [63, 96]. 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 [38, 85].

Learned stylization has significantly expanded the library of available styles by conditioning generation on example images [41, 49]. More recent diffusion models [39, 73] have improved fidelity and resolution, allowed for stylization [87], enabled text-based image editing [6], and provided generation conditioned on depth, keypoints, or edges when combined with ControlNets [94]. Latent Diffusion models generate images by progressively denoising a noisy 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

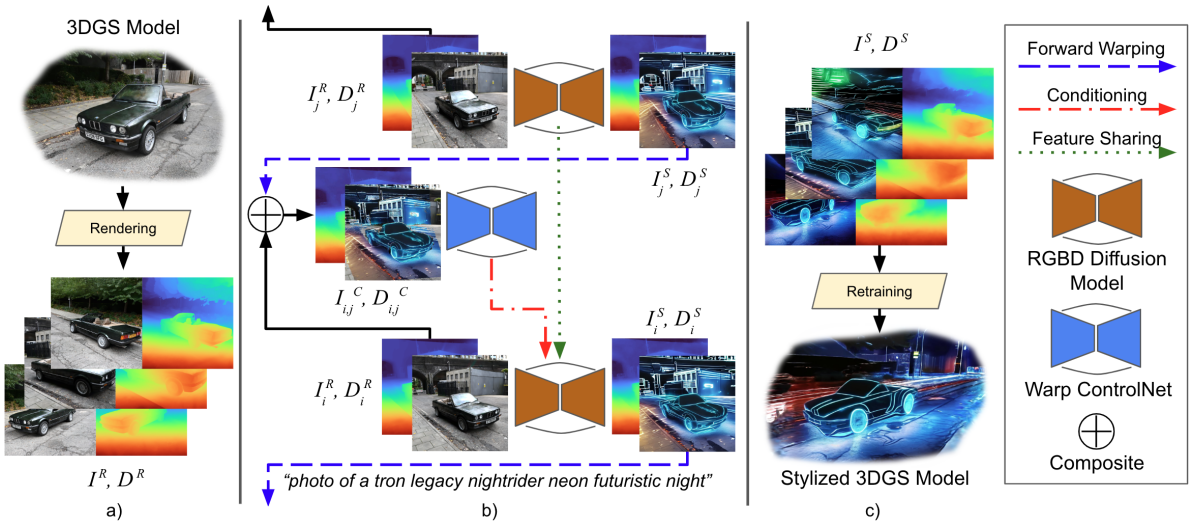

Figure 2. **Method Overview** a) Our pipeline takes as input a novel view synthesis model, in this case a 3D Gaussian Splatting (3DGS) model, and first renders a set of representative images and their depth maps $\{I^R, D^R\}$. b) Our pipeline stylizes rendered images autoregressively. We use a novel 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 $\{I^S, D^S\}$.

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 consistency from one frame to another is a challenge. While multiple frames can be generated or edited simultaneously as in video diffusion models [5, 40] or 2D-based 3D scene generation [27], 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 [32, 83], SNeRF [59], and VicaNeRF [19] 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 [62] has been used to generate 3D scenes from 2D models [62, 72, 79], 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 [7, 88], direct feature injection as in DGE [14, 36], flow- or depth-based warping [2, 21, 23], or warp-friendly noise representations [10]. 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 [54] 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 [13, 46], Gaussian-embedded features [51], or Gaussian primitive tracing [51]. Notably, ConsistDreamer's stylizations are surface level with local texture modification. In contrast, our method makes dramatic and controllable geometry changes to the source 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 a RGBD-informed Warp ControlNet (Section 3.2) to guide stylization of new

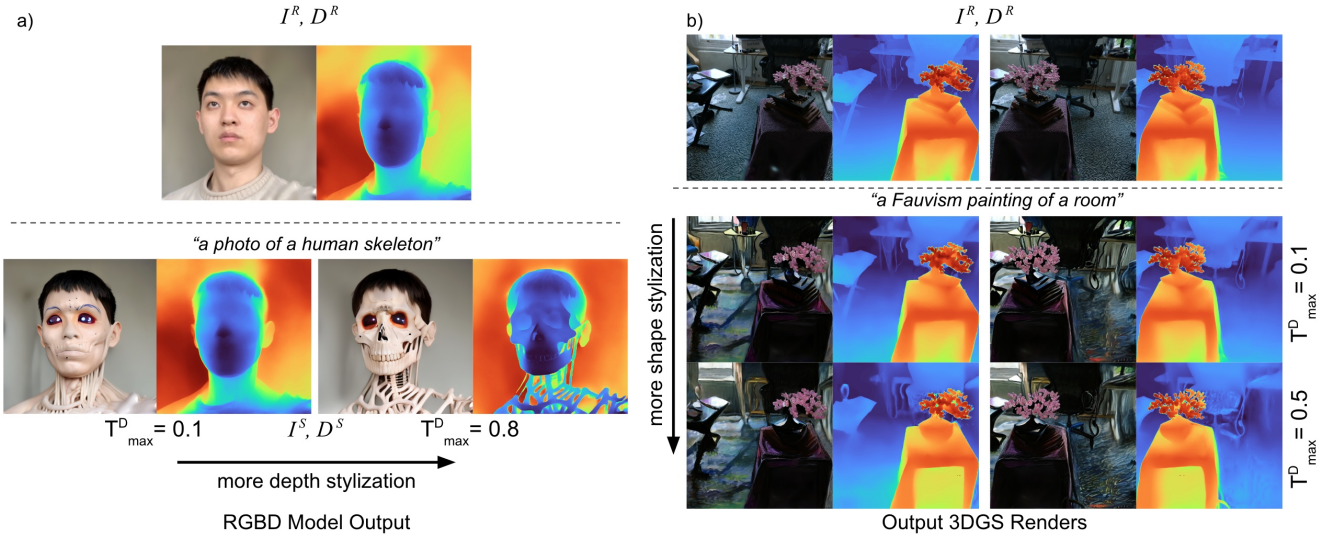

Figure 3. For the same prompt, we vary stylization strength for geometry. a) We show the output of our RGBD model for the same stylization prompt but with varying depth stylization strengths. Note how the depths change when we ask for higher depth stylization but the overall color gamut does not. b)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

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.

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. Specifically, we use DDIM inversion [57] on image and depth latents. In particular, 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.

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 [70], 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 stylized depth map, D_i^S, from the RGBD model 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 [94] 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. To support reference frames, we average guidance features from each pass of the ControlNet on every composite from each reference frame j. 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 utilize both feature injection [81] and cross-attention [7, 88]. These mechanisms help inform layers of the network of how reference frames were stylized. However, 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 [14], 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.

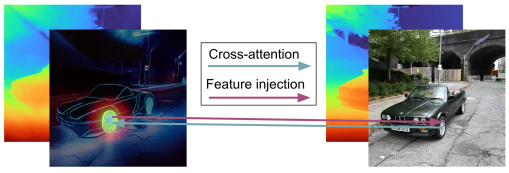

Figure 4. **Feature information sharing** We show a slice through the heatmap L for a single pixel in the target frame.

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, creating mesh edges between neighboring pixels, and cliping edges using a normals check w.r.t the camera look-at. We then render this mesh to the current view using PyTorch3D [67]. All results are generated using warps of and featuring sharing from the first and last stylized frames.

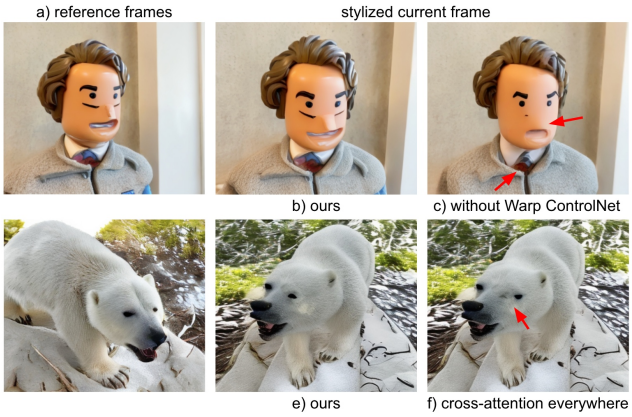

Figure 5. **Qualitative Ablations** We show two a) reference frames, the results of style propagation using ours in b) and e), and then ablations in c) and f). c) without our Warp ControlNet, the geometry and texture on the face and tie are not propagated correctly. f) by cross attending everywhere across 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 StableDiffusion 2.1 [70] as a base. We encode both RGB and depth images separately. Following [42], we encode the depth map by stacking it three times along the channel dimension and using the image encoder. For our denoising U-Net [70], the input channels are four encoded features channels and one time-mask channel for each of depth and RGB. We train our RGBD model on 500k images from the aesthetic subset of the Re-LAION-2B dataset [77]. 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 [89],

	CLIP		Image Consistency	
	CLIP Direction Similarity ↑	CLIP Direction Consistency ↑	RMSE ↓	LPIPS ↓
Instruct NeRF2NeRF [32]	.098	.531	.0463	.0540
Instruct GS2GS [82]	.097	.519	*.0501*	*.0403*
GaussCtrl [88]	*.123*	*.590*	.0471	.0438
DGE [14]	.113	.565	.0384	.0407
Ours	**.175**	**.606**	**.0370**	**.0378**

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 [24]. Ours outperforms other novel-view stylization methods.

Marigold [43], and GeoWizard [25]. We apply Gaussian blur to these depth maps 75% of the time. We use noise scheduler strategies from [50].

Our ControlNet model is based on the original implementation [94]. We change the hint input channel size to accommodate the 5 channels of our composited RGBD image and associated 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 optimize 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 [92], 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 [20]. We compute dot-product and cosine-similarity losses on normal maps from both depth maps made using cross products on local image gradients [74].

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) [9] 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 [32], GaussCtrl [88], ScanNet++ [91], Mip-NeRF360 [3], and our new scenes. For all baselines, we use official code releases.

Evaluation Frame Selection Our method uses a smooth camera trajectory through the splat, whereas our baselines all 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 poses between our trajectory and the original training views used by our baselines. We interpolate halfway between every pair of views. We use these views for evaluating quantitatively and for the user study.

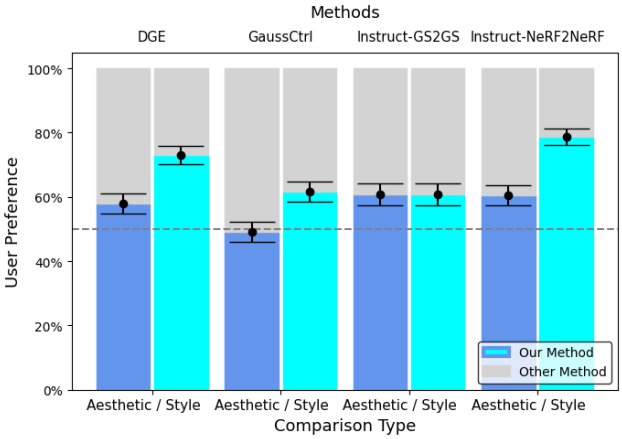

Figure 6. **A/B User Study** 31 participants consistently preferred our method's adherence to **style** prompts and found it to either match or exceed other methods in **aesthetic** quality.

Metrics We compute CLIP metrics defined in [6, 26, 32]. 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 [32, 88]. We experiment with varying the strength of our geometry stylization on both the intermediate RGBD diffusion model and the final 3DGS in Fig. 3.

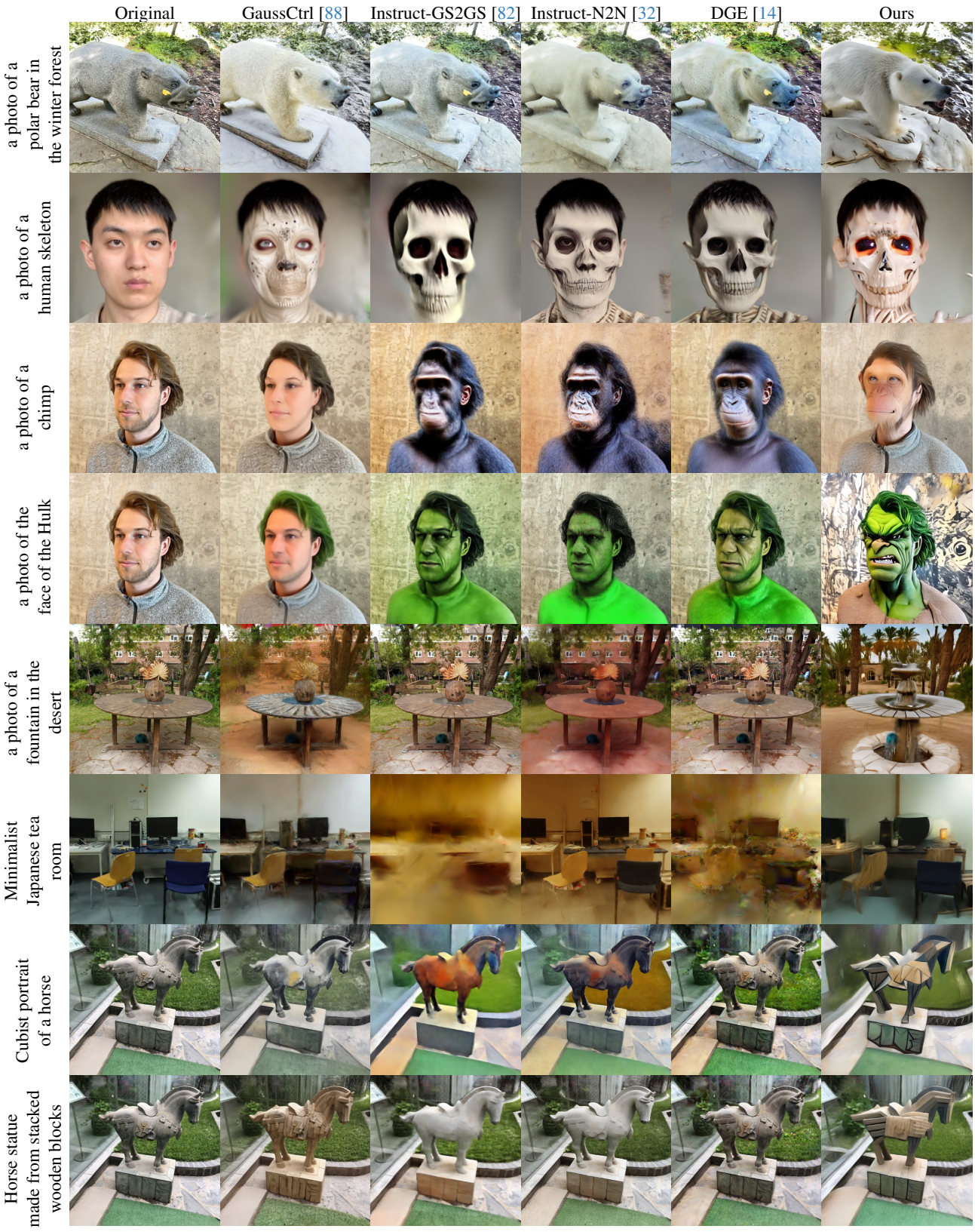

Figure 7. **Qualitative Comparison from Novel Views** Our method's ability to change the scene's shape allow its stylizations to be more aesthetically pleasing and exhibit more adherence to the style prompt.

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 [13] in Fig. 8. We show a side-by-side comparison of our model and baselines in Fig. 7; our model 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 [13] 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 [89]	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 novel-view synthesis model, we occasionally inherit errors and insert them into stylized output, see Figure 9.

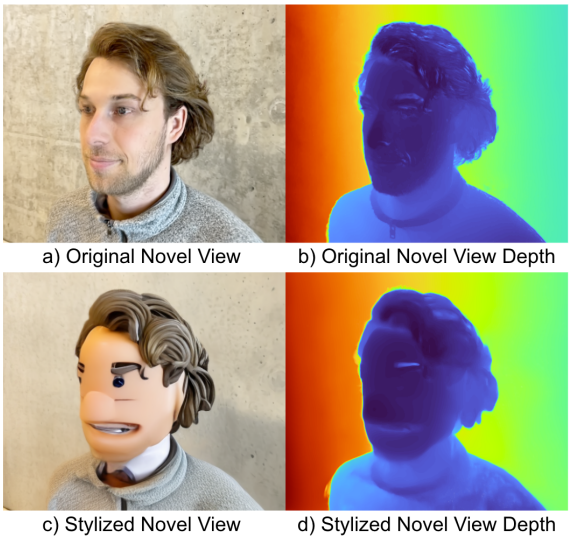

Figure 9. **Limitations** Given errors in the original 3D Gaussian Splat model (see the fuzziness on the right shoulder in a) and the incorrect depth on the eyebrow in b), our method will sometimes inherit these errors in the stylization.

7. Conclusion

We have presented a new method for 3D Gaussian Splat 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] Dubey Et al. The llama 3 herd of models, 2024. 20

[2] 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

[3] 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

[4] 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

[5] 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

[6] 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

[7] 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, 5

[8] 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

[9] 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

[10] 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

[11] 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

[12] 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

[13] 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, 8, 25

[14] Minghao Chen, Iro Laina, and Andrea Vedaldi. Dge: Direct gaussian 3d editing by consistent multi-view editing. 2024. 3, 5, 6, 15

[15] 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

[16] 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

[17] 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

[18] 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

[19] 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

[20] 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, 14

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

[22] 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. 14, 15

[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. ECCV, 2024. 3

[24] 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, 15

[25] Xiao Fu, Wei Yin, Mu Hu, Kaixuan Wang, Yuexin Ma, Ping Tan, Shaojie Shen, Dahua Lin, and Xiaoxiao Long. Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image. In *European Conference on Computer Vision*, pages 241–258. Springer, 2025. 6

[26] 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

[27] 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

[28] 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

[29] 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

[30] 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

[31] 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

[32] 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, 14, 15

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

[34] 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

[35] 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

[36] 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

[37] 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

[38] 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

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

[40] 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

[41] 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

[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 (CVPR)*, 2024. 5

[43] 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. 6

[44] 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

[45] 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

[46] 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

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

[48] 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

[49] 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

[50] 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. 6

[51] 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

[52] 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

[53] 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

[54] 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

[55] 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

[56] 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

[57] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023. 4

[58] 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

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

[60] 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

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

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

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

[64] 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

[65] 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

[66] 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

[67] 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

[68] 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

[69] 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

[70] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 4, 5, 14

[71] 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

[72] 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

[73] 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

[74] 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, 13, 14

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

[76] 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

[77] 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, 19

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

[79] 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

[80] 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

[81] 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. 5

[82] Cyrus Vachha and Ayaan Haque. Instruct-gs2gs: Editing 3d gaussian splats with instructions (2024). *URL https://instruct-gs2gs. github. io*. 6, 15

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

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

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

[86] 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

[87] 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

[88] 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, 5, 6, 14, 15

[89] 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

[90] 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

[91] 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

[92] 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, 14

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

[94] 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, 6

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

[96] 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

[97] 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

Supplemental Material

Contents

A. Note on Results

All results in the main paper, this supplemental, and the supplemental video are renders from our output stylized 3D Gaussian Splatting models unless stated otherwise.

We provide a supplemental video with extensive qualitative results and an overview of the paper.

B. Preliminaries

We use the phrase 'target frame' throughout this supplemental to refer to the frame currently being stylized. A 'reference frame' is a frame that is being warped to the target frame, or with which we do cross-attention while stylizing the target frame.

C. Splat regularization details

C.1. Normals smoothness prior

When training our stylized splats, we impose a TVL1 loss to encourage smoothness. To compute this loss, we generate a normal map $N_{i,j} \in \mathbb{R}^{H \times W \times 3}$ from a depth render of the 3D Gaussian Splatting model using cross products on local image gradients [74]. We define the loss function accumulated across pairs of adjacent pixels:

$$\mathcal{L}_{\text{TVL1}} = \sum_{i,j} \left(||N_{i+1,j} - N_{i,j}||_1 + ||N_{i,j+1} - N_{i,j}||_1 \right) \tag{2}$$

C.2. Supervised regularization

In addition to this smoothness prior, we also use supervised losses on the depths and normals when training splats. This requires ground-truth depth maps D^{gt} and normal maps N^{gt} for each training view. For the initial splat, we get D^{gt} and

N^{gt} for each frame from Metric3D [92] as discussed in the main paper. For the final stylized splat, we use our stylized depth maps as D^{gt} and obtain N^{gt} from D^{gt} using cross products on local image gradients [74].

We compute the scale-invariant depth-loss introduced by [20] between the rendered depth D and the ground-truth D^{gt}. We refer to this loss term as $\mathcal{L}_D$.

As discussed in C.1, we also obtain a normals image from the splat. In addition to the smoothness prior introduced in that section, we also supervise those normals using the ground-truth, N^{gt}, with a dot-product loss (which is equivalent to the cosine distance):

$$\mathcal{L}_{\text{N}} = -\sum_{i,j} \left(N^{\text{gt}}_{i,j} \cdot N_{i,j} \right) \tag{3}$$

Certain scenes, such as Face (from Instruct-NeRF2NeRF [32]) and Fangzhou (from GaussCtrl [88]) still exhibit artifacts such as shimmering and popping due to the people moving during the capture.

C.3. Hyperparameters

The total loss when training splats is a sum of the above terms, plus the usual photometric loss $\mathcal{L}_{\text{phot}}$:

$$\mathcal{L} = \mathcal{L}_{\text{phot}} + \lambda_{\text{TVL1}}\mathcal{L}_{\text{TVL1}} + \lambda_{\text{N}}\mathcal{L}_{\text{N}} + \lambda_{\text{D}}\mathcal{L}_{\text{D}} \tag{4}$$

While training our stylized splats, we wait for 3000 training steps until enabling our regularization terms, after which we use the following parameters for all scenes: $\lambda_{\text{TVL1}} = 0.05$, $\lambda_{\text{N}} = 0.001$, and $\lambda_{\text{D}} = 0.5$. When training the initial splats which we use as inputs for both our model and our baselines, we vary these parameters for different scenes to obtain higher-quality splats.

C.4. Ablation

We investigate the impact of the three regularization terms introduced above by switching them all off and recomputing our metrics. We show our scores table in ref. A1. On the CLIP similarity and consistency metrics, our method remains the best, even without these regularisation terms. The increased cloudiness of the geometry unsurprisingly does cause a worsening in the RMSE and LPIPS scores.

(We do not include our sequential metrics in this table because they are computed on the stylized images that our pipeline generates rather than from the splat, and therefore they are unaffected by this ablation.)

D. Video Diffusion Baseline

Video diffusion models have shown great promise for consistent frame-to-frame generation and editing. However, they cannot stylize NVS models on their own. For evaluation, we take an off-the-shelf video editing model, CCEdit [22], and use it to stylize frames within our pipeline. We report scores in Table A1 and a qualitative comparison in Fig. A1.

Figure A1. Here, we present output samples from our test set generated using both CCEdit [22] and our method. The results demonstrate that our approach achieves significantly better stylization.

By default, CCEdit uses a stylized reference frame as guidance when editing the rest of a video sequence. We use StableDiffusion 2.1 [70] to generate the stylized reference using their parameters and the prompt our method uses.

CCEdit is limited to generating a sequence of 17 frames at a time. While we could limit 3DGS renders to just 17 frames, CCEdit cannot handle large frame-to-frame changes given large scenes with large camera baselines. To remedy this, we take inspiration from CCEdit's description for handling longer sequences. We start with the exact same rendered frames we use in our pipeline. We pass chunks of 17 overlapping frames through CCEdit. The last stylized frame in each chunk serves as the style guidance frame in the midpoint of the next chunk. We continue this until all frames are exhausted. We pass the stylized frames through the retraining phase of our pipeline.

	CLIP		Image Consistency	
	CLIP Direction Similarity ↑	CLIP Direction Consistency ↑	RMSE ↓	LPIPS ↓
Instruct NeRF2NeRF [32]	.098	.531	.0463	.0540
Instruct GS2GS [82]	.097	.519	.0501	*.0403*
GaussCtrl [88]	.123	.590	.0471	.0438
DGE [14]	.113	.565	*.0384*	.0407
CCEdit [22] w/ our pipeline	.135	**.629**	.0915	.0581
Ours w/o resplat reg	*.171*	.606	.0450	.0454
Ours	**.175**	.606	**.0370**	**.0378**

Table A1. **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 [24]. Ours outperforms other novel-view stylization methods.

CCEdit underperforms compared to our approach across all metrics, except for CLIP Direction Consistency. The subjective quality of the CCEdit results is poor, as indicated in fig. A1.

E. Evaluation Metrics

For the sequential RMSE and sequential LPIPS metrics, we use our mesh warper to warp one stylized frame from our pipeline to the next. In the case of RMSE, we warp the RGB image; for LPIPS, we use a pretrained VGG16 network to extract features, normalize them along the channel dimension, and then warp those features. In both cases we measure the inter-frame consistency by computing the RMSE between the warped previous frame and the next frame, excluding any pixels which lie outside the valid region of the warp.

Mesh warping inherently creates artifacts and so these metrics inherit them. This explains why our 'Warp + RGB Inpaint + Depth ControlNet' ablation in Table 2 outperforms our full pipeline on the Seq. RMSE metric: it is rewarded by the RMSE metric for not correcting the warping artifacts. Because warping artifacts tend to look perceptually unlike the original image, this ablation still underperforms our full method quite substantially on the sequential LPIPS metric. This suggests that sequential LPIPS may be a better measure of inter-frame consistency, because it is less likely to inherit issues with warping artifacts from the underlying warping method.

Our method outperforms all of our competitors on the CLIP similarity and consistency metrics (Table 1), and this aligns with our qualitative figures (Figures 7 and A7), in which our method generally adheres better to the prompt. However, Table 2 indicates that our 'without feature sharing' and 'full x-attn, no feature injection' ablations sometimes outperform our full pipeline on the CLIP metrics. We attribute this finding to a trade-off between *per-frame prompt adherence* and *cross-frame consistency*: stylizing a frame conditional on some previously stylized frames is a more constrained problem then simply stylizing the frame on its own. This likely accounts for why these ablations outperform our full method on CLIP similarity but underperform it on sequential RMSE and sequential LPIPS: they are less constrained by previous frames and therefore more free to align with the prompt. However, since we train a splat on the stylized frames, we prefer the 'more consistency' side of this trade-off, to avoid blurriness, floaters and view-dependent artifacts in the final splat.

F. Compositing details

As discussed in the main paper, we form composites for each reference frame by converting the RGBD frame to a mesh, rendering it from the view of a target frame to be stylized, and compositing them together. For each pixel inside the valid region of the warp, we must choose whether to use either the warped reference frame or the unstylized new frame. We do this by computing a compositing score $S_{i,j}$ for each pixel:

$$S_{i,j} = \lambda_1 \left|\sin(\theta_{i,j})\right| - \lambda_2 \frac{d'_{i,j}}{d_{i,j}} + \lambda_3 \left(\text{idx} - \text{idx}'\right) \tag{5}$$

In the first term above, $\theta_{i,j}$ is the angle between the mesh normal at target-frame pixel i, j and the camera look-at vector of the *reference* frame (even though we are compositing in the target frame). This term rewards geometry that is viewed face-on in the reference frame (and is therefore less likely to have warping artifacts).

In the second term, $d'_{i,j}$ is the depth of pixel i, j in the warped stylized reference frame, and $d_{i,j}$ is the depth of that same pixel in the unstylized target frame. This is an occlusion term which tends to favour whichever frame is occluding the other.

In the final term, idx and idx' are the indices of the target and reference frame respectively within our camera trajectory. This term tends to favour reference frames that are older, because earlier reference frames are less likely to contain artifacts arising from repeated warping. Note that because of the autoregressive nature of our pipeline, $\text{idx} - \text{idx}'$ is always positive.

The three λ parameters are weights for each of these factors. We use the same values everywhere: $\lambda_1 = 1.0$, $\lambda_2 = 3.0$, and $\lambda_3 = 0.02$.

G. Full list of prompts

We list all scenes and prompts used for the evaluation and user study in Table A2, A3 and A4. For reproducibility purposes, we include information on the positive, negative, and inversion prompts together with guidance scale, color, and depth strengths.

H. Fixed depth noise vs RGB noise

In the main paper we showed our RGBD model's ability to change depths for varying depth stylization while preserving the the overall color gamut. In Figure A3 we show the opposite and demonstrate our model's ability to change the colors for varying color stylization without considerably changing geometry.

I. Cross Attention implementation details

I.1. Construction of the heatmaps

Our depth-guided attention is based on heatmaps $L_{i,j}$ encoding whether a token i from the reference image forward-warps to the location of token j in the target image. We perform a forward-warp from the reference frame to the target frame by converting the reference RGBD image to a mesh and rendering it into the target frame, as discussed in the main paper. Using this we construct a flow field encoding how a reference-frame pixel (u, v) maps to a target-frame pixel (u', v') under the forward warp:

$$\mathbf{F} \in \mathbb{R}^{H \times W \times 2}, \quad \mathbf{F}_{u,v} = (u', v') \tag{6}$$

We then expand this flow field into a 4D tensor:

$$A_{u,v,u',v'} = \delta \left(\text{round} \left(\mathbf{F}^T_{u,v} \right) - \begin{pmatrix} u' \\ v' \end{pmatrix} \right) \tag{7}$$

This tensor $A_{u,v,u',v'}$ will be unity only where the reference-frame coordinates (u, v) map to the target-frame coordinates (u', v') under the forward-warp, and zero elsewhere.

Finally, we blur this tensor along the spatial axes of the reference. This will have the effect of causing a target-frame feature to attend not just to the reference feature that warps to it, but also to all other features in its immediate vicinity in the reference image. We do this by convolving with a blur kernel:

$$L_{u,v,u',v'} = A_{u,v,u',v'} * K \tag{8}$$

where $*$ is the convolution operator (which we apply only along the u and v axes, i.e. the reference axes). For the blur kernel K, we choose a simple linear decay towards zero:

$$K(\Delta u, \Delta v) = \max \left(1 - \frac{||\Delta u, \Delta v||_2}{d_{\text{max}}}, 0 \right), \tag{9}$$

where d_{max} is the number of pixels over which the attention decays to zero. We use $d_{\text{max}} = 100\,\text{px}$ everywhere. The heatmap L now encodes our desire that target-frame pixels should attend only to reference-frame pixels that map to them. In practice we then clip this so that we always have nonzero attention with all reference-frame features:

$$L_{u,v,u',v'} \leftarrow \max \left(L_{u,v,u',v'}, L_{\text{min}} \right) \tag{10}$$

This allows the target frame to be stylistically guided by the reference frame even in areas of the target frame that were not visible from the reference. The constant L_{min} is a hyperparameter controlling the locality of the cross-attention; when L_{min} is unity, we attend equally to the entire reference frame, whereas in the limit $L_{\text{min}} \to 0$ we completely disregard reference-frame features that do not warp to a given target-frame feature's location. We choose $L_{\text{min}} = 0.5$ for all forward-facing scenes and $L_{\text{min}} = 0.3$ for all other scenes.

Since the attention layers inside the diffusion U-Net operate at downscaled resolutions, we must also downscale $L_{i,j}$. We do this by reshaping it to a 4D tensor indexed by pixel coordinates u, v in the reference frame and u', v' in the target frame, $L_{u,v,u',v'}$ (as opposed to a 2D tensor indexed by reference- and target-frame tokens). Then we max-pool across the u and v dimensions, and then – in a separate max-pool – across the u' and v' dimensions. This allows us to downscale the entire tensor across the spatial axes. We then reshape back to a 2D tensor $L'_{i,j}$ of the correct dimensionality for the downscaled feature maps on which the attention layer acts.

In the following sections we describe how in practice we use the above matrices to control the attention layers inside the diffusion U-Net.

I.2. Depth-conditioned cross-attention

As described in the main paper (eq. 1), we modify the U-Net's cross-attention layers so that the keys (and values) are the concatenation of the target-frame keys with all of the reference-frame keys (for however many reference frames we have). We then construct the heatmap $L_{i,j}$ as described in the previous section, and it enters the mask Δ in eq. 1 of the main paper as described in the paragraph following that equation. We modify all self-attention layers of the RGBD diffusion model U-Net in this way, though we leave the ControlNet unchanged.

I.3. Feature injection

Feature injection is a more direct means of sharing features between reference and target images. Here we simply mix

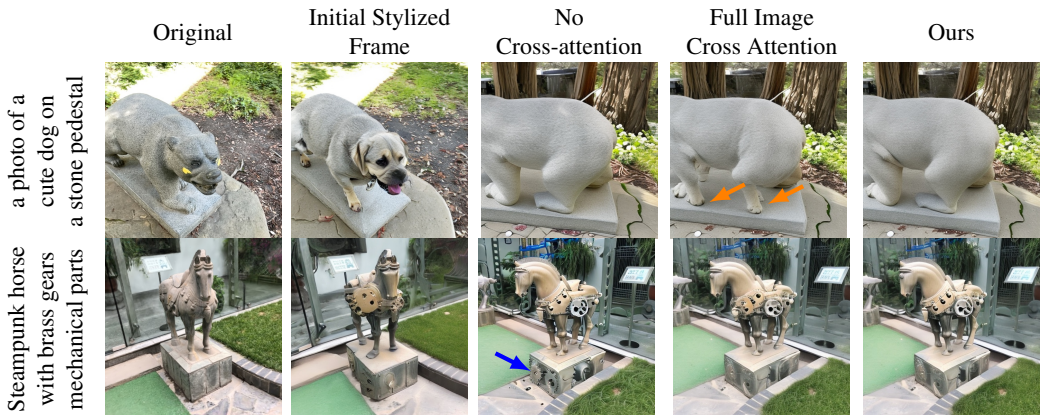

Figure A2. **Qualitative Comparison from Novel Views for Ablations** Here we ablate our feature-sharing contributions. With no cross-attention, the cogs are copied over from the body of the horse onto the pedestal's base (blue arrow). With full-image cross-attention (without our depth-informed heatmaps), the legs of the bear point the wrong way and are misshaped (orange arrows).

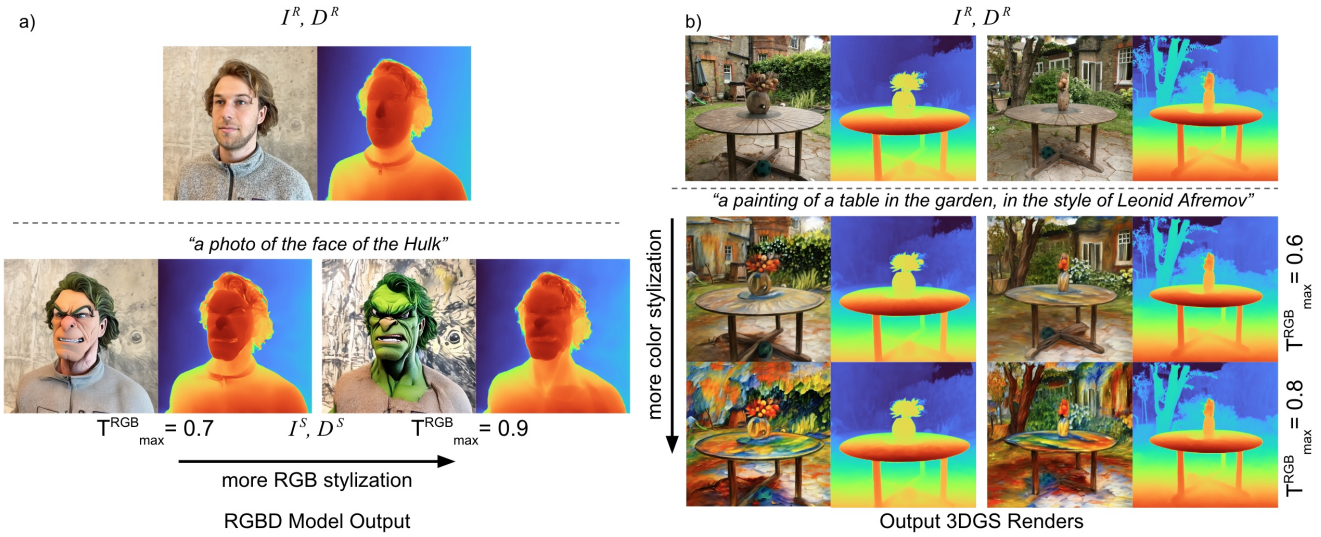

Figure A3. For the same prompt, we vary stylization strength for RGB. a) we show the output of our RGBD model for the same stylization prompt but with varying RGB stylization strengths. b) We show the effect of color stylization in output 3DGS models from our method.

in reference features directly, rather than having the target frame attend to them. This mixing is performed on the final hidden states after each depth-conditioned cross-attention operation.

We first compute a mixing matrix, M:

$$M_{i,j} = \text{softmax}_i\left(\frac{L_{i,j}}{T}\right) \tag{11}$$

This softmax increases the 'locality' of the mixing matrix, suppressing elements that were not near unity. It also ensures normalization over the reference-frame axis. This is important because we wish to take weighted averages of reference- and target-frame features, not sums. We choose a very low value for the temperature T of 0.001, which effectively turns this into an argmax operation. We do this because we only want to inject reference features that warp directly to a given target feature; otherwise the result tends to be somewhat blurry.

We define a weight for injection into a given target hidden state j as follows:

$$w_j = \max_i L_{i,j}\lambda_{\text{inject}}, \tag{12}$$

where the constant λ_{inject} is a hyperparameter, which we set to 0.15 for all forward-facing scenes and 0.2 for all other scenes. This weight allows us to suppress feature injection in cases where no reference-frame features warp to a given target-frame feature.

Then we use the mixing matrix M to inject reference-frame hidden states into the target-frame hidden states, with

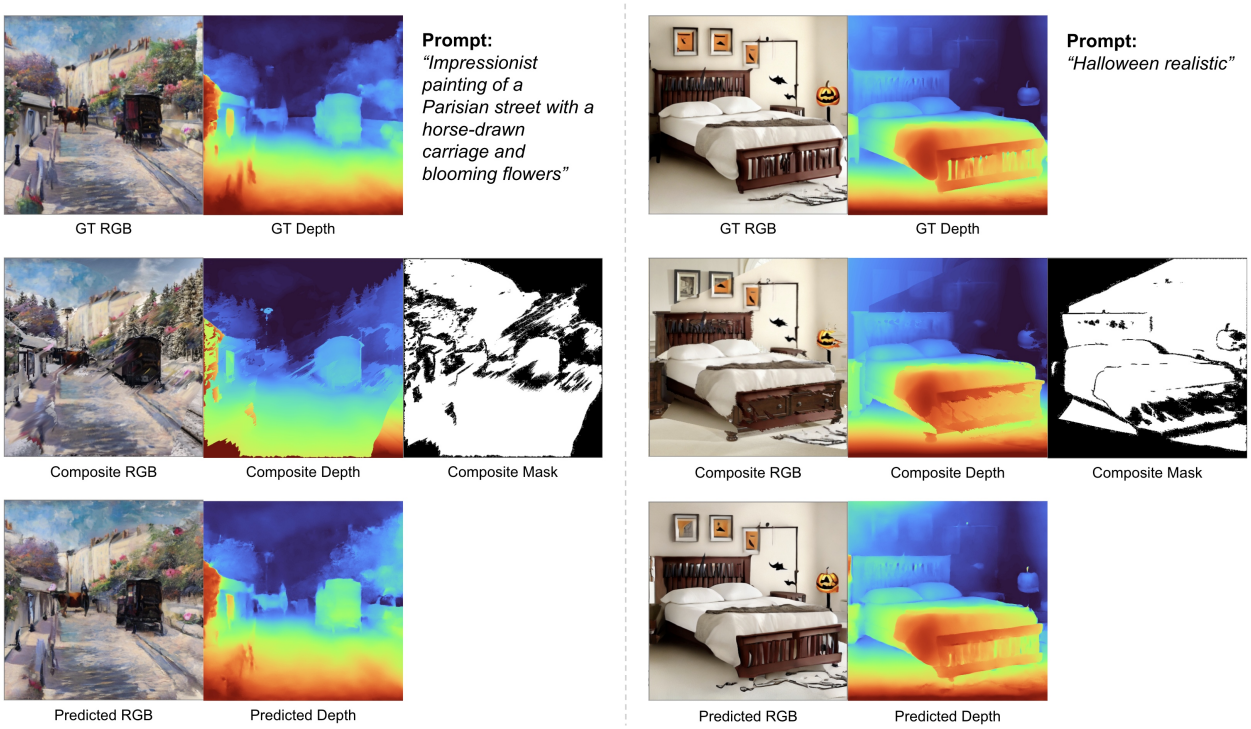

Figure A4. **ControlNet training samples. Top row** depicts the stylized RGBD images as well as the prompts used to stylize the RGBD images; These are the images we are training the ControlNet to predict. **Middle row** shows the composite RGBD images and composite masks we give the ControlNet as conditioning. Notice the warping artifacts in the composite images. **Bottom row** shows the predicted RGBD image from the ControlNet for the given conditioning images and noised target images. The model successfully preserves the warped pixels while correcting warping artifacts.

the amount of mixing per hidden state weighted by w_j:

$$h'_j = (1 - w_j)h_j + w_j \sum_j h_i^{\text{ref}} M_{ij} \tag{13}$$

J. RGBD Diffusion Model details

The RGBD diffusion model stylizes RGB and disparity maps. We use disparity maps as opposed to depth since disparity has higher fidelity closer to the camera.

J.1. Training the RGBD model for separate controls over geometry and appearance

To train our RGBD model and our Warp ControlNet for separate control over geometry and appearance, we introduce time parameters $T_{\text{max}}^{\text{RGB}}$ and $T_{\text{max}}^{\text{D}}$ to independently control the amount of noise applied to the RGB and depth channels; these are defined relative to the maximum training timestep, and so vary from zero (no noise) to one (pure Gaussian noise). We also add two new channels to the UNet input to receive masks that inform the model of whether its predicted step will be discarded or retained for each of the RGB and depth channels. This is similar to the way that inpainting models are trained, except that our masks do not vary across the spatial dimensions of the image. We train the model as follows:

1. Randomly select $T_{\text{max}}^{\text{RGB}}$ and $T_{\text{max}}^{\text{D}}$.
2. Noise the RGB channels up to $T_{\text{max}}^{\text{RGB}}$, and noise the depth channels up to $T_{\text{max}}^{\text{D}}$.
3. Generate two binary masks: RGB mask, where the value is 0 if $T_{\text{max}}^{\text{RGB}} < t$ and 1 otherwise (a value of zero indicating updates to RGB should be ignored at this point), and a depth mask, which is 0 if $T_{\text{max}}^{\text{D}} < t$ and 1 otherwise.
4. Pass these masks into the U-Net as part of the input. This way, the model knows which updates will be discarded, allowing it to condition its processing accordingly.
5. Run the U-Net as usual and compute the loss based on the denoising tasks it performs. Training is guided by standard supervision losses.

Half of the time we randomly sample values $T_{\text{max}}^{\text{RGB}}$ and $T_{\text{max}}^{\text{D}}$ independently from a uniform distribution between 0 and 1. For the other half of the time, we choose $T_{\text{max}}^{\text{RGB}} = T_{\text{max}}^{\text{D}}$.

J.2. Training hyper-parameters

We train our RGBD model for 30k steps, batch size 4 and accumulation of 8 batches, for an effective batch size of 32. We use an exponential learning rate scheduler which updates at every step. We use a learning rate of 3e-5 with a warmup of 100 steps, decaying to 3e-7 by 25k steps.

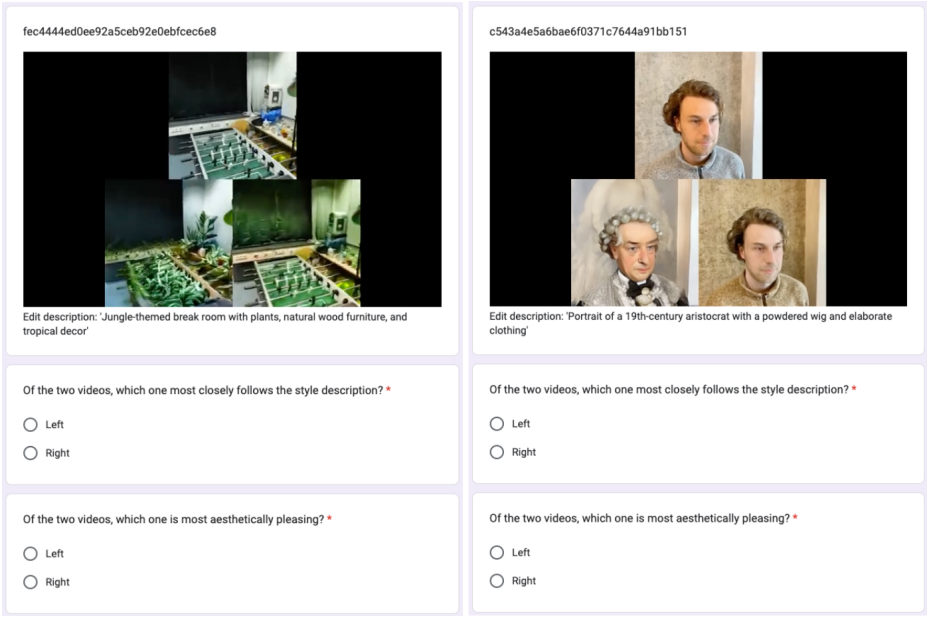

Figure A5. **User Study Prompt** Sample A/B video and associated questions from our user study. The original render is on top, and two methods are on the bottom. The method order is randomized.

J.3. Inference

Congruently with our training regime, at inference time we discard the model's RGB updates when $T^{\text{RGB}}_{\text{max}} < t$, and we discard its depth updates when $T^{\text{D}}_{\text{max}} < t$. This allows us to use $T^{\text{RGB}}_{\text{max}}$ and $T^{\text{D}}_{\text{max}}$ to control the extent to which RGB and depth are edited.

K. Warp ControlNet training details

The Warp ControlNet is conditioned on the composite of the target frame and a warped stylized reference frame alongside the composite mask, and then guides the RGBD diffusion model when stylizing the target frame.

K.1. RGBD training image composites

As discussed in the main paper, our Warp ControlNet takes a composite image containing a stylized region warped from a previous frame and an unstylized region from the new frame. Its task is: 1) to stylize the unstylized region consistently with the stylized region, and 2) to correct warping artifacts within the stylized region that were introduced by the warp from the previous frame. Here we expand on the procedure by which we train the Warp ControlNet for this task.

To train the ControlNet, we need a dataset of composite images. We generate these composites as follows:

1. Sample an image I from the Re-LAION-2B [77] dataset.
2. Generate a monocular depth map D for the image with an off-the-shelf detector.
3. Stylize the image with a random prompt using our RGBD diffusion model (with no ControlNet), producing a stylized image I_S and depth map D_S.
4. Forward-warp I_S and D_S to a randomly chosen new camera view using the generated depth map, producing warped stylized RGBD I'_S and D'_S.
5. Forward-warp I'_S and D'_S back to the original view, producing doubly-warped RGB I''_S and depth D''_S.
6. Composite together I''_S and D''_S with I and D to produce a new RGB I_C and depth D_C.

Fig. A4 shows example images from this process. The top row of the figure shows stylized RGBDs I_S and D_S generated by step (3) above. The second row shows the final composite, I_C and D_C, arising from step (6).

One can see from Fig. A4 that unlike the original I_S and D_S, the doubly warped I''_S and D''_S contain both warping artifacts and gaps where stylized information is not available. By training the ControlNet to try and reconstruct the original stylized RGBD I_S and D_S by using I''_S and D''_S as input, the ControlNet learns to fix warping artifacts and to stylize newly visible areas of the scene consistently with the previous frame. This is exactly what we need for our task of stylizing a new view consistently with some previously stylized view of the same scene. Moreover, even though this is a multiview task, this method of generating the training data means we can learn to solve this task using only monocular datasets such as Re-LAION-2B.

When compositing depth, we need to scale the depths of D''_S to match the original depth map D, because our RGBD model only predicts depths up to scale. We compute the scale factor using only pixels that lie near the boundaries between the stylized and unstylized regions in the composited depth map, to minimize the extent of any depth dis-

continuities between stylized and unstylized regions. One can still see seams between D and D'_S in the composites on Fig. A4 (which the 'Predicted Depth' images show that the ControlNet learns to fix), but this approach helps to reduce their extent.

K.2. Stylization prompts

The stylization prompts for the RGBD training pairs were obtained by querying Llama-3.1-405b-instruct [1], we share some example prompts below:

1. "*Ethereal, mystical landscape with glowing, luminescent plants*"
2. "*A futuristic cyberpunk cityscape at dusk*"
3. "*Victorian-era style illustration of a fantastical steampunk airship soaring above the clouds*"

K.3. Training hyper-parameters

We train our Warp ControlNet for 180k steps, batch size 6 and accumulation of 2 batches for effective batch size of 12. We use a constant learning rate of 1e-4.

L. Details of inversion

When stylizing a new frame, we begin by inverting that frame to noise using DDIM inversion. This leads to more consistent results because, unlike simply adding Gaussian noise, it does not destroy the information content of the original RGBD image. During inversion we condition on a prompt describing the content of the unstylized scene (which we refer to elsewhere as the 'inversion prompt'), although one can use an empty prompt with little impact on the quality of the results.

We use a standard procedure for DDIM inversion, based on the approximation that the steps predicted by the diffusion model will be similar to latents at successive timesteps: $\epsilon(x_{t+1}) \approx \epsilon(x_t)$. This then leads to a simple DDIM inversion scheme in which one inverts an image to noise by repeatedly obtaining a step from the diffusion model and then stepping in the opposite direction.

In practice, we find that, owing to our use of depth models at training time, our model is somewhat sensitive to the high-frequency characteristics of the RGBD frame which is used as input, leading to visual artifacts in cases where splat artifacts are present in the input. To improve robustness to these high-frequency characteristics, we instead perform a 'partial inversion' in which we first add noise to the input RGBD to get to some noise level T_{noise}, and then use the above-mentioned inversion procedure for the remaining steps. This allows us to destroy the high-frequency components in the first few steps, avoiding the issue with the model's sensitivity to them. We use a relatively conservative amount of random noise, $T_{\text{noise}} = 0.05$, throughout, which ensures that we do not destroy too much of the lower-frequency signal in the depth map.

For the case where $T^{\text{RGB}}_{\text{max}} \neq T^{\text{D}}_{\text{max}}$, we adapt the inversion as follows: if $T^{\text{RGB}}_{\text{max}} > T^{\text{D}}_{\text{max}}$, we perform partial inversion as usual up until $T^{\text{D}}_{\text{max}}$ is reached, after which we continue inverting but discard the updates to the depth channels until $T^{\text{RGB}}_{\text{max}}$ is reached. If $T^{\text{RGB}}_{\text{max}} < T^{\text{D}}_{\text{max}}$, we do the converse.

We note that the relatively non-destructive inversion procedure we use often has the desirable result of tending to preserve view-dependent effects, such as reflections, from the original frame.

M. Details on the user study

We sampled views for the user study by selecting a random selection from the prompts in our evaluation set (see Table A2), and then – for each prompt – selecting a random evaluation view. In some cases there are evaluation views that are particularly uninteresting, or which give poor coverage of the scene. For example, in the Scannet++ scenes, some of the evaluation views just show a wall at close range. In these cases, we manually eliminate the views from the user study.

For each evaluation view selected for the study, we rendered a circular trajectory in the camera plane centred on that view, which helps to convey a sense of the geometry of the scene from that evaluation view.

The participants in the user study were presented with three splat renders for the circular trajectory: the original splat, our stylized splat, and the stylized splat from one of our baselines. They were asked to indicate preferences between the two stylized splats for prompt adherence and for aesthetic value. A screenshot of the questions for two videos is shown in Figure A5. Our study had 31 participants, and contained 32 questions.

N. Details of our train view trajectories

The 3DGS models that we use as an input to our method are trained on all camera views available in the source dataset. However, those views are usually an independent set of photos with corresponding poses that do not follow any smooth camera motion. As our method relies on a continuous trajectory, we create a new smooth sequence of camera views that pass through the original set of training views. We visualize an example of such a trajectory in Fig. A6(a).

O. Further details on evaluation views

As discussed in the main paper, we select evaluation views that are fair to both our method (which uses smooth camera trajectories) and our baselines (which use the original training views of the splat) by choosing a view from each set and then interpolating halfway between them.

Half of the time we select a view from our trajectory and then interpolate halfway to the nearest original training view; the other half of the time we do the reverse, selecting

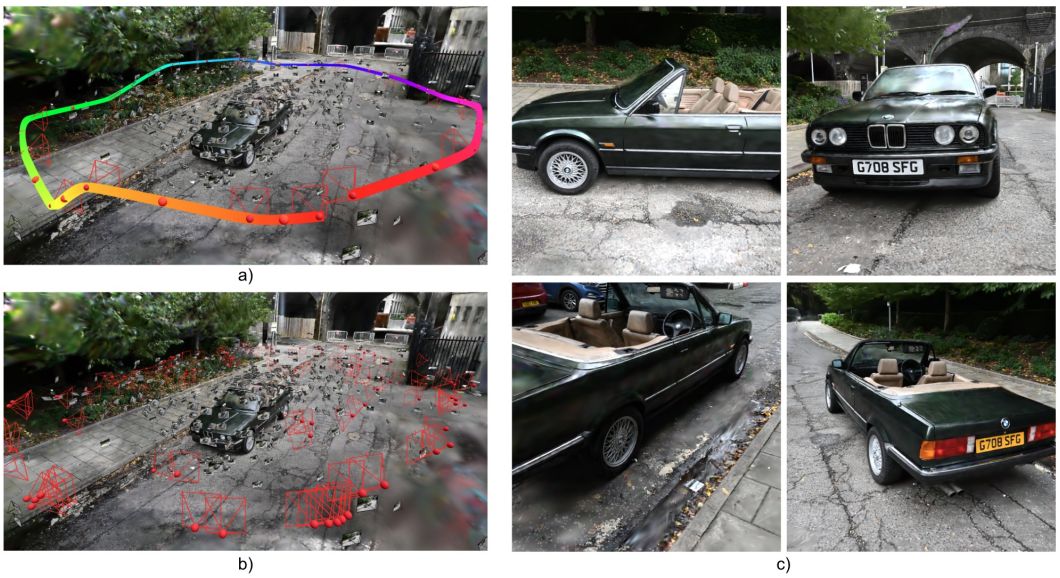

Figure A6. **Training and evaluation camera trajectories** a) Smooth camera trajectory used by our method together with set of camera views used for the input 3DGS model training b) All camera views used for evaluation of our method on this scene c) Example rendered views used for the evaluation

one of the original training views and interpolating halfway to the nearest view in our trajectories.

We linearly interpolate the camera positions and slerp the camera orientations. Sometimes a pair of views will have the cameras looking in very different directions, and then interpolating them may lead to an evaluation view that does not look somewhere sensible. For this reason we reject pairs of views whose camera orientations differ by more than $90°$.

We limit the number of evaluation views to at most 100, in order to avoid excessive evaluation runtimes. We select the evaluation views once per scene and use them everywhere, across all prompts for that scene.

P. Further qualitative results

We show additional qualitative results in Figure A7 and Figure A8.

Q. Qualitative results of ablations

We give further qualitative results for our ablations in fig. A9, showing the impact of our cross-attention contributions.

R. Runtime

We use an Nvidia A100 40GB to run our pipeline. On this hardware our method takes around 15 minutes to run on a typical scene, consisting of about 10 minutes for stylization and 5 minutes for training the stylized splat. These timings were measured with FP16 precision for inference, which we find produces no discernible decrease in quality.

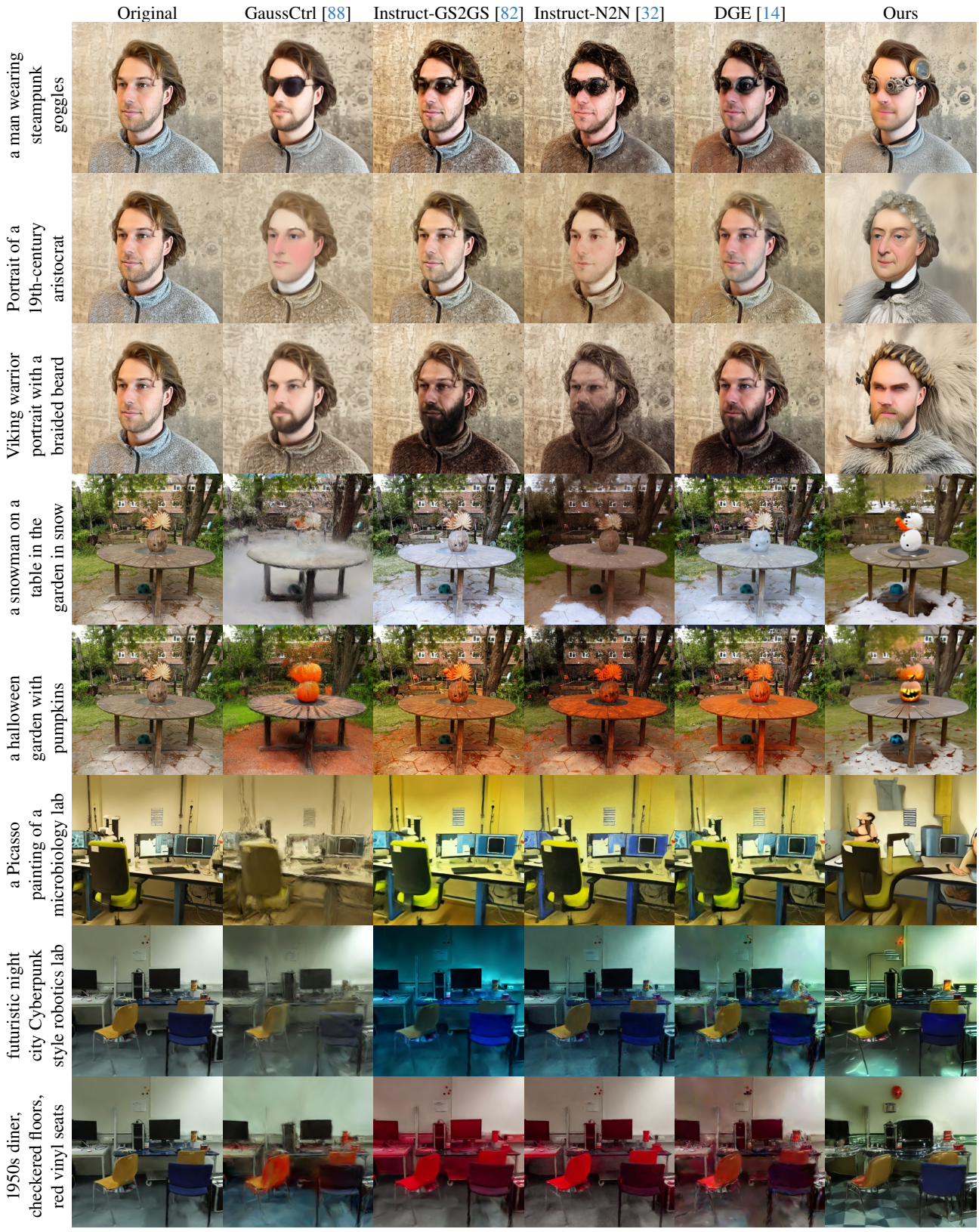

Figure A7. **Further Qualitative Comparison from Novel Views** Our method's ability to change the scene's shape allow its stylizations to be more aesthetically pleasing and exhibit more adherence to the style prompt.

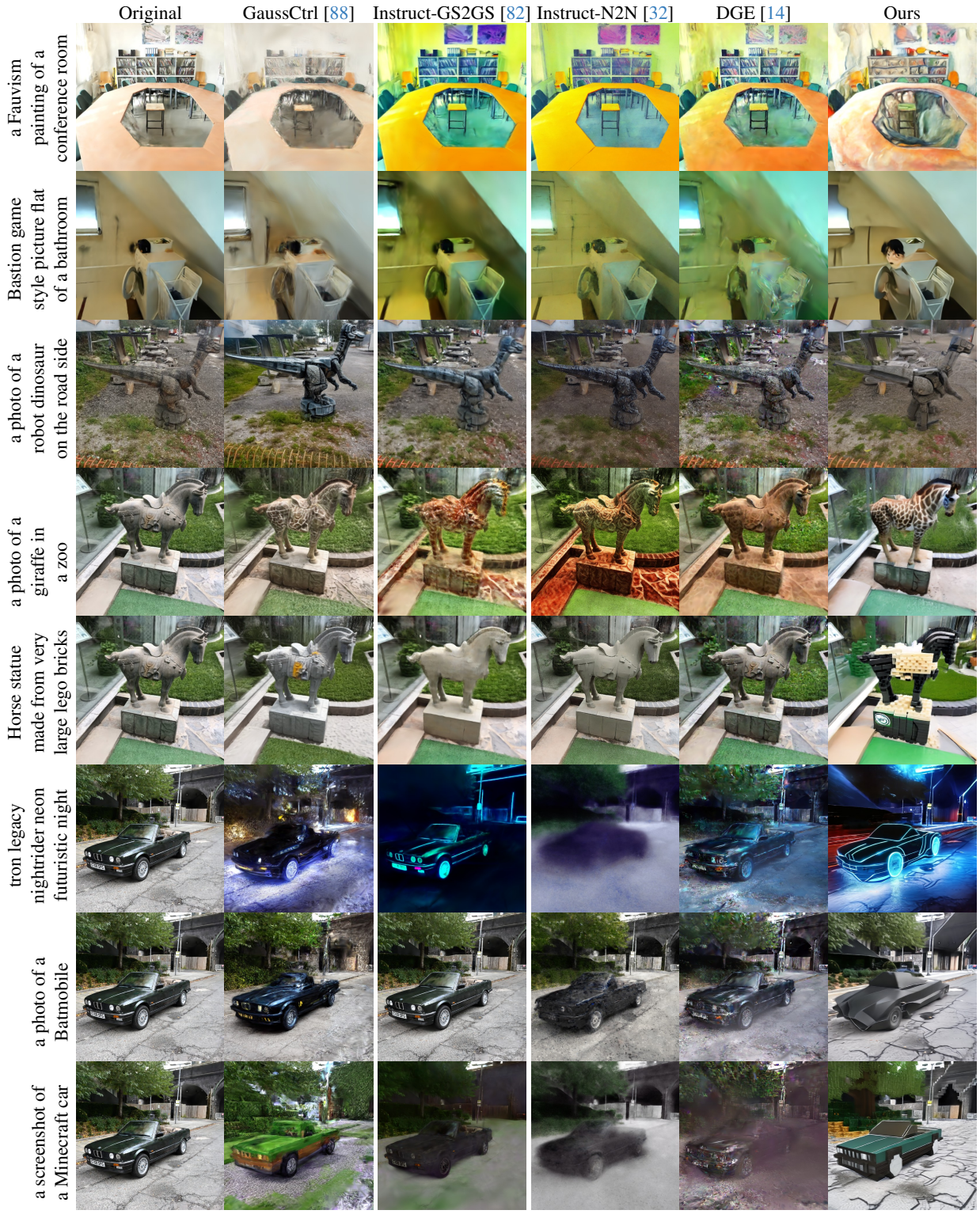

Figure A8. **Further Qualitative Comparison from Novel Views** Our method's ability to change the scene's shape allow its stylizations to be more aesthetically pleasing and exhibit more adherence to the style prompt.

Dataset	Scene Name	Positive Prompt	Negative Prompt	Inversion Prompt	Guidance Scale	Color Strength	Depth Strength	Used in the User Study
BlendedMVS	Dinosaur	"Photo of a robot dinosaur"	"a photo of a dinosaur statue on the road side lowres, grainy, blurry"	"a photo of a dinosaur statue on the road side"	7.50	0.50	0.40	55 No
		"Photo of an ostrich"	"a photo of a dinosaur statue on the road side lowres, grainy, blurry"	"a photo of a dinosaur statue on the road side"	5.00	0.50	0.30	51 Yes
		"a photo of a giraffe on a desert"	"a photo of a dinosaur statue on the road side lowres, grainy, blurry"	"a photo of a dinosaur statue on the road side"	5.00	0.50	0.30	51 Yes
GaussCtrl	Fangzhou	"a photo of a face of a man with a moustache"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	7.50	0.70	0.70	55 No
		"a photo of a face of an old man with wrinkles"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	7.50	0.70	0.70	55 No
		"a photo of a human skeleton"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	10.00	0.80	0.80	51 Yes
		"a photo of a man wearing a pair of glasses"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	7.50	0.70	0.70	55 No
		"a photo of a man with a hat on"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	7.50	0.60	0.80	55 No
	Stone Horse	"Cubist portrait of a horse"	"a photo of a stone horse in front of the museum lowres, grainy, blurry"	"a photo of a stone horse in front of the museum"	7.50	0.60	0.40	55 No
		"Horse statue made from stacked wooden blocks, rustic and minimalistic"	"a photo of a stone horse in front of the museum lowres, grainy, blurry"	"a photo of a stone horse in front of the museum"	5.00	0.60	0.40	51 Yes
		"Horse statue made from very large lego bricks in front of a museum"	"a photo of a stone horse in front of the museum lowres, grainy, blurry"	"a photo of a stone horse in front of the museum"	10.00	0.70	0.30	55 No
		"Photo of a dinosaur"	"a photo of a stone horse in front of the museum lowres, grainy, blurry"	"a photo of a stone horse in front of the museum"	7.50	0.60	0.40	55 No
		"Steampunk horse statue with brass gears, pipes, and mechanical parts"	"a photo of a stone horse in front of the museum lowres, grainy, blurry"	"a photo of a stone horse in front of the museum"	5.00	0.60	0.40	55 No
		"a photo of a cyberpunk sci robot horse"	"a photo of a stone horse in front of the museum lowres, grainy, blurry"	"a photo of a stone horse in front of the museum"	5.00	0.60	0.40	51 Yes
		"a photo of a giraffe in zoo"	"a photo of a stone horse in front of the museum lowres, grainy, blurry"	"a photo of a stone horse in front of the museum"	7.50	0.65	0.65	51 Yes
		"a photo of a medieval horse in front of the castle"	"a photo of a stone horse in front of the museum lowres, grainy, blurry"	"a photo of a stone horse in front of the museum"	10.00	0.60	0.40	51 Yes
Instruct Nerf2Nerf	Bear	"a photo of a hippopotamus next to a river"	"a photo of a bear statue in the forest lowres, grainy, blurry"	"a photo of a bear statue in the forest"	7.50	0.60	0.35	55 No
		"a photo of a panda in the bamboo forest"	"a photo of a bear statue in the forest lowres, grainy, blurry"	"a photo of a bear statue in the forest"	7.50	0.50	0.50	51 Yes
		"a photo of a polar bear in the winter forest"	"a photo of a bear statue in the forest lowres, grainy, blurry"	"a photo of a bear statue in the forest"	5.00	0.70	0.10	51 Yes
		"a photo of a rhino"	"a photo of a bear statue in the forest lowres, grainy, blurry"	"a photo of a bear statue in the forest"	4.50	0.70	0.60	51 Yes
	Face	"Portrait of a 19th-century aristocrat with a powdered wig and elaborate clothing"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	7.50	0.75	0.60	51 Yes
		"Viking warrior portrait with a braided beard, fur cloak, and a fierce expression"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	7.50	0.75	0.75	51 Yes
		"a photo of a chimp"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	5.00	0.50	0.70	51 Yes
		"a photo of a face of an old man with wrinkles"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	5.00	0.50	0.60	51 Yes
		"a photo of a man wearing a hat"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	5.00	0.50	0.70	51 Yes
		"a photo of a man wearing a moustache"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	5.00	0.50	0.65	51 Yes
		"a photo of a man wearing a pair of glasses"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	5.00	0.50	0.70	55 No
		"a photo of a man wearing steampunk goggles"	"a photo of a face of a man lowres, grainy, blurry"	"a photo of a face of a man"	7.50	0.60	0.60	51 Yes

Table A2. **Evaluation Prompts** Full list of prompts used for evaluation and user study.

Dataset	Scene Name	Positive Prompt	Negative Prompt	Inversion Prompt	Guidance Scale	Color Strength	Depth Strength	Used in the User Study
ScanNet++	A [13]	"Ancient alchemist's lab with potion bottles, parchment, and mystical symbols"	"a photo of a microbiology laboratory with microscopes at a university lowres, grainy, blurry"	"a photo of a microbiology laboratory with microscopes at a university"	7.50	0.70	0.40	51 Yes
		"Medieval herbalist lab with dried plants, wooden tables, and candle lighting"	"a photo of a microbiology laboratory with microscopes at a university lowres, grainy, blurry"	"a photo of a microbiology laboratory with microscopes at a university"	7.50	0.70	0.40	51 Yes
		"Steampunk laboratory with brass instruments, gears, and retro scientific equipment"	"a photo of a microbiology laboratory with microscopes at a university lowres, grainy, blurry"	"a photo of a microbiology laboratory with microscopes at a university"	7.50	0.70	0.40	55 No
		"a Picasso painting of a microbiology laboratory with microscopes at a university"	"a photo of a microbiology laboratory with microscopes at a university lowres, grainy, blurry"	"a photo of a microbiology laboratory with microscopes at a university"	10.00	0.70	0.50	55 No
		"an Edvard Munch painting of a microbiology laboratory with microscopes at a university"	"a photo of a microbiology laboratory with microscopes at a university lowres, grainy, blurry"	"a photo of a microbiology laboratory with microscopes at a university"	7.50	0.60	0.10	51 Yes
	036bce3393	"1950s diner-style lounge with checkered floors, red vinyl seats, and jukebox"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	10.00	0.70	0.40	51 Yes
		"Jungle-themed break room with plants, natural wood furniture, and tropical decor"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	10.00	0.70	0.40	51 Yes
		"Medieval tavern with wooden tables, stone walls, and candle lighting"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	10.00	0.70	0.40	51 Yes
		"Minimalist Japanese tea room with tatami mats, low wooden tables, and paper lanterns"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	10.00	0.70	0.40	55 No
		"Modern art studio with paint splatters, canvases, and abstract sculptures"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	10.00	0.70	0.40	55 No
		"Retro arcade room with classic game machines, colorful lights, and vintage posters"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	10.00	0.70	0.40	51 Yes
		"Victorian parlor with ornate furniture, dark wallpaper, and chandeliers"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	10.00	0.70	0.40	55 No
		"a Picasso painting of a robotics laboratory with computers and a sofa at a university"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	10.00	0.70	0.50	51 Yes
		"a Van Gogh painting of a robotics laboratory with computers and a sofa at a university"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	7.50	0.60	0.20	55 No
		"a near-futuristic night city Cyberpunk2077 style picture of a robotics laboratory with computers and a sofa at a university"	"a photo of a robotics laboratory with computers and a sofa at a university lowres, grainy, blurry"	"a photo of a robotics laboratory with computers and a sofa at a university"	7.50	0.80	0.10	51 Yes
	1b75758486	"Fauvist painting of a room with a table and chairs"	"a photo of a conference room with tables and chairs lowres, grainy, blurry"	"a photo of a conference room with tables and chairs"	12.50	0.70	0.10	55 No
	0cf2e9402d	"a Hiroshige Utagawa painting of a kitchen"	"a photo of a kitchen lowres, grainy, blurry"	"a photo of a kitchen"	7.50	0.60	0.20	51 Yes
	3db0a1c8f3	"a Bastion game style picture of a flat with a bathroom, living room and a kitchen"	"a photo of a flat with a bathroom, living room and a kitchen lowres, grainy, blurry"	"a photo of a flat with a bathroom, living room and a kitchen"	7.50	0.60	0.20	55 No

Table A3. **Evaluation Prompts (continued)** Full list of prompts used for evaluation and user study.

Dataset	Scene Name	Positive Prompt	Negative Prompt	Inversion Prompt	Guidance Scale	Color Strength	Depth Strength	Used in the User Study
Mip-NeRF 360	Garden	"a photo of a fountain in the desert"	"a photo of a fake plant on a table in the garden lowres, grainy, blurry"	"a photo of a fake plant on a table in the garden"	10.00	0.80	0.50	55 No
		"a photo of a halloween garden with pumpkins"	"a photo of a fake plant on a table in the garden lowres, grainy, blurry"	"a photo of a fake plant on a table in the garden"	10.00	0.65	0.40	51 Yes
		"a photo of a snowman on a table in the garden in the snow"	"a photo of a fake plant on a table in the garden lowres, grainy, blurry"	"a photo of a fake plant on a table in the garden"	7.50	0.70	0.40	55 No
Our Capture	Car	"80s retro car on a neon grid landscape with pink and blue glowing lights"	-	"Photo of a car"	10.00	0.85	0.30	51 Yes
		"Monster truck car with oversized wheels and a lifted body"	-	"Photo of a car"	10.00	0.70	0.30	51 Yes
		"Photo of a tron legacy nightrider neon futuristic night"	-	"Photo of a car"	5.00	0.95	0.20	51 Yes
		"Tank with treads and heavy armor in a rugged environment"	-	"Photo of a car"	10.00	0.70	0.30	55 No

Table A4. **Evaluation Prompts (continued)** Full list of prompts used for evaluation and user study.

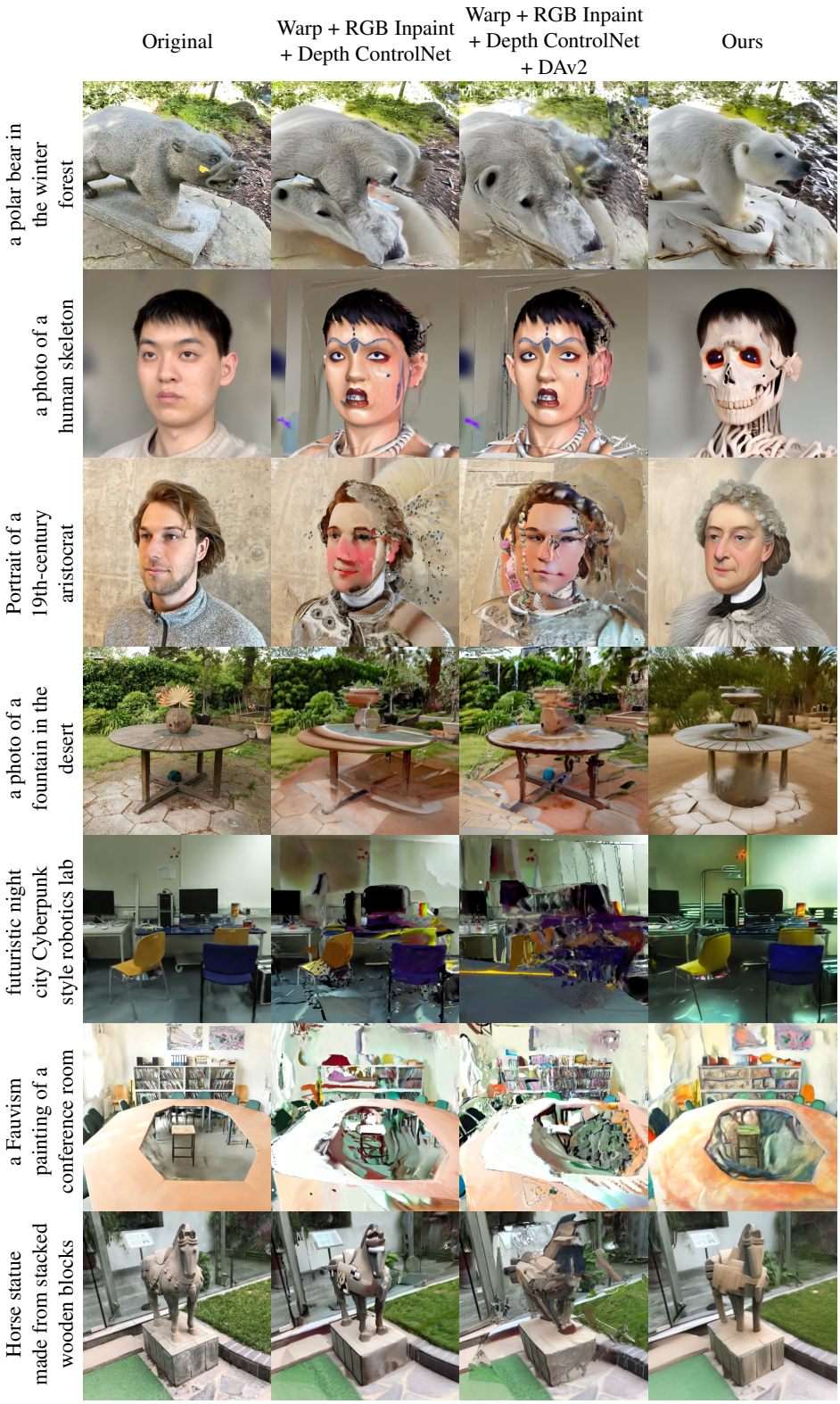

Figure A9. **Qualitative Comparison from Novel Views for Ablations**