CamCtrl3D: Single-Image Scene Exploration With Precise 3D Camera Control

Stefan Popov
spopov@google.com

Amit Raj
amitrajs@google.com

Michael Krainin
mkrainin@google.com

Yuanzhen Li
yzli@google.com

William T. Freeman
wfreeman@google.com

Michael Rubinstein
mrub@google.com

Google DeepMind

Figure 1. Our method CamCtrl3D generates videos of scene fly-throughs, given an initial image for frame #0 and a 3D camera trajectory (bottom row). The generated videos are high-quality and closely match the ground truth (top row).

Abstract

We propose a method for generating fly-through videos of a scene, from a single image and a given camera trajectory. We build upon an image-to-video latent diffusion model [5]. We condition its UNet [25] denoiser on the camera trajectory, using four techniques. (1) We condition UNet's temporal blocks on raw camera extrinsics, similar to MotionCtrl [36]. (2) We use images containing camera ray parameters, similar to CameraCtrl [14]. (3) We re-project the initial image to subsequent frames and condition on the resulting video. (4) We introduce a global 3D representation using 2D ⇔ 3D transformers [32], which implicitly conditions on the camera poses. We combine all conditions in a ContolNet-style [42] architecture. We then propose a metric that evaluates overall video quality and the ability to preserve details with view changes, which we use to analyze the trade-offs of individual and combined conditions. Finally, we identify an optimal combination of conditions. We calibrate camera positions in our datasets for scale consistency across scenes, and we train our scene exploration model, CamCtrl3D, demonstrating state-of-the-art results.

1. Introduction

Generating fly-through videos of a scene from a single image and a predefined camera trajectory has been a long-standing challenge in the fields of computer graphics and computer vision. The ultimate goal is to provide users the ability to walk into their own photographs; to turn a single, specific view of a scene into a full, immersive viewing experience with minimal capturing effort.

Recent advances in image and video generation techniques [2, 5, 16, 36], have brought us closer to realizing this goal. In this work, we present an approach that integrates precise 3D camera controls directly into a pre-trained generative video model. Our approach leverages the priors learned by the video model to generate realistic and controllable explorations of a scene captured in a single image.

Several recent works have explored incorporating camera control into existing video models using indirect conditioning signals, such as raw camera extrinsics [36] or images with camera ray coordinates [14, 37]. We adopt these two signals and propose two additional, novel approaches: (1) integrating a global 3D representation into the video generation model, using a physically accurate

2D ⇔ 3D feature exchange mechanism (Section 3.5), and (2) re-projecting the initial image over subsequent frames and using the resulting video as a conditioning signal (Section 3.4). The first approach introduces explicit 3D understanding in the model and enables inter-frame interactions that are consistent with principles of light transport. This implicitly conditions the model on the 3D camera poses. The second approach generates re-projected sequences that closely resemble the ground truth for surfaces observed in the initial image, allowing the network to efficiently copy these regions with minimal modification.

We implement these four conditioning approaches (raw camera extrinsics, camera rays, 2D ⇔ 3D transformer, initial image reprojection) into a unified framework and propose a ControlNet-style approach for their combination (Section 3.6). To identify the optimal combination, we examine the trade-offs of individual and combined conditions, using a dataset [10] with precise metric-scale camera poses (Section 4.3). For precise evaluation, we introduce a metric that considers both the overall quality of the generated videos and the model's ability to accurately preserve input image details during view changes (Section 4.1).

Finally, we use the identified optimal conditioning combination (substantial weight given to camera extrinsics, 2D ⇔ 3D, and initial image re-projection; small weight, albeit still important and improving results quality, given to camera rays), and we train our scene exploration video model CamCtrl3D (Section 4.4). We use two datasets that offer crisp videos with natural framing and diverse content. The camera poses in these datasets are estimated with structure-from-motion [28], and are thus precise only up to an unknown per-scene global scaling factor. Thus, to ensure accurate interpretation of scale during camera movement, we calibrate both datasets to metric scales, using a contemporary metric depth estimation method (Section 4.2).

In summary, our contributions are: (1) We propose two novel camera conditioning techniques based on principles of light transport; (2) We integrate these with techniques from existing works [14, 36, 37] into a unified framework; we analyze the trade-offs of individual and combined conditions and propose an optimal combination, and then train a scene exploration model CamCtrl3D with the optimal combination of conditioning strategies, demonstrating state-of-the-art results. (3) We propose a precise metric that evaluates both overall quality and ability to preserve details with view changes. We then calibrate camera positions in our datasets, enabling models to interpret scales correctly.

2. Related work

Novel view synthesis Gaussian splatting and NeRF-based methods [2, 3, 19, 21] achieve high quality novel view synthesis, but require a large number of images as input and are often trained on a per-scene basis. SparseFusion [46] and ReconFusion [40] combine NeRF with diffusion model priors to reduce the required input images, but still need more than one. While these methods can generate fly-through videos, they all require more than one image as input, and significantly more for non-object-centric cases. Two recent works, CAT3D [11] and 4DiM [37], demonstrate impressive novel view synthesis results from as few as a single image, but require extensive training data. CAT3D is trained ≈1M posed videos, while 4DiM is trained on 30M unposed videos and ≈250K posed ones. In contrast, our model is trained on just 10K posed videos. We quantitatively benchmark our method against 4DiM in Section 4.4, utilizing their reported FVD and PSNR metrics on the RealEstate10K dataset [45]. We do not compare to CAT3D, due to the absence of both single-image quantitative evaluation results and publicly available source code for this method.

Video models as priors The success of diffusion models in image generation [16, 22, 31] has inspired a wave of recent research on video generation [1, 5, 13, 15, 17, 24, 44], both from textual prompts and from single images. Our method builds on one of these works, namely Stable Video Diffusion [5] (SVD). Most of these methods offer only coarse control over the generated videos, primarily through the input textual prompts and images. AnimateDiff [12] allows transferring motion between videos, while VideoComposer [34] allows control through textual, spatial, and temporal 2D conditions. Two recent works, MotionCtrl [36] and CameraCtrl [14], condition video models on 3D camera trajectories. We adopt their conditioning signals in our model (Sections 3.2 and 3.3). We further compare to MotionCtrl in Sections 4.3 and 4.4. CameraCtrl requires additional textual prompts with input images which is not needed in CamCtrl3D

Global 3D representations Several recent works propose the use of structured latent 3d representation for novel-view-synthesis. SynSin [39] and WorldSheet [18] reconstruct 3D geometry from a single view and use a differentiable renderer to propagate gradients after re-projection. Our re-projection condition (Section 3.4) also leverages geometry, but uses a traditional forward-only renderer. Additionally, the strong priors of the base video model eliminate re-sampling artifacts common to SynSin and WorldSheet. DeepVoxels [29] extracts a volumetric representation from several views of a scene, using a voxel-based encoder-decoder. PixelNerf [41] extracts pixel-aligned features from a sparse set of input views and uses them to predict volumetric density and color at Nerf query points. GenNVS [7] lifts a single input view into a volumetric latent feature grid, with the help of a neural network. VQ3D [27] relies on depth estimation to build a tri-plane representation of a scene. Several concurrent works [8, 9, 38] rely on generalizable gaus-

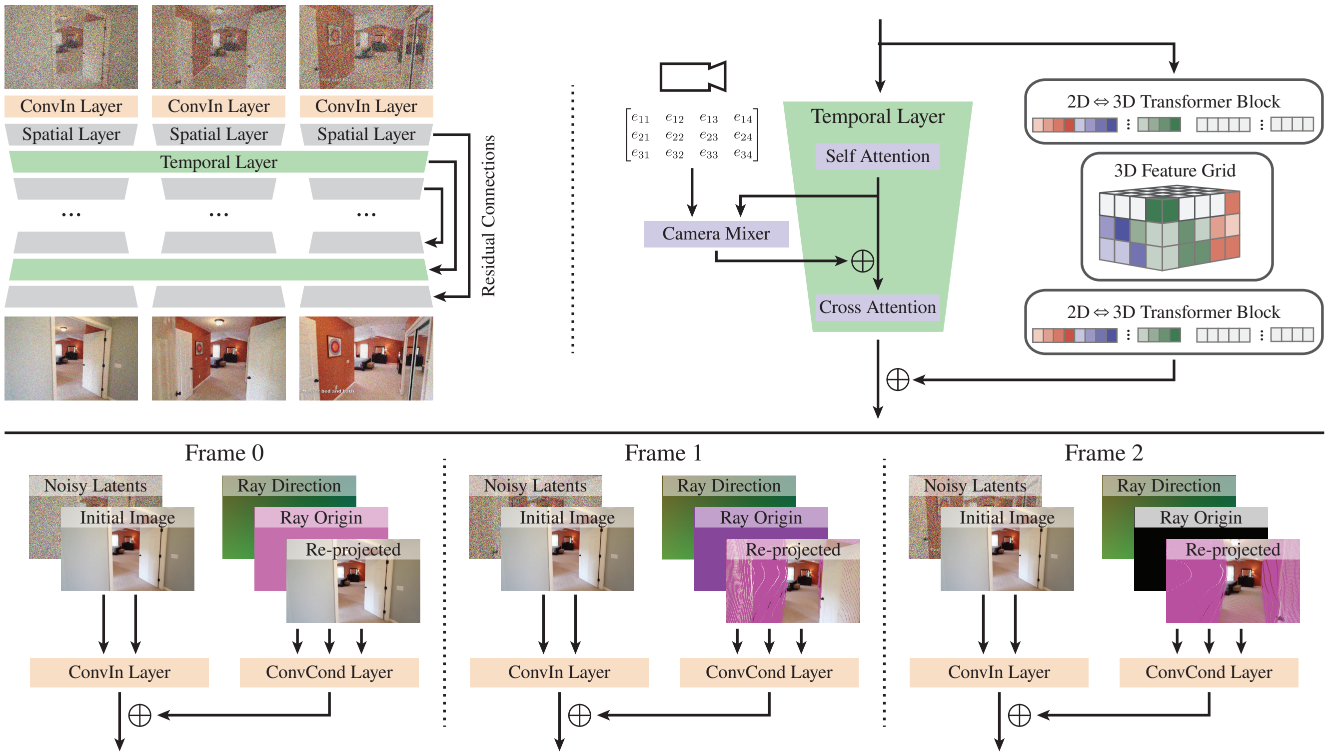

Figure 2. **Top left:** We add camera conditioning to the UNet denoiser of SVD [5] by modifying its layers. **Top right:** We attach the camera extrinsics and the 2D⇔3D transformer conditions to UNet's temporal layers (Sections 3.2 and 3.5). **Bottom:** We add additional top-level convolutional layers for the camera ray and re-projected image conditions (Sections 3.3 and 3.4)

sian splatting to renderer novel views from a sparse set of inputs. RayTran [32] relies on sparse 2D⇔3D transformers to build a global 3D representation for both 3D reconstruction and rendering novel views. Our method takes inspiration from these works and proposes a conditioning approach built using RayTran's 2D ⇔ 3D sparse transformers.

3. Proposed Approach

CamCtrl3D takes an initial RGB image $\mathbf{I_0}$ and a sequence of camera poses $\{\mathbf{c}_i\}_{i=0}^N$ as input. The image depicts a virtual 3D scene $\mathbf{V}$ from the perspective of the first camera. As output, CamCtrl3D generates a sequence of views $\{\mathbf{I}_i\}_{i=0}^N$ of the virtual scene $\mathbf{V}$, corresponding to the remaining cameras.

To achieve this, we modify a pretrained video generation model, Stable Video Diffusion [5] (SVD), and more precisely its UNet denoiser. We condition UNet's temporal blocks on the raw camera extrinsics (Section 3.2). We further provide UNet with images containing the camera ray origin $\mathbf{o}_i \in \mathbb{R}^{W\times H\times 3}$ and directions $\mathbf{d}_i \in \mathbb{R}^{W\times H\times 3}$ for each frame i (Section 3.3). We re-project the input image with the camera poses using estimated depth and we condition UNet on the resulting video (Section 3.4). We introduce 3D understanding to UNet, using a global 3D representation and sparse 2D ⇔ 3D transformer blocks, and we condition UNet in 3D (Section 3.5). Finally, we combine all conditions in a ControlNet-style [42] architecture (Section 3.6).

3.1. Preliminaries

SVD [5] is a latent video diffusion model [6], fine-tuned for high-resolution image-to-video generation. It uses a reverse diffusion [30] process with a learned UNet denoiser to generate videos in a latent representation, and a variational autoencoder [6] to convert to and from an RGB representation. UNet has an encoder-decoder architecture with residual connections. It is built from alternating spatial and temporal blocks. Spatial blocks operate on video frames independently, across their pixels. Temporal blocks operate across time, independently within each pixel. In the following sections, we introduce camera pose conditioning, by modifying UNet's inputs and its temporal blocks.

3.2. Condition on raw camera extrinsics

Temporal blocks consist of self-attention across time, followed by cross-attention, with features extracted from the input image using CLIP [23]. We condition on the raw camera extrinsics, by inserting a residual block between the two

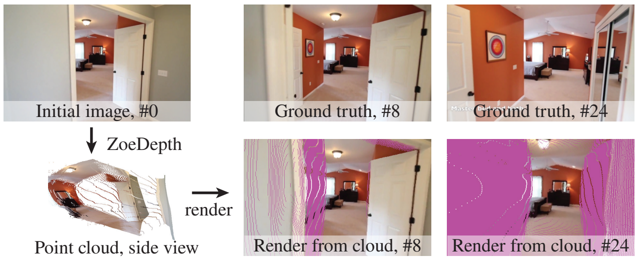

Figure 3. We re-project the surfaces observed on the initial image to all subsequent frames, using ZoeDepth [4] to estimate a point cloud. We use the resulting frames as a condition (Section 3.4) and during evaluation (Section 4.1).

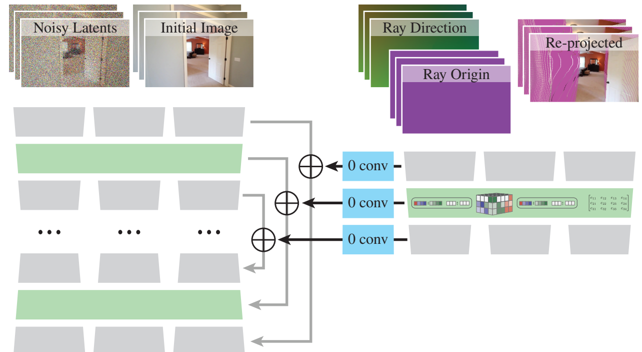

Figure 4. We apply conditions to a clone of the UNet encoder (Section 3.6), and we add its outgoing residual connections to those of the original encoder, after passing through zero convolutions [42].

attention layers (Figure 2). In it, we concatenate the 12 entries of the 4×3 camera extrinsics matrix to the features of each pixel in each frame. We then use a feed forward network to compress the features to match the cross-attention dimensions. This is similar to MotionCtrl [36], however we incorporate the feed forward outputs as residuals to facilitate back propagation.

3.3. Condition on camera rays

For each frame, we compute two new guiding images: $\mathbf{d}_i$ and $\mathbf{o}_i$. The first contains the direction of the camera rays passing through each pixel of frame i in world space coordinates, the second contains the camera origins, again in world space coordinates. We normalize $\mathbf{d}_i$'s values to the range $[0, 1]$, by adding 1 and dividing by 2. To ensure all values in $\mathbf{o}_i$ are positive, we offset all camera origins into the positive octant (+++) beforehand.

We encode the two resulting videos into a latent representation using SVD's VAE. We feed the result into a new convolutional layer and add its output to UNet's first convolutional layer (Figure 2). Similar to Section 3.2, this conditions the model on the camera parameters, however this representation is more natural as the model can reason about camera motion at pixel level.

3.4. Condition on re-projected initial image

We re-project the surface observed in the initial image to the rest of the frames and condition UNet on the resulting video. To do this, we first apply a metric-space monocular depth estimation model ZoeDepth [4] to the input image. We combine the resulting depth with the pixel colors and unproject using the parameters of the first camera $\mathbf{c}_0$. This results in a point cloud, which we render onto the subsequent frames using their respective cameras $\mathbf{c}_i$ (Figure 3). We encode the resulting video into latent space, we feed the result into a new convolutional layer, and similar to Section 3.3 we add the output to UNet's first convolutional layer (Figure 2).

The point cloud captures the visible surface of the input image. Assuming a static scene, the rendered video tracks this surface consistently with the camera motion. Additionally, we use a distinct background color during rendering that is unlikely to occur naturally. This allows the model to both stay consistent with the initial image and generate new content in place of the background color.

3.5. Condition using 2D⇔3D transformers

Intrinsically, UNet operates in 2D, on arrays of 2D grids corresponding to each frame. They are connected through the time dimension, in UNet's temporals blocks. We propose to supplement these with a new type of block that operates on a global 3D representation (Figure 2).

As input, the block accepts an array of 2D features, as well as their corresponding camera parameters (intrinsic and extrinsic camera matrices). We use a voxel grid of features as a 3D representation. The grid has fixed dimensions and it is centered w.r.t. the camera origins. Its resolution varies, depending on where the block is placed inside UNet. We use sparse ray-traced attention [32] to project the input 2D array into the voxel grid, with the given camera parameters. We then use a convolutional 3D encoder-decoder with residual connections to enable reasoning in 3D. Finally, we project back onto the 2D array using sparse ray-traced attention once more. We also embed time into the feature vectors before a 2D ⇒ 3D projection, using positional encoding. This allows reasoning across time in the 3D representation, thus enabling dynamic scenes. We use the new 3D blocks alongside UNet's temporal blocks, and we add their outputs.

Ray-traced attention embeds knowledge about the image formation process directly into the model. It allows the network to jointly analyze all views and to consolidate the extracted information into a global 3D representation. It is known to work well for 3D reconstruction from RGB videos [26, 32], as well as for view interpolation [32, supplementary material]. In our case, ray-traced attention al-

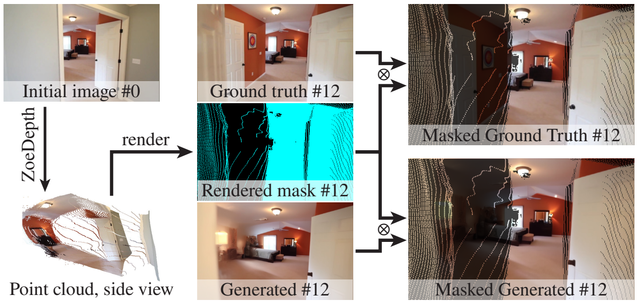

Figure 5. Re-projection (Sec. 3.4) identifies regions within a frame originating from the initial image (*e.g*. frame #12 here). We apply the resulting mask to both ground truth and generated images and measure image difference (Section 4.1) to assess the model's ability to maintain visual consistency during camera change.

lows the network to reason about the world contents directly in 3D, and to then project this into the individual video frames.

3.6. Combine conditions with ControlNet

We incorporate the above conditions into UNet in a ControlNet-style [42] architecture. We clone UNet's encoder and we attach all conditioning layers to it. We attach zero convolution layers [42] to its outgoing residual connections and we add their outputs to the respective residual connections in the original encoder (Figure 4).

4. Experiments

We first perform a set of ablations to study the trade-offs of the different conditioning methods (Section 4.3). Building on insights from this study, we develop an optimal conditioning strategy and we train a final high-quality video model CamCtrl3D (Section 4.4).

4.1. Evaluation metric

To assess the quality of the generated videos, we consider two key factors: (1) the overall quality of the generated videos, and (2) the model's ability to maintain details from the input image as the view changes.

For (1), we compare the distribution of the generated videos to that of the test set, using Fréchet Video Distance [33] (FVD). For (2), we assume a static scene and known pixel depths in the first frame. Similar to Section 3.4, we re-project the first frame onto each subsequent frame, using the provided camera poses. This creates a binary mask per frame, identifying pixels that originate from the first frame versus those with new content (Figure 5). We use this mask to compare the generated video against ground truth, ensuring masked pixels match exactly. We measure peak signal-to-noise ratio (PSNR) for the difference of these pixels, along with LPIPS [43] and SSIM [35]. In some cases, we also report peak signal-to-noise ratio computed on the full video frame (FPSNR), disregarding the mask.

Figure 6. Metric-calibration on DL3DV, frames #12 and #24. Uncalibrated re-projections (top row) deviate significantly from the ground truth (middle row), hindering both the re-projected condition (Section 3.4) and evaluation (Section 4.1). Calibration (bottom row) rectifies this discrepancy.

4.2. Datasets

We use three datasets in our experiments: ScanNet [10], RealEstate10K [45], and DL3DV [20]. ScanNet contains videos of indoor spaces, captured with an RGB-D sensor. It offers ground-truth depth maps and precise camera poses. Because of this, we use it in our ablation studies, training on 1194 videos, and evaluating on 312. RealEstate10K and DL3DV contain videos of indoor and outdoor spaces. They offer crisp videos with natural framing and diverse content. We use them for our final model, training on 4937 RealEstate10K and 4497 DL3DV videos, and evaluating on 312 different videos from each dataset respectively.

Sampling clips The videos in these datasets are much longer than our models' output. Therefore, we sample clips matching that length, at varying sampling speeds. During training, we choose a random starting frame f and a random fractional sampling speed s, between 1 and 10. We take the F frames with indices $\lfloor f + is \rfloor$, where F is the output length of our model, and $i \in [0, F)$ is an integer. Conversely, during evaluation we always start from the first frame and we use a fixed sampling speed of either 1, 2, 4, or 8.

Varying the sampling speed offers control over camera motion during both training and evaluation. Furthermore, it influences the ratio of pixels observed on the initial image versus newly generated pixels within a frame. For the test split of ScanNet, this ratio increases proportionally with sampling speed: 4.9% at speed $\times 1$, 10.5% at speed $\times 2$, 21.8% at speed $\times 4$, and 36.7% at speed $\times 8$.

Condition	Speed ×1				Speed ×2				Speed ×4				Speed ×8			
	FVD↓	PSNR↑	SSIM↑	LPIPS↓	FVD↓	PSNR↑	SSIM↑	LPIPS↓	FVD↓	PSNR↑	SSIM↑	LPIPS↓	FVD↓	PSNR↑	SSIM↑	LPIPS↓
Baseline	218.4	13.0	0.49	0.41	149.7	13.3	0.52	0.38	163.0	13.7	0.58	0.34	303.6	14.2	0.65	0.28
X	172.1	13.9	0.51	0.38	132.2	14.0	0.53	0.36	158.9	14.4	0.59	0.32	248.2	14.7	0.66	0.27
C	78.3	17.1	0.59	0.28	106.8	16.8	0.60	0.28	138.4	16.1	0.63	0.28	172.8	16.0	0.69	0.24
P	47.3	23.9	0.79	0.15	71.5	22.7	0.78	0.16	**107.2**	21.4	0.77	0.18	146.6	20.3	0.78	0.18
R_1	51.5	21.8	0.73	0.18	76.3	20.9	0.72	0.19	118.0	20.0	0.73	0.20	149.8	19.2	0.76	0.19
R_2	47.9	22.4	0.75	0.17	73.1	21.3	0.74	0.18	116.3	20.3	0.74	0.19	152.3	19.5	0.76	0.19
R_3	49.0	22.2	0.74	0.17	73.5	21.2	0.74	0.18	117.4	20.3	0.74	0.19	146.9	19.4	0.76	0.19
R_4	50.6	21.9	0.73	0.18	74.8	20.9	0.72	0.19	113.8	20.0	0.73	0.20	148.5	19.3	0.76	0.19
$C + R_2$	49.5	22.2	0.74	0.17	73.6	21.1	0.73	0.18	110.1	20.2	0.74	0.19	145.7	19.3	0.76	0.19
$X + R_2$	48.6	22.8	0.76	0.16	72.7	21.6	0.75	0.18	111.2	20.5	0.75	0.19	150.6	19.5	0.77	0.18
$X + R_2 + P$	44.4	**24.8**	**0.82**	**0.14**	68.3	23.3	**0.80**	**0.15**	109.2	21.9	**0.79**	**0.17**	**143.3**	20.9	**0.80**	**0.17**
$X + R_2 + P + C$	**42.6**	**24.8**	**0.82**	**0.14**	**68.2**	**23.4**	**0.80**	**0.15**	108.8	**22.1**	**0.79**	**0.17**	143.7	**21.0**	**0.80**	**0.17**
MotionCtrl [36]	221.6	14.1	0.44	0.41	198.9	11.2	0.47	0.11	176.6	16.1	0.61	0.67	252.4	13.3	0.63	0.34

Table 1. Performance of our ablation models from Section 4.3 and MotionCtrl [36] on the test set of ScanNet, at different sampling speeds. The best result for each metric is highlighted in bold. X denotes raw extrinsics conditioning (Section 3.2), C denotes camera rays (Section 3.3), P denotes initial image re-projection (Section 3.4), and R_x denotes 2D⇔3D transformers (Section 3.5) attached to x UNet layers. We use an SVD model fine-tuned over ScanNet as baseline. See text for more detail.

Metric calibration The camera poses in RealEstate10K and DL3DV are estimated with structure-from-motion [28] (SfM), and are thus precise only up to an unknown per-scene global scaling factor. This is problematic when conditioning on a single image, as the model cannot learn the meaning of scale in the user-provided input camera path. Moreover, the video's motion becomes inconsistent with the re-projected motion described in Section 3.4, making this conditioning approach inappropriate (Figure 6).

We thus calibrate the two datasets. For each frame, we first estimate a metric-scale depth map, using ZoeDepth [4]. We then project the SfM point cloud onto the frame. For each SfM point, the ratio of its camera depth to the depth provided by ZoeDepth serves as an estimate of the global scaling factor for the entire video. We calculate a robust estimate of this factor by taking the mean of the depth ratios across all points and frames, after excluding the smallest and largest 10% of values. We then apply the global scaling factor to the camera positions for the video, multiplying them accordingly. To assess the accuracy of this estimation, we examined 10 random videos from each dataset, along with the 10 videos exhibiting the highest variability in per-point scales. In every instance, the observed motion within the videos closely aligned with the motion of the re-projected first frame from Section 3.4.

4.3. Ablation studies

To investigate the trade-offs of different conditioning techniques, we conduct ablation studies, starting with individual experiments for each of the methods from Section 3: raw extrinsics denoted as X below (Section 3.2), camera rays denoted as C (Section 3.3), and initial image re-projection denoted as P (Section 3.4). For the 2D⇔3D transformers condition (Section 3.5), denoted as R_x, we additionally ablate on the number x of UNet blocks that the condition is attached to (1, 2, 3, or 4), as the decreasing resolution of the 2D grids in deeper UNet blocks could potentially degrade the performance of this conditioning technique. We then conduct experiments on combinations of conditioning techniques.

In each experiment, we train a model to generate 14-frame videos with a resolution of 512x320 pixels. We use the *train* split of ScanNet [10] for training, the *val* split for evaluation, and we sample clips as described above. We resize the dataset videos to the model's resolution in an aspect-preseving way, using center cropping.

Given the resolution difference between our models and the publicly released SVD model, we first fine-tune the latter on ScanNet videos for 360K steps. We use the resulting model to initialize training in our ablation studies and we also benchmark against it. We train all ablation models for 250K steps. Consistent with ControlNet [42], we observe sudden convergence, at around 25K steps for the R_x and P conditions, while C and X converge later, at around 60K steps.

The results of our experiments are summarized in Table 1. All conditioning methods demonstrate an improvement over the baseline. When evaluated independently, the re-projected image condition P performs best. This is expected, as significant portions of the condition closely resemble the ground truth videos, allowing the network to readily incorporate them with minimal modification. The 2D⇔3D transformer conditions R_x follow closely in performance. Among them, attaching to two UNet layers performs best (R_2). Models conditioned directly on raw extrinsic matrices X perform worst, as this approach re-

Method	Dataset and speed	FVD↓	PSNR↑	LPIPS↓	FPSNR↑
MotionCtrl [36]	RealEstate10K $\times 1$	777.5	16.1	0.37	15.6
4DiM [37]	RealEstate10K $\times 1$	195.1	-		18.1
CamCtrl3D	RealEstate10K $\times 1$	**72.8**	**21.4**	**0.13**	**20.6**
CamCtrl3D	RealEstate10K $\times 2$	105.1	20.0	0.15	18.6
CamCtrl3D	RealEstate10K $\times 4$	152.7	18.9	0.16	16.5
CamCtrl3D	DL3DV $\times 1$	245.1	17.7	0.26	15.9

Table 2. Performance of our final model CamCtrl3D (Section 4.4) and MotionCtrl [36], on the RealEstate10K and DL3DV datasets, at different sampling speeds. We also include metrics for 4DiM, as reported in [37]. Our model achieves significantly better quality, compared to MotionCtrl and 4DiM.

quires learning the complex relationship between 3D extrinsic matrix values and their corresponding 2D image changes across all frames.

Combining raw camera extrinsics with 2D⇔3D transformers linked to two UNet layers, outperforms either conditioning method alone ($X + R_2$). Adding initial image re-projection further enhances performance ($X + R_2 + P$), outperforming all individual conditioning methods. Adding camera ray conditioning yields marginal improvements, resulting in the optimal technique ($X + R_2 + P + C$).

We also measure the performance of MotionCtrl [36] on our test set using our evaluation metric (see Table 1). For a fair comparison, we maximize MotionCtrl's performance on our test set by tuning its FPS and motion magnitude parameters. MotionCtrl performs on-par with our re-implementation X. All other conditioning methods outperform it. Our optimal configuration $X + R_2 + P + C$ has 5.2 times lower FVD, and 10.7 dB higher PSNR, at video sampling speed $\times 1$.

4.4. Scene exploration model CamCtrl3D

For our final scene exploration model CamCtrl3D, we employ the optimal combination of conditioning strategies identified in Section 4.3. We maintain a resolution of 512×320, while generating 25 frames per sequence. We initialize our model with the weights from the official 25-frame SVD model. We use the RealEstate10K and DL3DV datasets, with splits and sampling strategies as described in Section 4.2. Rather than pre-training and freezing as in Section 4.3 we train the full model, including the original UNet encoder and decoder, for 1.66M steps.

Figure 7 shows the outputs of our model on the RealEstate10K and DL3DV test sets, while Table 2 presents its quantitative evaluation. Our model generates high-fidelity videos with accurate camera trajectories, even for complex scenes, typical for the RealEstate10K and DL3DV datasets.

We benchmark our model against two state-of-the-art methods for camera control in video generation: MotionCtrl [36] and 4DiM [37]. Direct comparison on identical datasets and metrics (Table 2) shows our model's significant improvement over MotionCtrl, achieving an order of magnitude better FVD and 5.3 dB PSNR gain. Without access to runnable source code for 4DiM, we compare our model's FVD and FPSNR to their reported values. We use the metric-calibrated test set of RealEstate10K at $\times 1$ sampling speed, since both works report numbers on it, albeit with different calibration approaches. Our model achieves significantly better quality, with 2.7× lower FVD (72.8 *vs*. 195.1) and 2.5 dB higher FPSNR (20.6 *vs*. 18.1). At the same time our model requires significantly less training data (10K posed videos) than 4DiM (30M unposed videos and ≈250K posed videos).

5. Discussion

We have shown that by leveraging priors from video models, along with a carefully selected set of conditioning techniques, CamCtrl3D can generate fly-throughs of scenes from a single image. Due to the nature of our task, our training sets primarily consist of videos of static scenes. Thus, CamCtrl3D tends to mainly output videos of static scenes. Occasionally, the model is able to animate parts of them (*e.g*. waves moving in the ocean) due to the video priors. We observe that the model relies on the different conditioning techniques to a varying extent based on the content of the initial image, thus allowing for motion in certain kinds of scenes. We expect that fine-tuning the model on a dataset of dynamic scenes with calibrated camera parameters will enhance its ability to generate videos that accurately capture complex motion and scene dynamics.

We observed that models generating 25-frame sequences are better at maintaining video quality with greater camera motion, compared to models generating 14-frame sequences. We attribute this to better temporal reasoning enabled by the smaller inter-frame changes. However, extending sequences further (up to 80 frames) led to a decline in quality. Since we fine-tune SVD *with only 10K videos*, it is likely that the limited number of training examples does not provide sufficient information to effectively train the base model for generating longer sequences.

In conclusion, we introduced two novel camera conditioning techniques based on light transport principles and combined these with existing methods within a unified framework. Our approach enables the generation of fly-through videos from a single image and a camera trajectory, achieving state-of-the-art performance.

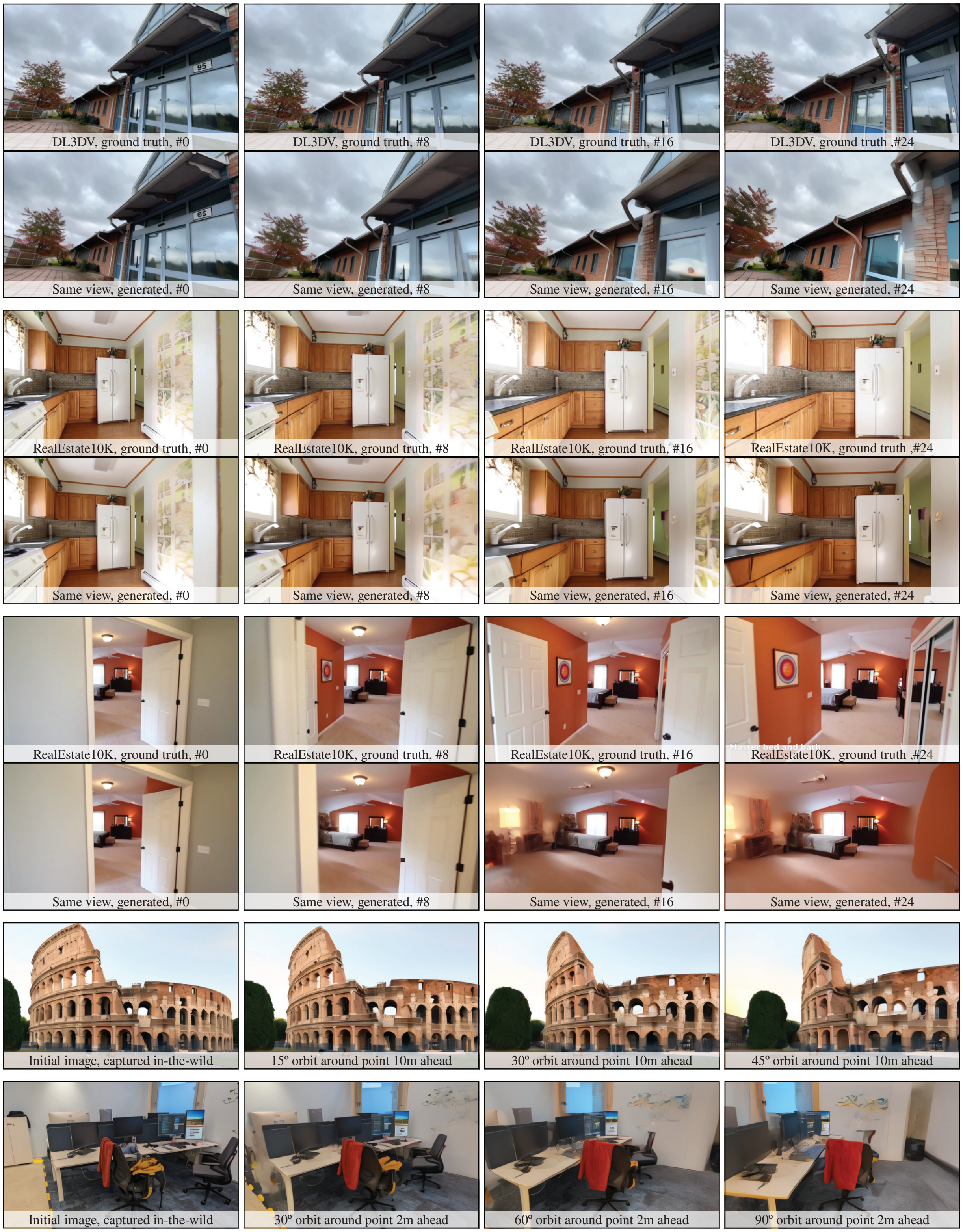

Figure 7. Results generated by the final model (Section 4.4). In all examples, we show frames #0, #8, #16, and #24 from the 25 frame video. The top 3 examples show videos from the test sets of RealEstate10K and DL3DV. Each example contains two rows, one showing ground truth (top) and another showing generated results (bottom). The bottom two examples contain video sequences generated from images in the wild. Both examples show orbiting camera motion.

References

[1] Omer Bar-Tal, Hila Chefer, Omer Tov, Charles Herrmann, Roni Paiss, Shiran Zada, Ariel Ephrat, Junhwa Hur, Guanghui Liu, Amit Raj, Yuanzhen Li, Michael Rubinstein, Tomer Michaeli, Oliver Wang, Deqing Sun, Tali Dekel, and Inbar Mosseri. Lumiere: A space-time diffusion model for video generation, 2024. 2

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 1, 2

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. *ICCV*, 2023. 2

[4] Shariq Farooq Bhat, Reiner Birkl, Diana Wofk, Peter Wonka, and Matthias Müller. Zoedepth: Zero-shot transfer by combining relative and metric depth, 2023. 4, 6

[5] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets, 2023. 1, 2, 3

[6] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023. 3

[7] Eric R. Chan, Koki Nagano, Matthew A. Chan, Alexander W. Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. GeNVS: Generative novel view synthesis with 3D-aware diffusion models. In *arXiv*, 2023. 2

[8] David Charatan, Sizhe Lester Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19457–19467, 2024. 2

[9] 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. In *ECCV*, pages 370–386, 2024. 2

[10] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 2, 5, 6

[11] Ruiqi Gao*, Aleksander Holynski*, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul P. Srinivasan, Jonathan T. Barron, and Ben Poole*. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv*, 2024. 2

[12] Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023. 2

[13] Agrim Gupta, Lijun Yu, Kihyuk Sohn, Xiuye Gu, Meera Hahn, Li Fei-Fei, Irfan Essa, Lu Jiang, and José Lezama. Photorealistic video generation with diffusion models, 2023. 2

[14] Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation, 2024. 1, 2

[15] Yingqing He, Tianyu Yang, Yong Zhang, Ying Shan, and Qifeng Chen. Latent video diffusion models for high-fidelity long video generation. *arXiv preprint arXiv:2211.13221*, 2022. 2

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1, 2

[17] 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. 2

[18] Ronghang Hu, Nikhila Ravi, Alexander C. Berg, and Deepak Pathak. Worldsheet: Wrapping the world in a 3d sheet for view synthesis from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12528–12537, 2021. 2

[19] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 2

[20] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, Xuanmao Li, Xingpeng Sun, Rohan Ashok, Aniruddha Mukherjee, Hao Kang, Xiangrui Kong, Gang Hua, Tianyi Zhang, Bedrich Benes, and Aniket Bera. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22160–22169, 2024. 5

[21] 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. In *ECCV*, 2020. 2

[22] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023. 2

[23] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 3

[24] 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

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted*

intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18, pages 234–241. Springer, 2015. 1

[26] Danila Rukhovich, Anna Vorontsova, and Anton Konushin. Imvoxelnet: Image to voxels projection for monocular and multi-view general-purpose 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2397–2406, 2022. 4

[27] Kyle Sargent, Jing Yu Koh, Han Zhang, Huiwen Chang, Charles Herrmann, Pratul Srinivasan, Jiajun Wu, and Deqing Sun. VQ3D: Learning a 3D-aware generative model on ImageNet. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 2

[28] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2, 6

[29] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2019. 2

[30] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 3

[31] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2

[32] Michał J. Tyszkiewicz, Kevis-Kokitsi Maninis, Stefan Popov, and Vittorio Ferrari. Raytran: 3d pose estimation and shape reconstruction of multiple objects from videos with ray-traced transformers. In *European Conference on Computer Vision (ECCV)*, 2022. 1, 3, 4

[33] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. FVD: A new metric for video generation. In *Deep Generative Models for Highly Structured Data, ICLR 2019 Workshop, New Orleans, Louisiana, United States, May 6, 2019*. OpenReview.net, 2019. 5

[34] Xiang Wang, Hangjie Yuan, Shiwei Zhang, Dayou Chen, Jiuniu Wang, Yingya Zhang, Yujun Shen, Deli Zhao, and Jingren Zhou. Videocomposer: Compositional video synthesis with motion controllability. In *Advances in Neural Information Processing Systems*, pages 7594–7611. Curran Associates, Inc., 2023. 2

[35] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. 5

[36] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. *arXiv preprint arXiv:2312.03641*, 2023. 1, 2, 4, 6, 7

[37] Daniel Watson, Saurabh Saxena, Lala Li, Andrea Tagliasacchi, and David J. Fleet. Controlling space and time with diffusion models, 2024. 1, 2, 7

[38] Christopher Wewer, Kevin Raj, Eddy Ilg, Bernt Schiele, and Jan Eric Lenssen. Latentsplat: Autoencoding variational gaussians for fast generalizable 3d reconstruction. In *ECCV*, pages 456–473, 2024. 2

[39] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. Synsin: End-to-end view synthesis from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[40] Rundi Wu, Ben Mildenhall, Philipp Henzler, Keunhong Park, Ruiqi Gao, Daniel Watson, Pratul P. Srinivasan, Dor Verbin, Jonathan T. Barron, Ben Poole, and Aleksander Holynski. Reconfusion: 3d reconstruction with diffusion priors. *arXiv*, 2023. 2

[41] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 2

[42] 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. 1, 3, 4, 5, 6

[43] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 5

[44] Shiwei Zhang, Jiayu Wang, Yingya Zhang, Kang Zhao, Hangjie Yuan, Zhiwu Qin, Xiang Wang, Deli Zhao, and Jingren Zhou. I2vgen-xl: High-quality image-to-video synthesis via cascaded diffusion models. *arXiv preprint arXiv:2311.04145*, 2023. 2

[45] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 37, 2018. 2, 5

[46] Zhizhuo Zhou and Shubham Tulsiani. Sparsefusion: Distilling view-conditioned diffusion for 3d reconstruction. In *CVPR*, 2023. 2