ABSTRACT

Title of Dissertation: LEARNING FROM HIDDEN SIGNALS IN VISUAL DATA
Hadi Alzayer
Doctor of Philosophy, 2026
Dissertation Directed by: Professor Jia-Bin Huang
Department of Computer Science

Cameras capture rich signals from the physical world, but extracting them requires incorporating additional priors. The work in this thesis draws on two sources of priors: known physical and geometric facts, and generative priors learned from large-scale data. In the first part, the structure comes from physics and geometry. We use this structure to extract information from reflections, motion, and defocus cues in photos and videos. We reconstruct 3D scenes from the reflections in a person's eyes, using the known geometric prior on the human cornea. We magnify subtle motions in handheld videos, by performing motion magnification in 3D, which naturally separates scene motion from camera motion. We enable post-capture defocus control on smartphones using the dual-camera setup. We use the thin-lens model combined with stereo depth to map the focus plane to a defocus map, and train a model to match any target defocus map by blending the two cameras' outputs. In the second part, the structure comes from generative models. We use them to enable image editing, 3D reconstruction under inconsistent lighting, consistent multi-view editing, and simulation of robot interactions. We use video frame pairs as natural supervision for photorealistic image editing, and build a system that takes a coarse user edit and makes it realistic. Beyond 2D images, we train a multi-view relighting model that

harmonizes a collection of photos taken under inconsistent illumination to a single consistent lighting. Then, we use the harmonized photos to reconstruct the object in 3D while preserving specular highlights and reflections. For generalized multi-view editing, we show that two pretrained diffusion models, one for 2D editing and one for multi-view generation, can be composed at inference by steering their sampling trajectories, enabling consistent multi-view editing for relighting, stylization, and spatial editing. Finally, we use the generative prior of video models to simulate robot interactions, expressing actions as partially revealed motion in the video itself. Depending on the motion revealed, the model can either act as a simulator that visualizes the effects of different robot policies, or synthesize the robot motion that achieves a desired object motion. The generality of this action representation yields a unified world model in a scalable and embodiment-agnostic setting.

LEARNING FROM HIDDEN SIGNALS IN VISUAL DATA

by

Hadi Alzayer

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2026

Advisory Committee:
Professor Jia-Bin Huang
Professor Jiajun Wu
Professor Christopehr A. Metlzer
Professor Ruohan Gao
Professor Shuvra S. Bhattacharyya

© Copyright by
Hadi Alzayer
2026

To my loving parents and sisters.

Acknowledgements

First and foremost, I would like to thank my advisor, Jia-Bin Huang, who supported me and guided me throughout my PhD journey. Jia-Bin is one of the best communicators in the world, and I learned greatly from him while writing every paper, making every presentation, and designing every figure throughout my PhD. He continually inspires me to pursue perfection when sharing my ideas, and taught me how communication is an essential part of being a great researcher. I am lucky to have Jia-Bin as an advisor, as he always puts the students' interests first, and guides them to be the best version of themselves. I am also incredibly grateful for Jiajun, who has been co-advising me for the latter half of my PhD. Some of my favorite moments with Jiajun have been the thought provoking philosophical conversations; what is good research? is language enough to achieve intelligence? and I hope to have many more of those with him. I would also like to thank my thesis committee members. Christopher Metzler who inspried me through one of my first classes at UMD and I am grateful we had the chance to work together. Ruohan Gao, I have always been inspired by your research directions, and I am hoping that we get the chance to collaborate in the future. Finally, I would like to thank Shuvra Bhattacharyya for serving as the dean's representative.

I would also like to thank my labmates and peers at both UMD and Stanford. At UMD, it was my pleasure to spend time with Sachin Shah, Kevin Zhang, Yao-Chih Lee, Songwei Ge, Brandon Feng, Yiran Xu, Yue Feng, Seungjae Lee, Ting-Hsuan Liao, Yi-Ting Chen, Quinh Phung, Haowen Liu, Grace Su, Vinayak Gupta, Vatsal Agarwal, Yiannai Karabatis, and Amir-Hossein Shahidzadeh. At Stanford, I am grateful for spending time with Stephen Tian, Kyle Sargent, Joy Hsu, Yunzhi Zhang, Cristobal Eyzaguirre, Chen Geng, Zizhang Li, Koven Yu, Jeff Tan, Ziyu Chen, Yue Gao, Yanjie Ze, Emily Jin, Fan-Yun Sun, Samuel Clarke, Michelle Guo, Keshik Chandrasegaran, Kushal Kedia, Wenlong Huang, Haonan Chen, and Alan Baade.

Throughout the summers, I spent some time at amazing internships at Adobe and Google. I'm grateful for all my internship metnors Abhishek Kar, Ryan Chan, Abdullah Abuolaim, Michael Gharbi, Eli Shechtman, Cecilia Zhang, Zhihao Xia, Dor Verbin, Pratul Srinivasan, and Jon Barron, who mentored me and remained as mentors and friends beyond the internship. I want to give a special shoutout for Michael Gharbi, who after mentoring me at Adobe, became a close friend and mentor beyond the internship, and remained as a source of inspiration for me.

Beyond the labs and offices, the outdoors and sports community have played a critical role in keeping me sane, as well as continuing to teach me how to push my limits. The experiences I have had through the outdoors is certainly one of the highlights of my PhD experience. Most importantly, I met some lifelong friends while running, cycling, swimming, or hiking together. It was an honor being part of UMD club running and tri team as well as Stanford tri team, cycling club, alpine, and running club. Thank you for all the amazing memories, and helping me push myself both mentally and physically.

I would also like to acknowledge all the people that supported me before starting my PhD. I had my first experience doing research Cornell throughout my undergrad and Master's, and I'm grateful for Kavita Bala and Bharath Hariharan who advised me at the start of my PhD journey. I'm thankful for Hubert Lin and Utkarsh Mall, who closely mentored me as I was finding my place in doing research. It is always a pleasure to see everyone from Cornell's vision community at the annual CVPR Cornell dinner, and I hope to continue to join. I would like to thank my friends who visited and remained in touch with me throughout my PhD including Kumail Alhamoud, Elias Alzayer, Jacob Revelo, George Sellers, Max Marwin. I am also thankful for the Saudi Arabian Cultural Mission for its continued support for me in the US since I have started my undergrad and all the way to the end of my PhD.

Finally, I would like to thank my parents and my sisters Noor, Joud, Fatimah, and Zainab who supported me throughout my entire academic journey. I would not be where I am without their contin-

ued support, and they continued to push me and challenge me to become the best version of myself.

Table of Contents

Dedication ii

Acknowledgements iii

Table of Contents vi

List of Tables ix

List of Figures xi

Chapter 1: Introduction 1
1.1 Thesis organization . . . 2

I Handcrafted Priors 5

Chapter 2: 3D Reconstruction from Eye Reflections 6
2.1 Introduction . . . 6
2.2 Related Work . . . 9
2.3 Background: Eye Model . . . 13
2.4 Method . . . 15
2.5 Experiments . . . 19
2.5.1 Synthetic data evaluation . . . 19
2.5.2 Real-world experiments . . . 20
2.5.3 Limitations . . . 22
2.6 Conclusions . . . 22

Chapter 3: 3D Motion Magnification 26
3.1 Introduction . . . 26
3.2 Related Work . . . 30
3.3 Preliminaries . . . 33
3.4 Method . . . 35
3.4.1 NeRF with Positional Encoding . . . 36
3.4.2 NeRF with Tri-plane Learnable Embedding . . . 38
3.5 Experimental Results . . . 39
3.5.1 Implementation Details . . . 41
3.5.2 Synthetic Scenes . . . 42
3.5.3 Real-world Scenes . . . 43
3.6 Limitations . . . 44
3.7 Conclusion . . . 44

Chapter 4: Defocus Control using Dual-Camera 46

4.1 Introduction . . . 46
4.2 Related Work . . . 49
4.3 Learning to Refocus as a Proxy Task . . . 52
4.4 Method . . . 53
4.4.1 Data Processing . . . 53
4.4.2 Model Architecture . . . 54
4.4.3 Training Details . . . 55
4.5 Experimental Results . . . 56
4.5.1 Defocus Deblurring . . . 57
4.5.2 Shallow DoF Rendering . . . 58
4.5.3 Image Refocus . . . 60
4.5.4 Ablation Study . . . 61
4.6 Limitations and Conclusion . . . 62

II Generative Priors **65**

Chapter 5: Learning Photo Editing from Dynamic Videos 66
5.1 Introduction . . . 66
5.2 Related Work . . . 69
5.3 Method . . . 71
5.3.1 Specifying coarse structure with simple transforms . . . 73
5.3.2 From coarse edits to realistic images using diffusion . . . 73
5.3.3 Extracting details from the reference image . . . 74
5.3.4 Image synthesis by detail transfer to the coarse edit . . . 75
5.3.5 Training with paired supervision from video data . . . 78
5.3.6 Implementation details . . . 80
5.4 Experimental Results . . . 80
5.4.1 Quantitative evaluation . . . 84
5.4.2 Evaluation on user edits . . . 86
5.4.3 Ablation studies . . . 90
5.4.4 Generalization beyond real photos . . . 93
5.5 Limitations and conclusions . . . 94
5.6 Appendix . . . 94
5.6.1 Reproducibility . . . 94
5.6.2 Samples of our training data . . . 95
5.6.3 Additional comparisons with text based methods . . . 95
5.6.4 Ablation on models inputs . . . 96
5.6.5 Quantiative evaluation using CLEVR . . . 97
5.6.6 Collage transform interface . . . 98
5.6.7 Expanded user study with SDEdit . . . 99

Chapter 6: Generative Multi-View Relighting for 3D Reconstruction 107
6.1 Introduction . . . 107
6.2 Related Work . . . 109

6.3 Method . . . 112
6.3.1 Relighting model . . . 112
6.3.2 3D Reconstruction . . . 116
6.4 Architecture and Training . . . 117
6.5 Results . . . 119
6.5.1 Comparisons to State-of-the-Art Methods . . . 119
6.5.2 Ablation Studies . . . 122
6.6 Discussions and Conclusions . . . 124

Chapter 7: Composing Pretrained Generative Models at Inference 126
7.1 Introduction . . . 126
7.2 Related work . . . 129
7.3 Method . . . 131
7.3.1 Background . . . 131
7.3.2 Coupled DDPM Sampling . . . 131
7.4 Experiments . . . 134
7.4.1 Multi-view spatial editing . . . 136
7.4.2 Multi-view Stylization . . . 138
7.4.3 Multi-view Relighting . . . 140
7.5 Analysis Experiments . . . 142
7.6 Discussion and Conclusion . . . 143

Chapter 8: Masked Visual Actions for Unified World Modeling 144
8.1 Introduction . . . 144
8.2 Related Work . . . 146
8.3 Masked Visual Actions . . . 148
8.4 Method . . . 151
8.4.1 Dataset construction . . . 151
8.4.2 Model implementation and training . . . 153
8.5 Experiments . . . 154
8.5.1 Controllable video generation . . . 154
8.5.2 Robotics applications . . . 157
8.6 Discussion and Conclusions . . . 161

Chapter 9: Conclusion 162

Bibliography 164

List of Tables

4.1 **Defocus deblurring evaluation.** Performance on generating all-in-focus images from a single slice in the focus stack. The best results are in bold numbers. 56
4.2 **Bokeh blurring evaluation.** performance on simulating different slices of the focus stack from the all-in-focus image. 58
4.3 **Image refocus evaluation.** Performance on re-synthesizing focus planes given an input with different focus plane from the same scene. 59
4.4 **Ablations on Image Input**. Comparison on different input types. Although performance increases by removing the occlusion mask, qualitative performance drops (see Figure 4.7). 60

5.1 **Quantitative evaluation.** Perceptual loss, SSIM, and flow error on a held-out validation subset of our video dataset. Our method significantly outperforms all the baselines across all metrics. 87
5.2 **Quantitative ablations.** Perceptual loss on a held-out validation set from our video dataset. 92
5.3 **Ablation on models inputs**. Here we ablate the inputs to the detail-extractor and synthesis UNets. We find that the providing the mask is essential to the synthesis network, but provides no additional information to the detail extractor network. We also find that providing the timestep embedding is essential for performance, which indicates that the detail extractor network extracts different features from the reference throughout the denoising process. 97
5.4 **CLEVR rearranging evaluation.** We generate a version of the CLEVR dataset where synthesize 50 random collections of objects, and rearrange each collection in three different ways. We evaluate the method's performance in realistically rearranging the objects against the ground truth. 98

6.1 **Novel view relighting on Objaverse.** Our approach of relighting and robustly optimizing a radiance field outperforms previous state-of-the-art methods in rendering novel views under target lighting. We outperform prior work despite giving the baseline a significant advantage by providing it the true lighting. 121
6.2 **Novel view relighting on real photos from NAVI.** We show that we outperform prior work on synthesizing novel views from "in-the-wild" images. 121
6.3 **The effect of the number of frames N output simultaneously by our model.** Increasing the number of frames simultaneously processed by our model improves its performance, therefore we use $N = 64$. 123
6.4 **Training dataset ablation.** We show that training on assets with standard materials as well as objects with highly reflective materials improves performance for both types of assets. 123
6.5 **Per-image embedding ablation.** We show that our shading embeddings improve performance compared to NeRF-Casting without any per-image code and that standard appearance embeddings have an adverse effect on reconstruction quality. 124

7.1 Quantitative comparison on spatial editing. We evaluate against GT renders of the target, and use MEt3r for consistency. 136
7.2 Quantitative comparison on stylization. We evaluate the temporal and subject consistency, and MEt3r score for geometric consistency. CLIP score is computed against the edit prompt. The best metrics across methods equivalent to ours are bolded. 137
7.3 Quantitative comparison on relighting. We evaluate against GT relighting results in terms of per-image metrics, and evaluate multi-view consistency with MEt3r. 140

8.1 **Baseline comparison on diverse embodiments.** We evaluate our method against Ctrl-World [1] on DROID as a seen robotic embodiment, as well as BEHAVIOR, which uses a bimanual robotic embodiment that's unseen for all methods. Our model outperform the baseline on both datasets, and we include image-to-video and trajectory conditioned video models as a reference. 155
8.2 **Ablation on visual conditioning signal.** When using sparse conditioning signal, the performance on held out data from the training distribution on DROID is similar to using masked visual actions. However, when using the same robot with unseen gripper (such as on our real world data), or on a robot from unseen embodiment in BEHAVIOR, the gap increases significantly between our masked actions and the other conditioning methods. 157

List of Figures

2.1 **Radiance field reconstruction using eye reflections.** The human eye is highly reflective. We show that from a sequence of frames that capture a moving head, we can reconstruct the radiance field and render the scene of what the person is observing using only the reflections off their eyes. 6

2.2 **NeRF for non-line-of-sight scene.** The typical NeRF capture setup requires multiple posed images (e.g., captured from a moving camera) for reconstruction. In our setup, we gather multi-view information of the scene through light reflected from the eyes of a moving person. 9

2.3 **Cornea geometry.** The cornea size and shape is uniform across all healthy humans. It can be modeled as an ellipsoid, with eccentricity of 0.5, and curvature of 7.8mm. The key fact that we exploit is that radius of the base r_{iris} is approx. 5.5mm, and the height of the cornea h is 2.18mm. 12

2.4 **Joint optimization of radiance field and iris texture.** We trace rays that bounce off the cornea to render the scene using volume rendering. Additionally, we model the iris texture as a 2D neural field Φ whose input is the intersection of the casted ray from the camera and the cornea plane and add it to the rendered color from the radiance field to estimate the cornea measurement and compute the loss against the ground truth cornea measurement to optimize all the neural fields. To ensure that the scene is not absorbed into the iris texture, we regularize the texture field Φ with a radial loss L_{radial} that encourages the estimated texture to maintain a rotational consistency. 12

2.5 **Data processing pipeline.** To compute the iris ellipse parameters, we first obtain eye bounding boxes using GroundingDINO [2] and then conduct ellipse fitting using ELLSeg [3]. Since we only want to use the visible regions of the cornea in our radiance field optimization, to handle occlusion, we generate a segmentation mask of the iris from the approximated cornea ellipse using SAM [4]. 13

2.6 **Pose optimization quantiative eval.** We evaluate the effects of pose optimization in simulation against various noise levels on the quality of synthesized views and the accuracy of recovered poses. For the XY and Z estimation plots, the x-axis represents the percent error in the estimated quantity relative to the true iris radius in pixels. . . . 13

2.7 **Texture Decomposition Ablation.** We show that using a neural field to decompose the iris texture from the reflection im- proves reconstruction performance. 14

2.8 **Additional real results.** We show that our method works in a variety of capture conditions, like smaller objects as in the small plant on the top row, and varying eye colors. 16

2.9 **Captures without structured lighting.** We show our method on an indoor office space scene and an outdoor scene in a field using natural light, and demonstrate that we can recover the scene from the eye reflection successfully without controlled illumination. Note that the photographer is also visible in the background. 17

2.10 **Ablating texture decomposition and cornea pose optimization.** *Left*: not decomposing the iris texture can introduce artifacts in the reconstruction. *Right*: pose optimization is the most critical component for our method to reconstruct the scene. 24

2.11 **Ablating radial regularization.** Without radial regularization, the reconstructed iris texture contains parts of the scene with low disparity among observed views. 25

3.1 **3D motion magnification**. (a) Novel view synthesis with a gymnast doing a handstand while magnifying the small movements of the arms needed to remain balanced. (b) Motion magnification based on targeted frequencies through temporal filtering, where the left tuning fork vibrates at 64Hz and the right at 128Hz. We visualize x-t (space-time) slices to demonstrate the motion. 26

3.2 **Motion magnification from a handheld video.** Prior 2D motion magnification approaches (e.g. [5]) cannot handle videos captured by a moving camera, producing severe artifacts. In contrast, our approach can naturally separate camera motion from object motion, allowing us to magnify only the motion of the subject of interest. See Figure 3.7 for our magnified output. 27

3.3 **Method overview.** (a) 3D scene representation with NeRF consists of two main components: 1) Coordinate Transformation uses an embedding function E to map the input point $p \in \mathbb{R}^3$ to a high-dimensional embedding vector $E(p) \in \mathbb{R}^n$. The embedding function can be analytical (positional encoding) or learned (tri-plane). 2) The Projection Function θ (usually an MLP) takes in the point embedding and viewing direction, and regresses them into the output color c and opacity σ at p. (b) We study scenes with **subtle motions**. To model the tiny variations with NeRF, we change E over time while fixing the projection function. (c) At a given point p, we analyze its embedding variations over time: $[E_{t_0}(p), ..., E_{t_{T-1}}(p)]$. We perform temporal filtering to isolate and amplify embedding variations within a certain frequency range and then send the amplified embedding to the MLP θ, resulting in motion-magnified 3D rendering. 28

3.4 **3D motion magnification on synthetic scenes.** We generated each synthetic scene by periodically vibrating object parts. We magnify the subtle motion encoded in NeRF reconstruction using the approaches discussed in Sec. 3.4, and visualize the motion here as a 2D space-time slice image. The corresponding location of each space-time slice is indicated by a red line on the rendered view. All four approaches successfully capture and magnify the motion, although the linear Eulerian approach, *Linear - Tri-Plane*, is more prone to intensity overshooting [6], manifested as bright and dark spots in the space-time slice. 32

3.5 **Quantitative comparison.** We evaluate the quality of motion-magnified renderings as a function of the magnification factor used, using positional encoding (*Left*) and tri-plane (*Right*) as the point embedding function. With positional encoding, we evaluate two approaches to vary point embedding through phase shifts in the sine waves: *Position Shift* (shifting each 3D point) and *Encoding Shift* (shifting each frequency). With tri-plane, we evaluate two approaches to vary learned point embeddings: *Linear - Tri-Plane* (linear magnification on tri-plane) and *Phase - Tri-Plane* (phase-based magnification on tri-plane). For both embedding functions, we compare against two baseline methods for video motion magnification: *Linear - Video* (linear magnification on the NeRF-rendered video) and *Phase-Video* (phase-based magnification on the NeRF-rendered video). Results from two embedding functions are separated to enable better assessments of the impact of different magnification approaches and avoid confounding with the inherent performance gap between different embedding functions and MLP architectures. 34

3.6 **Real-world multi-view motion magnification**. Using multi-view videos from the HumanNerf dataset [7], we can capture and magnify true 3D motion. We visualize the motion here as a 2D space-time image, where the corresponding location of each space-time slice is shown on the rendered view as a red line. 37

3.7 **Real-world single-camera motion magnification**. Despite the monocular view of the dynamic scene, we can render novel views while magnifying subtle motion. We visualize the motion as a 2D image slice through time. 37

3.8 **Varying the angles of deviation from observed views**. As the deviation angle increases, magnifying through the embedding space consistently outperforms the baseline approaches that operate on the color space. 40

3.9 **Varying noise levels in training views**. When we perform Eulerian magnification on the tri-plane embedding space, *Phase - Tri-Plane* is more robust than *Linear - Tri-Plane* in the presence of noise. The finding is analogous to previous analysis [6] on color space magnification (validated by the two baseline results: *Linear - Video* and *Phase - Video*). 40

4.1 **Post-capture depth-of-field (DoF) control from dual camera.** (*Left*) Using photos captured from dual cameras with a wide field of view (FoV) and ultra-wide FoV, our method enables various DoF manipulations. (*Right*) We showcase our results of refocusing (changing the focal plane), deblurring (creating all-in-focus imagery), and synthesizing a shallower DoF (producing bokeh effects). 46

4.2 **Image refocus as a proxy task.** Since we cannot gather a *real* dataset for arbitrary focus manipulation, our idea is to train a model to perform *image refocus* using a target defocus map as an input. At the test time, our trained model can perform arbitrary focus manipulation by feeding it an arbitrary target defocus map. 49

4.3 **Data processing and high-level architecture.** (*Left*) To be able to use the reference inputs for our Detail Fusion Network, we need to align the inputs and a depth estimate to approximate the defocus map of the reference **W** and the target defocus map we would like to synthesize. We use flow-based alignment with PWCNet [8] and use the stereo depth estimated using portrait mode [9]. (*Right*) Our Detail Fusion Network (DFNet) consists of refinement modules to refine the reference inputs combined with a fusion module that predicts blending masks to combine the two refined inputs. . . . 52
4.4 **Defocus deblurring.** We showcase the results of our method against SoTA single image defcous deblurring DRBNet [10]. Note that our method restores severely blurred regions in the background that single-image based methods often struggle with. 56
4.5 **Blurring results.** Our method can synthesize shallow DoF from an all-in-focus image with a performance competitive with SoTA in bokeh rendering [11]. 58
4.6 **Refocus results.** We shift the focus plane and demonstrate that we can match the desired refocused image and the target blur without completely deblurring the input. Our method outperforms the baseline that refocuses by deblurring and reblurring the image. 59
4.7 **Ablation.** Using the occlusion mask helps the model avoid transferring warping artifacts to the final image, while using **W** only hinders deblurring performance, and **UW** only suffers from warping artifacts and lower resolution. 61
4.8 **Creative applications**. (*Top*) we apply tilt shift effect that makes large objects appear as miniatures. (*Bottom*) we use segmentation mask to deblur objects of interest and blur the background. 63
4.9 **Failure cases.** (*Top*) In regions where **UW** is out of focus, as in the table shown above, our method would not be able to restore the sharp texture shown in the **W** image when focused on that object. (*Bottom*) Aligning the **UW** using optical flow based alignment could suffer when **W** is severly blurred like shown on the warping artifacts on the window. 64

5.1 **Applications of Magic Fixup.** We propose a diffusion model for image editing. Starting from an input image (a), a user specifies their desired changes by rearranging automatically segmented scene objects using simple 2D transforms to produce a coarse edit (b). Our model transforms this coarse edit into a realistic image (c), correctly accounting for secondary effects critical for realism, such as reflections on the water (top) or changes in depth-of-field (bottom), producing much more plausible edits than state-of-the-art methods (d). Photos sourced from ©Unsplash. 66
5.2 **Comparison with text based control.** Our method directly takes a coarse user edit and makes it photorealistic. Our editing is both easy and precise, and our model can harmonize the global illumination appropriately. Text-based editing methods [12, 13] on the other hand, are not able to perform such edits, resulting in global appearance changes [12] or unrealistic image [13]. 67

5.3 **Overview.** Our pipeline (left panel) uses two parallel models, a detail extractor (top) and a synthesizer (bottom), to generate a realistic image from a coarse user edit and a mask recording missing regions caused by the edit. The detail extractor processes the reference image, a noisy version of the reference and the mask, to produce a set of features that guide the synthesis and allow us to preserve the object appearance and fine details from the reference image. The synthesizer generates the output conditioned on the mask and coarse edit. The features from the detail extractor are injected via cross-attention at multiple stages in the synthesizer, in order to transfer details from the input. Both models are finetuned on our paired dataset. The right panel shows a detailed view of our cross-attention detail transfer operator. Photos sourced from ©Adobe Stock. . . 72
5.4 **Dataset synthesis.** Our dataset synthesis pipeline starts by sampling two frames from a video. We set one frame to be the reference, which we warp by one of our motion models, and set the other frame to be the target that we warp the reference towards. To generate aligned training pairs, we use 2 motion model to warp the reference frame towards the ground truth (target frame). The first model uses optical flow (left). It provides the most accurate alignment but does not correspond to what the user would provide during inference. This motion model encourages adherence of our model's output to the layout specified using the coarse edit. To generate training pairs closer to the collage-like user inputs, we use a second motion model (right). For this, we segment everything in the image [14] and apply similarity transforms to each segment, estimated from the flow within the segment. Figure 5.10 analyses the impact of these motion models on the final result. Photos sourced from ©AdobeStock. 76
5.5 **Effects of Latent Initialization.** Starting from pure noise, as is standard practice, the model struggles to generate images with deep blacks and synthesizes nonsensical content to keep the image's mean and standard deviation close to the starting Gaussian noise. This is a known issue with current diffusion models [15, 16]. Instead, during inference, we initialize the latent to the warped image with a very large amount of Gaussian noise before running the diffusion. This simple change makes a drastic difference and lets the model preserve the image content. Photo sourced from ©Unsplash. 76
5.6 **MagicFixup applications.** Our method can generalize to a large number of different applications. We just need to create a coarse edit, and MagicFixup realistically clean up the edit. Here we show a selection of tasks we used MagicFixup to achieve, like image recomposition, perspective editing, reposing, 3D transformation, and colorization. We show the coarse edits for these examples throughout the appendix. Photos sourced from ©Unsplash and CC licensed images. 81
5.7 **Spatial editing.** We show example of scene recompositing. Our model is capable of synthesizing compelling effects that harmonize realistically with the rest of the image such as: changing the depth of field (row 2), adjusting the global illumination (green reflection on the cube, row 3), and removing or adding reflections (row 6). Photos sourced from ©Unsplash. 82

5.8 **Comparison to Anydoor [17].** Anydoor was trained to insert objects from one image to another. We can repurpose their approach for our image editing task by using the same image as source and target. Their approach does not preserve the dog's identity in this example. AnyDoor also does not harmonize the lighting properly (the sun direction and shadows are wrong), the image is too bright, and some blending seams are visible. On the other hand, our output shows natural shadows and plausible contacts with the ground, adding realistic moving sand consistent with the pose. Photo sourced from ©Unsplash. 83
5.9 **Comparison with DragDiffusion.** We use the Drag Diffusion [18] to generate the results in the right column. We seed dragging control points this method expects for each of the modified image segments, and displace them using the same affine transform used to produce our coarse edit (second column). DragDiffusion generates fairly conservative image edits, and collapses with more drastic reposing edits. However, our method successfully handles wide range of reposing levels. Photos sourced from ©Unsplash. 84
5.10 **Motion models ablation.** We compare how the 2 motion models we use to create our coarse edits (column 2) during training affect the model's behavior. If we warp the reference frame (column 1) using the flow only (column 3), the model learns how to harmonize the edges of the edited regions, but remains very conservative and does not add much details to increase realism. On the other extreme, if we only use the piecewise affine motion model (column 4), the model learns to hallucinate excessively, losing its ability to preserve object identity. Our full solution trains with both motion models (column 5) to increase the model versatility, allowing the model to generate realistic details while still maintaining good adherence to the user input. Photos sourced from ©Unsplash. 85
5.11 **Architecture ablation.** Without the detail extractor branch and using CLIP to extract the reference features (3rd column), the model struggles with spatial reasoning as it cannot access the grounding of the original reference image (1st column). This ablation's outputs are overly conservative, not steering too far away from the coarse edit (2nd column). Our full model produces much more realistic edits (4th column), with harmonious shadows and object-background contact. It refines object boundaries and synthesizes plausible reflections. Photos sourced from ©Roeselien Raymond and ©Unsplash. 86
5.12 **User study results.** We asked 21 users to compare the editing results from our method and the outputs using baselines. For each baseline, the user directly compares between our method's output and the baseline's. Overall, the users overwhelmingly preferred our method. Out of 231 votes, only 14 votes were for SDEdit [19], 10 votes were for InstructPix2Pix [12], and 19 votes for MasaCtrl [13] 89
5.13 **Ablating reference input.** When passing an arbitrary image as a reference to the detail extractor network independent of the input of the synthesis network, we find an effect similar to style transfer. The model preserves the spatial structure of the "coarse edit" input while maintaining the style and global appearance of the "reference." This behavior likely contributes to the model robustness in generalization to new domains and processing new types of edits. The scream painting is now in public domain, and remaining photos were sourced from ©Unsplash. 90

5.14 **Beyond real photos.** Despite training the model on exclusively real videos, we find that the model can generalize to new domains beyond the training data, like cartoons and vector art. Photos are using public domain materials and CC licensed images. . . . 92
5.15 **Limitations**. Since the model can was only trained on spatial edits by rearranging the parts of a single image, the model struggles in inserting objects from outside the original image. Here we see that the model attempt to stylize the bunny to have an appearance similar to the teddy bear, but struggles to preserve the bunny's identity. Photos sourced from ©Unsplash. 93
5.16 **CLEVR rearranging.** We modify the CLEVR dataset generation pipeline to generate random sets of objects in different spatial arrangements, and synthesize a coarse edit that corresponds to re-aligning one arrangement to the other. Compared to the baselines, our method can better preserve the objects and harmonizes it with the environments. We find that Drag based methods like DragDiffusion struggle in generating motion beyond reposing, and SDEdit drastically morphs the objects. 98
5.17 **Colorization.** Even though our motion models only included spatial transformations, we experiment with running the model on out of domain coarse edits. Surprisingly, we find the model to synthesize realistic colorized outputs. The model also cleans up uneven coarse edges. For example, the lightning drawn on the Mustang contains uneven curvy edges, and the model cleans it up nicely. We also show multiple samples to highlight the diversity of the outputs the model can generate to address these edits. Photos sourced from ©Unsplash. 100
5.18 **Perspective editing.** By unprojecting the scene, then reprojecting different regions using variable camera parameters, we can manipulate perspective and make distant objects appear larger. While it is time consuming to create a high quality perspective edit (ZoomShop [20] takes as long as 4 hours of manual labor), by using MagicFixup we can take a coarse reprojection and make it realistic. Here we attempt to reproduce the results from ZoomShop with our method, and include their results as a point of reference. Photos sourced from ZoomShop [20]. 100
5.19 **3D transformations.** By unprojecting the image, applying 3D transformations on the unprojected point cloud, and reprojecting, we can achieve coarse 3D edits. We show that MagicFixup can addresses the artifacts generated from the reprojection, while preserving the image identity. On the other hand, we find DiffusionHandles [21] to alter the background identity on the right wall of the first example, the number of cars in the second example, and the shading of the plate and altering the identity of the spoon next to the white mug. Photos sourced from ©Unsplash. 101
5.20 **Photoshop's puppet warp.** We experiment with finer grain deformation using Photoshop's puppet warp feature. We deform the bonsai tree to resemble a dancing figure, and introduce a rough smile to a person's face. Our model then improves the realism of the edit. Note that in the second row, the model introduced a natural smile along with wrinkles around the mouth and eyes to display a more natural smile. Photos sourced from ©Unsplash and CC materials. 102

5.21 **Comparison with model trained on public data.** To maximize reproducibility, we train a version of the model on public video datasets that we plan to publicly release and open source. Here we show that the model trained on public data can similarly address secondary artifacts like reflections, and clean up artifacts due to coarse selection and editing as shown in the first row example with a coarse segmentation of the fox. Photos sourced from ©Roeselien Raymond ©Unsplash. 103
5.22 **Additional text comparisons.** We compare our method against text conditioned baselines MasaCtrl [13] and InstructPix2Pix [12]. To preserve the input identity, MasaCtrl requires a DDIM inversion step to the input image, which is prone to failing in reconstructing the input (as we show in the first two rows, the output identity is completely different from the input due to DDIM inversion failure), and in the cases where it succeeds in DDIM inversion, it is not possible to convey the user intent through a text prompt. Similarly, InstructPix2Pix either completely changes the identity of the image, or fails into editing the image to follow the text instruction. We show the text captions we used for both baselines under the image. We show the SDEdit [19] output as a reference, and we see that it is much more effective in following the user edit than the text baselines, which is why we rely on it as our main baseline. Photos sourced from ©Roeselien Raymond, ©Unsplash, and public domain data. 104
5.23 **Dataset samples.** We highlight some examples from our dataset along with the outputs of the flow and affine motion models. Note that the flow model densely aligns the reference image to the target, while the affine transformations provide much coarser alignment. For example, notice in the last row that in the flow warped image, the woman's smile and facial expression is aligned with the target, while in the affine warped we only see an alignment through scaling and shifting the person's segmentation mask. Media sourced from ©Adobe Stock. 105
5.24 **Collage transform interface.** To create user edits while maintaining correspondences between the original image and the edit, we created the Collage transform interface. The user can select any object or part they would like to edit, apply the desired affine transformation, duplication, or deletion. The correspondence map that we maintain allow us to accurately and fairly compare against the baselines by computing flow or drag keyhandles using the correspondence maps. Photo sourced from ©Roeselien Raymond. 106

6.1 **3D reconstruction under extreme illumination variation.** We propose a method for 3D reconstruction from a set of images captured under strongly varying illumination. Our method recovers high-fidelity appearance details including specular highlights that prior state-of-the-art approaches cannot recover (*top baseline*: NeRF-Casting [22] with appearance embeddings, *bottom baseline*: NeROIC [23]). 107
6.2 **Method overview.** We first apply a relighting diffusion model that converts N images I_1, ..., I_N with known camera poses π_1, ..., π_N, captured under extremely different illuminations, to a set of images with the same poses, but rendered under the illumination of the reference image I_1 (highlighted in orange). We then optimize a neural radiance field to obtain a consistent 3D representation with a novel per-image shading embedding, which can be used to render new views of the scene from unobserved poses. 113

6.3 **A comparison of our multiview relighting with prior work on single-image relighting.** Our method first relights a set of inconsistently-lit images (one of which is shown in (a)) to match the illumination of a selected reference image (b) in that set. Single-image relighting techniques such as IllumiNeRF [24] (c) struggle to disambiguate geometry, lighting, and materials, leading to an inaccurate relighting. In contrast, our model jointly relights a set of inconsistently-lit frames, which reduces ambiguities and results in a significantly more accurate result (d) when compared with the ground truth (e). 115

6.4 **Visual comparison of novel view renderings on the Objaverse dataset**. (b) We show sample input images under extreme illumination variation. (c) Adding a per-image latent code to NeRF-Casting [22] ("NeRFCast + AE") cannot accurately explain away the variations, leading to erroneous reconstruction. (d) Due to the ill-posed nature of the problem, inverse rendering-based methods such as NeROIC [23] tend to produce lower-quality renderings with mostly diffuse appearance. (e) IllumiNeRF [24] leverages diffusion prior for single-image relighting but produces inconsistent output samples, resulting in excessive blur in rendered novel views. Note that IllumiNeRF requires access to the target illumination's environment map as input. We provide IllumiNeRF with the ground truth environment map corresponding to the reference image (a), while other methods only have access to the reference image itself. (f) Our method renders accurate appearance with specular highlights close to those in the ground truth images (g). 120

6.5 **Comparison on real world photos.** We use our method to reconstruct objects from in-the-wild photos taken in different environments. Our method can render novel views under the illumination conditions of any input image we select as the reference. Unlike prior work, our technique accurately preserves shadows (*e.g.* the bunny's ear shadow in the first row) and reflections (*e.g.* the box's handle in the second row and the car's specularities in the last row) that appear in the reference image we would like to match. 122

7.1 **Applications of coupled diffusion sampling.** Our approach enables lifting off-the-shelf 2D diffusion editing models into multi-view by combining the sampling process of 2D models with multi-view generation models to produce consistent edits. Here we showcase example view-consistent results using three 2D editing models: spatial editing, stylization, and text-based relighting. 126

7.2 **Limitations of baselines.** Using image-to-multiview model conditioned on a single edited image cannot be faithful to the remaining input views, while editing each image individually with a 2D model produces inconsistent results. While prior work [25] proposes a method to compose diffusion models, their approach produces inconsistent and flickering results. 128

7.3 **Overview of the proposed coupled sampling method.** Given two target statistical distributions modeled with diffusion models: (a) standard DDPM sampling generates two instances independently, using scores from each distribution, which leads to samples without spatial alignment; (b) in contrast, the proposed coupled DDPM sampling introduces coupling terms ∇U that pull the two sample paths together, producing spatially and semantically aligned outputs; and (c) as illustrated, the 2D samples get increasingly consistent with coupling, and the coupled multi-view samples (which we set to be our final output) are faithful to the input views. 132
7.4 **Multi-view stylization.** We show three examples of multi-view stylization of our method against the baselines. Prior work on combining diffusion models [25, 26] suffer from inconsistencies across frames. When provided with a GT mesh, SDS based methods [27] suffer from severe artifacts while Hunyuan 3D's results follow the prompt loosely when doing retexturing. 135
7.5 **Qualitative comparison on multi-view spatial editing.** The baselines struggle in preserving the identity of the input, and produce flickering artifacts across edited frames, while our results achieve both editing targets and multi-view consistency. 138
7.6 **Qualitative comparison on environment map based relighting.** Other methods tend to produce flickering artifacts (notice the change in color in the first two rows for [25, 26]), and using NeRF makes the lighting changes to be baked into the view dependent effects. Our method achieves the best overall result. 139
7.7 **Image space coupling.** Using Flux, we perform coupled sampling on different prompts. We show that the coupled samples are spatially aligned while being faithful to the prompt. 141
7.8 **Video space coupling.** Using Wan [28], we perform coupled sampling on videos conditioned on different prompts, and show that the coupled samples are highly aligned in each frame while being faithful to the prompt. *Refer to the project webpage for full videos.* 141
7.9 **Guidance strength analysis.** As the guidance strength increases, the reconstruction improves but the consistency drops. 142

8.1 **Masked Visual Actions.** We finetune a video model to condition on masked trajectories of robots, representing robot actions as pixel-space masked motions. Efficiently finetuned on only 15 hours of data, a single checkpoint of the model can act as an action-conditioned forward model to simulate robotic interactions with diverse and unseen embodiments. By conditioning it on object motion, it can also act as an inverse model that synthesizes the robot motion needed to achieve the desired outcome. We showcase the efficacy of our model for policy evaluation, model-based-planning, and action extraction where the video model acts as a policy. 144
8.2 **Comparing action representations for learning.** Low-dimensional robot actions are compact, but embodiment-specific and not aligned with the image observations used by video models. End-effector poses or robot skeletons are more visual, but remain sparse and require the model to infer geometry, contact, and interaction effects. Our masked visual actions provide dense, image-aligned conditioning, making robot motion and action directly visible, yielding a more learnable representation across embodiments and object interaction. 147

8.3 **Applications.** The masked visual actions allow using the video model as a forward model, conditioned on robot actions, or as an inverse model that predicts the robot motion that satisfies the object trajectory. The forward model can be used for planning and choosing the best trajectory sampled from a policy, or policy evaluation. On the other hand, the inverse modeling can be combined with an inverse dynamics model to estimate robot actions from the generated video. 151
8.4 **Dataset construction.** (a) to train our model, we need a reference frame of the initial scene, and the masked visual actions, and train it to reproduce a realistic video of the robot executing the input actions. (b) We use segmentation-based approach by segmenting the robot arm from robotics datasets as the masked visual actions. (c) However, to allow the user to provide arbitrary action trajectory at inference, the model needs to also accept simulated mesh visualization of those actions. As a result, we also include rendering-based dataset. Given that DROID also contains the robot state at each timestep, we render the robot URDF that matches the original video to construct masked visual actions. 152
8.5 **Generalization to unseen embodiment.** While using the raw action state such as in Ctrl-world [1] can work well within the training domain, it collapses on unseen embodiments. However, our method can generalize well to unseen embodiments. 153
8.6 **Comparing baselines on DROID.** Using image-to-video [29] or even trajectory conditioned video generation [30] with GT tracks fails to execute the robot motion or preserve the input scene. On the other hand, our model can competitively match and outperform models that take the raw robot actions [1] while maintaining generalization. 154
8.7 **Comparing action conditioning.** Training a video model on different conditioning signals on DROID such as masked visual actions, end effector visualization, or skeleton all work well within the training domain. However, when going beyond the training distribution, such as using a custom end-effector, the models trained on skeleton and end effector position would hallucinate the robot seen in training or transform the robot to match the training. Furthermore, on unseen embodiments such as bimanual robots in BEHAVIOR, using masked actions generalizes gracefully, while other conditioning signals transform and disfigure the robot. . . . 156
8.8 **Application on Planning.** By rolling out multiple trajectories from a pretrained diffusion policy, we can evaluate each trajectory by simulating the actions with the video model, and then using a VLM judge to pick the best action trajectory. We observe consistent improvement in task success when using the video model to roll out and choose best action sequences, as well as the positive correlation with the number of samples evaluated at test time. This demonstrates the ability of the model to simulate counterfactuals given the same initial condition. 158
8.9 **Policy evaluation.** Video-model rollouts consistently track ground-truth success rates across RoboCasa tasks. 159
8.10 **Action extraction.** Even without task-specific video-model training, inverse modeling recovers competitive robot behavior. 160

Chapter 1: Introduction

Every picture contains hidden stories. The motion blur of a cat's body tells us about its chase; the highlights on a vase tell us about the room's lighting; mirror reflections may show us what lies beyond corners. Unveiling those stories provides us with information on the structure of the physical world. However, each of these questions is underdetermined and we need to incorporate additional priors to recover the signal. Traditionally in computer vision, human labeled data can be used as the prior. However, we aim to augment human perception rather than match it, thus we need to find an alternative source of structure. Finding this source is the main question we ask in each chapter. The thesis is divided into two parts based on the source of the structure.

In the first part, we use known physical and geometric structure for each problem. In Chapter 2, we reconstruct 3D scenes using eye reflections in portrait photos, by using the near-universal geometry of the human eye as a prior. In Chapter 3, we magnify subtle motion in handheld videos, by using 3D as a prior that separates scene motion and camera motion. And in Chapter 4, we achieve the defocus control of a lightfield camera using a phone's dual-camera setup. We use the thin-lens model to turn depth into a defocus map, and use a small refocusing dataset to learn how to blend the two cameras' outputs to match any target defocus.

In the second part, we turn to generative models as our source of structure, either by training them on large-scale data or by composing them at inference. In Chapter 5, we use a large-scale dataset of videos as a natural source of image editing data. Frame pairs from the same video contain information on how objects naturally move, deform, and affect their surroundings, and we use them to train a model that takes a coarse user edit and makes it photorealistic. In Chapter 6, we reconstruct 3D objects from internet photos taken under extremely inconsistent lighting. We train a multi-view generative relighting model that harmonizes all the input images to the same lighting, and reconstruct

the object in 3D while preserving reflections and specular highlights. In Chapter 7, we move from training new diffusion models to composing existing ones. By composing a multi-view generation model with a 2D editing model at inference, we enable multi-view editing across a range of tasks including relighting, stylization, and spatial editing, without any new training. Finally, in Chapter 8, we use video models to simulate how robots interact with the scenes around them, letting us test robot actions without executing them in the physical world. We investigate the action space used to communicate robot motion to the video model, and use masked visualizations of the robot as a scalable, embodiment-agnostic action representation.

1.1 Thesis organization

This Thesis is organized into two parts, each presenting three works.

Part I: Handcrafted Priors. The first part uses handcrafted physical priors to extract information that is hidden in photos and videos.

1. **3D Reconstruction from Eye Reflections** (Chapter 2). Portrait photos contain information on the surrounding scene throughout the eye reflections. Given that the anatomy of the human eye is universally consistent, we use the known geometry with state-of-the-art 3D reconstruction methods to distill the reflections into a 3D representation. This chapter is based on the work in Alzayer et al. [31].

2. **3D Motion Magnification** (Chapter 3). Videos often capture subtle motion that are not perceptible, and motion magnification enables us to see the motion easily. To enable motion magnification with handheld cameras, we use 3D to naturally separate scene motion from camera motion. This chapter is based on the work in Feng et al. [32].

3. **Defocus Control using Dual-Camera** (Chapter 4). Smartphones typically have multiple cameras, each with different properties. By blending the photos captured by the different cameras, we can perform post-capture defocus control and simulate having a more expressive camera. We leverage the complementary defocus signals from a smartphone's dual-camera. This chapter is based on the work in Alzayer et al. [33].

Part II: Generative Priors. The second part replaces handcrafted priors with large-scale generative models to capture the structure of the visual world.

4. **Learning Photo Editing from Dynamic Videos** (Chapter 5). We use videos as a natural source of information for how objects interact and move in the real world. Using frames from videos, we train an image editing model that enables users to edit and move objects within a photo quickly and realistically. This chapter is based on the work in Alzayer et al. [34].

5. **Generative Multi-View Relighting for 3D Reconstruction** (Chapter 6). To reconstruct objects from internet photos, we need to harmonize the photos to make them look as if they were captured in the same environment. We train a multi-view relighting model to harmonize inconsistently lit photos, and then use reflection-aware radiance field to reconstruct the object without explicit inverse rendering. This chapter is based on the work in Alzayer et al. [35].

6. **Composing Pretrained Generative Models at Inference** (Chapter 7). Instead of training a new multi-view relighting model for each task, we introduce a training-free framework to compose generative models. We couple a 2D editing diffusion model with a multi-view generative model during sampling, to produce multi-view consistent editing. This chapter is based on the work in Alzayer et al. [36].

7. **Masked Visual Actions for Unified World Modeling** (Chapter 8). To use the prior of video models to simulate real-world interactions, we need to define an action space to interact with the video model that is scalable during training, and generalizable at test time. We introduce masked visual actions, where we reveal partial trajectories of a video as the interaction mechanism with the video model. By conditioning on visualized robot motion, our model simulates robotic interactions with diverse and unseen embodiments. Conversely, by conditioning on object motion, it can also act as an inverse model that synthesizes the robot motion needed to achieve the desired outcome. As a result, our model is a unified world model that can act as either forward or inverse model at test time. This chapter is based on the work in Alzayer et al. [37].

Part I

Handcrafted Priors

Chapter 2: 3D Reconstruction from Eye Reflections

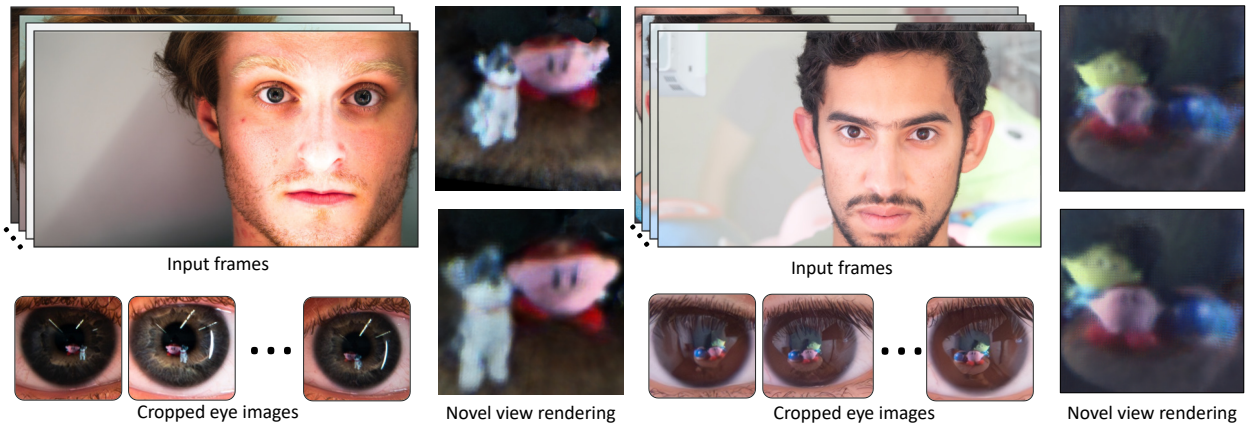


Figure 2.1: **Radiance field reconstruction using eye reflections.** The human eye is highly reflective. We show that from a sequence of frames that capture a moving head, we can reconstruct the radiance field and render the scene of what the person is observing using only the reflections off their eyes.

2.1 Introduction

The human eye is a remarkable organ that enables vision and holds valuable information about the surrounding world. While we typically use our own eyes as two *lenses* to focus light onto the photosensitive cells composing our retina, we would also capture the light reflected from the cornea if we look at someone else's eyes. When we use a camera to image the eyes of another, we effectively turn their eyes into a pair of *mirrors* in the overall imaging system. Since the light that reflects off the observer's eyes share the same source as the light that reaches their retina, our camera can form images containing information about the surrounding world the observer sees.

Prior studies have explored recovering a panoramic image of the world the observer sees and simple 3D structures like boxes that the observer is looking at from manually specified correspondences from a single image of two eyes [38, 39]. Follow-up works have further explored applications such as personal identification [40, 41], detecting grasp posture [42], focused object estimation [43], illumination estimation [44], and relighting [45]. Given the recent advancements in 3D vision and graphics, we wonder: Can we do more than reconstruct a single panoramic environment map, simple 3D structures, or recognize patterns? Is it possible to recover the 3D world seen by the observer?

In this paper, we answer these questions by reconstructing a radiance field from a sequence of eye images. We start from the insight that our eyes capture/reflect multi-view information as we naturally move our heads. We draw inspiration from the classical imaging formulation proposed by [38] and integrate it with the recent advances in radiance field reconstruction spearheaded by Neural Radiance Fields (NeRF) [46], which prior work has demonstrated can achieve high-quality view synthesis.

However, while conceptually straightforward, reconstructing a NeRF from eye images is extremely challenging in practice. The first challenge is to separate the cornea reflections from the iris textures of human eyes. Unlike the clear images of the scene typically assumed in standard captures, the eye images we obtain are inherently blended with iris textures. This composition disrupts the pixel correspondence and complicates the reconstruction process. The second challenge is cornea pose estimation. Unlike the standard NeRF capture setup, which requires a *moving camera* to capture multi-view information (often followed by camera pose estimation), our approach employs a *stationary camera* and extracts the multi-view cues from eye images under head movement, as shown in Figure 2.2. Thus, we need to estimate the 3D positions and orientation of the eyes accurately from image observations. However, this is difficult because of how small the eyes are in portraits.

To address these challenges, we repurpose NeRF for training on eye images by incorporating two

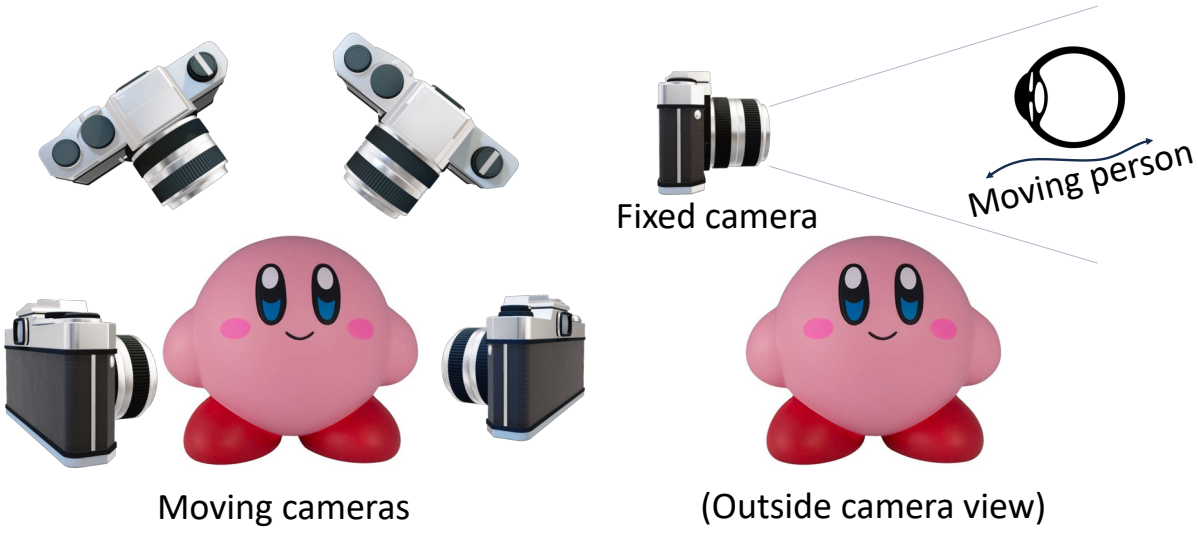
crucial components: a) iris texture decomposition, which leverages a simple radial prior to facilitate separating the iris texture from the radiance field for the scene, and b) 6DoF cornea pose refinement, which enhances pose estimation accuracy despite the challenges of the small size of eyes.

To evaluate the performance and effectiveness of our approach, we generate a synthetic dataset of a complex indoor environment with images that capture the reflection from a synthetic cornea with realistic iris texture. We further implement a real-world setup with multiple objects to capture cornea images. We conduct extensive experiments on synthetic and real-world captured eye images to validate several design choices.

Our primary contributions are as follows:

- **Radiance field recovery from eyes**. We present a method for reconstructing radiance fields of the observer's world from eye images, integrating earlier foundational work with the latest advancements in neural rendering.
- **Cornea pose refinement and analysis**. We incorporate a cornea pose refinement to correct the initial noisy pose estimates of the eyes. Using a synthetic dataset, we evaluate the sensitivity to pose errors and the necessity of pose optimization for improved view synthesis results.
- **Radial prior for irises**. We introduce a radial prior for iris texture decomposition in cornea images, improving the quality of the reconstructed radiance field.

These advancements extend the current capabilities of radiance field reconstruction through neural rendering to handle partially corrupted image observations obtained from eye reflections, opening up new possibilities for research and development in the broader area of accidental imaging [47–50] to reveal and capture 3D scenes beyond the visible line-of-sight.



(a) NeRF Setup (b) Our Setup

Figure 2.2: **NeRF for non-line-of-sight scene.** The typical NeRF capture setup requires multiple posed images (e.g., captured from a moving camera) for reconstruction. In our setup, we gather multi-view information of the scene through light reflected from the eyes of a moving person.

2.2 Related Work

Corneal imaging. Imaging techniques that use the corneas as mirrors fall under the more general category of *catadioptric imaging*. Catadioptric imaging techniques use a combination of lenses and mirrors to capture images. The word catadioptric is derived from *catoptrics* (related to the Greek words for specular and mirrors) and *dioptrics* (related to an Ancient Greek lens-like instrument). In essence, catadioptric imaging seeks to leverage an additional (often curved) mirror to expand a lens-based imaging system's effective field of view. Early studies in catadioptric imaging focused primarily on the design of the mirror profiles and their impact on the final image quality. Prior work [51] studied three design criteria of a catadioptric imaging system: 1) the shape of the mirrors, 2) the resolution of the cameras, and 3) the focus settings of the cameras. Prior work has also demonstrated that it is possible to extract depth information from multiple views captured using a catadioptric system by

extending classical depth-from-stereo ideas [52, 53].

Moreover, a creative way to realize an accidental catadioptric imaging system is by treating human eyes as external curved mirrors [38, 39]. The seminal work by Nishino and Nayar [38, 39] shows that it is possible to recover basic 3D structures like box meshes from manually specified pixel correspondences in the reflections in the eyes in a single image and the epipolar geometry that relates the two eyes. Other applications of using human eyes as part of the imaging system include estimating light direction from the eyes to perform relighting [45, 54], calibration and gaze tracking in head-mounted displays [55, 56], and calibration of display camera systems [57]. Our work advances the prior work on scene reconstruction from eye images by 1) reconstructing a radiance field, which enables new applications such as novel view synthesis and 2) removing the iris texture from scene reconstruction.

Neural radiance field. Neural radiance fields (NeRF) [46] represent a milestone in novel view synthesis. NeRF adopts differentiable volume rendering to represent a 3D scene and uses neural networks to learn the density and color of each scene point. Following the success of NeRF, a plethora of follow-up works have been introduced to improve its rendering quality [58, 59], ability to handle scene dynamics [60–63], inaccurate camera poses [64–69], and rendering speed [70–72]. Our work uses NeRF to parametrize the unknown scene we wish to recover from eye reflections. In particular, we modify the training framework from nerfstudio [73] to implement the NeRF-based scene reconstruction. Our input images are captured at a *fixed viewpoint*, differing from the typical NeRF setup that requires multi-view posed images.

Reflection removal. Removing reflections from captured images is a longstanding computational photography problem. The related literature on this topic can be summarized into two main categories:

multi-frame and *single-image*. Multi-frame reflection removal methods [74–79] often exploit the differences of motion patterns between the background and reflection layers and impose various image priors as regularization. Single-image reflection removal methods exploit visual cues available in a single image, such as depth-of-field [80, 81], defocus-disparity [82], or learned image features [83]. More recently, various NeRF-based methods have studied how to accurately model and extract specular reflections from shiny or metallic objects [84–87]. Nerfren [88] demonstrates that by fitting two NeRFs to model the reflection and transmittance components of the scene separately, reflections from planar surfaces like mirrors can be removed and re-rendered as a separate 3D scene. Our work differs from prior work on reflection removal with NeRF like Nerfren by regularizing the reconstruction of the diffuse color the human eye using a radial symmetry prior. Prior work has also focused on removing reflections, specifically from eyes, by exploiting the fact that the iris texture tends to be constant radially to design a radial autocorrelation prior [44]. Our work also differs from prior work on reflection removal from eyes by using a radial symmetry prior instead of the radial autocorrelation prior.

Non-line-of-sight imaging. Non-line-of-sight (NLOS) imaging aims to recover images of objects that are not directly visible by using light reflected off visible surfaces. Active NLOS imaging techniques involve using controlled light sources, such as time-of-flight sensors, to reconstruct the hidden scene [89]. Passive NLOS imaging, on the other hand, exploits natural or ambient light and does not require a controlled light source. Several works analyze the intensities changes on corner regions [47] or a blank wall [49] and reveal information about the hidden scene. Thermal reflections have been used to reconstruct the 3D body pose of non-line-of-sight humans [50]. Orca [86] uses reflections from a glossy object observed in multi-view images to train a NeRF for the surrounding environment. Unlike Orca, which relies on images captured with a moving camera while the "mirror" object is fixed, our

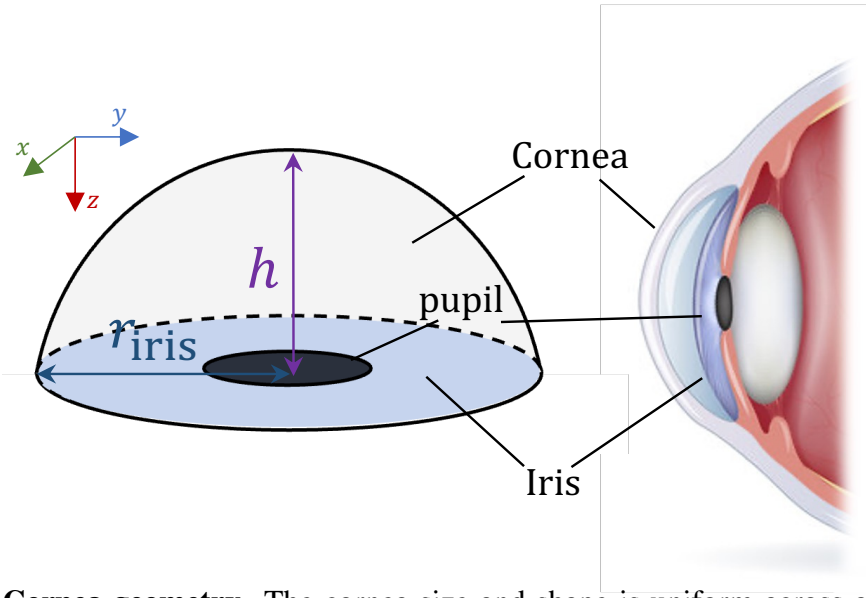


Figure 2.3: **Cornea geometry.** The cornea size and shape is uniform across all healthy humans. It can be modeled as an ellipsoid, with eccentricity of 0.5, and curvature of 7.8mm. The key fact that we exploit is that radius of the base r_{iris} is approx. 5.5mm, and the height of the cornea h is 2.18mm.

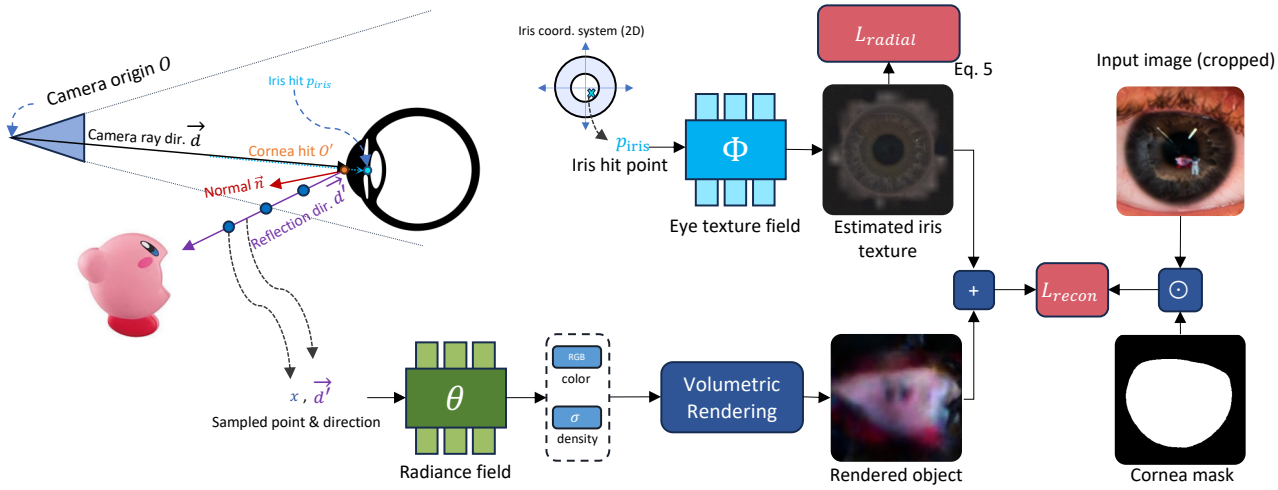
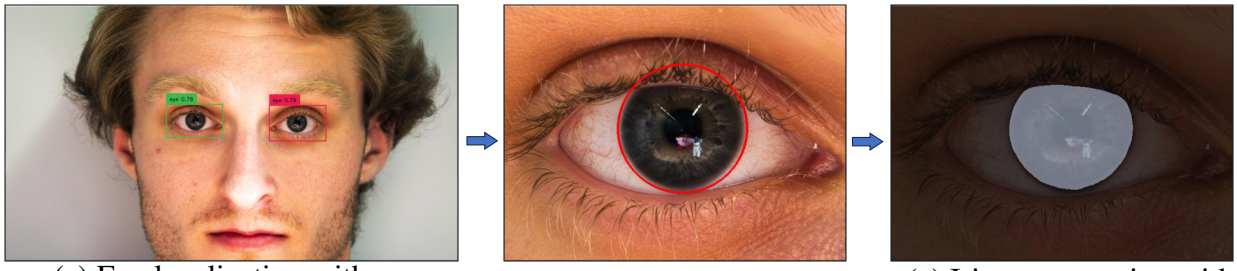
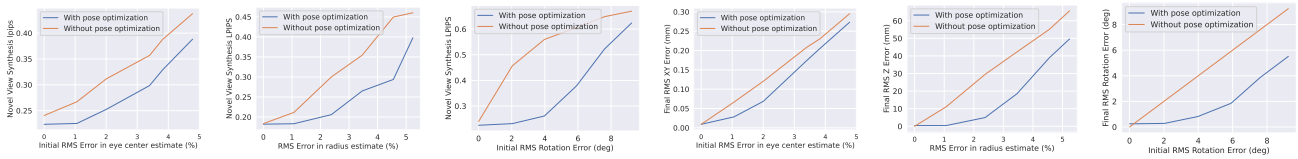


Figure 2.4: **Joint optimization of radiance field and iris texture.** We trace rays that bounce off the cornea to render the scene using volume rendering. Additionally, we model the iris texture as a 2D neural field Φ whose input is the intersection of the casted ray from the camera and the cornea plane and add it to the rendered color from the radiance field to estimate the cornea measurement and compute the loss against the ground truth cornea measurement to optimize all the neural fields. To ensure that the scene is not absorbed into the iris texture, we regularize the texture field Φ with a radial loss L_{radial} that encourages the estimated texture to maintain a rotational consistency.



(a) Eye localization with GroundingDINO (b) Ellipse fitting with ELLSeg (c) Iris segmentation with SAM

Figure 2.5: **Data processing pipeline.** To compute the iris ellipse parameters, we first obtain eye bounding boxes using GroundingDINO [2] and then conduct ellipse fitting using ELLSeg [3]. Since we only want to use the visible regions of the cornea in our radiance field optimization, to handle occlusion, we generate a segmentation mask of the iris from the approximated cornea ellipse using SAM [4].



Novel view synthesis performance GT pose recovery performance

Figure 2.6: **Pose optimization quantiative eval.** We evaluate the effects of pose optimization in simulation against various noise levels on the quality of synthesized views and the accuracy of recovered poses. For the XY and Z estimation plots, the x-axis represents the percent error in the estimated quantity relative to the true iris radius in pixels.

method works for a stationary camera and uses the natural movement of the human eye "mirrors", which is visualized in Figure 2.2.

2.3 Background: Eye Model

The geometry of the human eye has been extensively studied [90] and is well known. The known eye geometry provides a strong prior to model the camera ray interactions with the eye in closed form. The major components that are visible in the eye are: the sclera; which is the white region of the eye, and the cornea; which includes the iris and the pupil. The cornea is covered by a thin film of tear fluid, making it highly reflective. As noted by Nishino and Nayar [38], since the cornea can act as a mirror,

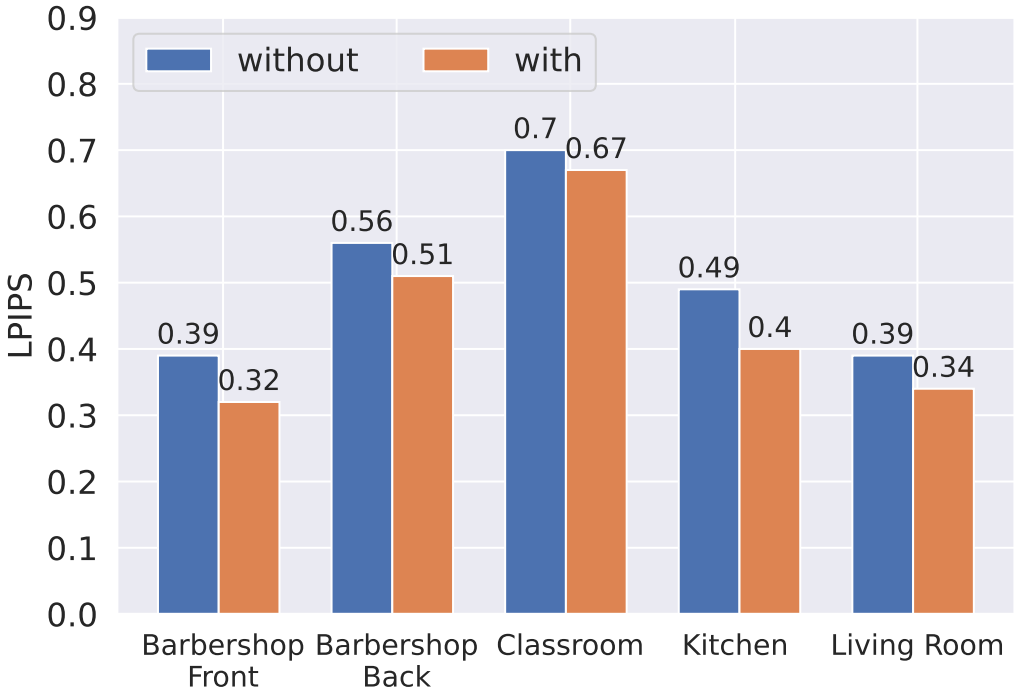


Figure 2.7: **Texture Decomposition Ablation.** We show that using a neural field to decompose the iris texture from the reflection im- proves reconstruction performance.

the combination of a camera and the cornea resembles a *catadioptric system*. In our work, we follow the same eye model adopted by Nishino and Nayar [38] for the geometry we assume for the eye.

The eye is modeled as a section of an ellipsoid, as illustrated in Figure 2.3, which can be described using

$$(1 - e)\, z^2 - 2Rz + r^2 = 0 \tag{2.1}$$

where e is the eccentricity, R is the radius of the curvature at the apex, and $r^2 = x^2 + y^2$. For an adult with healthy eyes, on average the eccentricity e is about 0.5 and curvature R is about 7.8 mm, and the radius at the base of the cornea r_{iris} is approximately 5.5 mm with a height h of 2.18mm. As in the prior work [45], we leverage the fact that humans have a known eye size to compute the average depth of the cornea in 3D by using the camera focal length f and the radius in image space r_{img} in the weak

perspective equation:

$$\text{depth}_{\text{avg}} = r_{\text{iris}} \frac{f}{r_{\text{img}}}. \tag{2.2}$$

that we use to initialize our estimate of the cornea position in our reconstruction pipeline. The initial pose is further refined, under correct perspective projection, during reconstruction using an optimization procedure that will be discussed more later. To compute the ray reflection direction, we need to compute the cornea's normal at the intersection point. Here the normal can be computed in closed form using ellipsoid equation Eq. 2.1:

$$\overrightarrow{n}\,(x, y, z) = \langle 2x, 2y, 2\,(1 - e)\,z - 2R \rangle\,, \tag{2.3}$$

which we use with the standard reflection equation to reflect the camera rays off the cornea. However, since the cornea is not a perfect mirror, we need to consider the transmitted rays that hit the iris and pupil. For simplicity, we model the iris and pupil as a flat and round plane directly behind the cornea. We can thus compute the hit points onto the iris via the camera ray-iris plane intersection.

2.4 Method

Radiance field from reflection. One can train a radiance field by minimizing the photometric loss between ground truth pixel values from captured images and the rendered color. Each pixel color is computed through volume rendering using the color and density values of the sampled points along a camera ray. The ray associated with a pixel starts from the camera origin, denoted by O, and the viewing direction, denoted by $\overrightarrow{d}$.

However, in our setup, what we are interested in reconstructing the radiance field of the scene

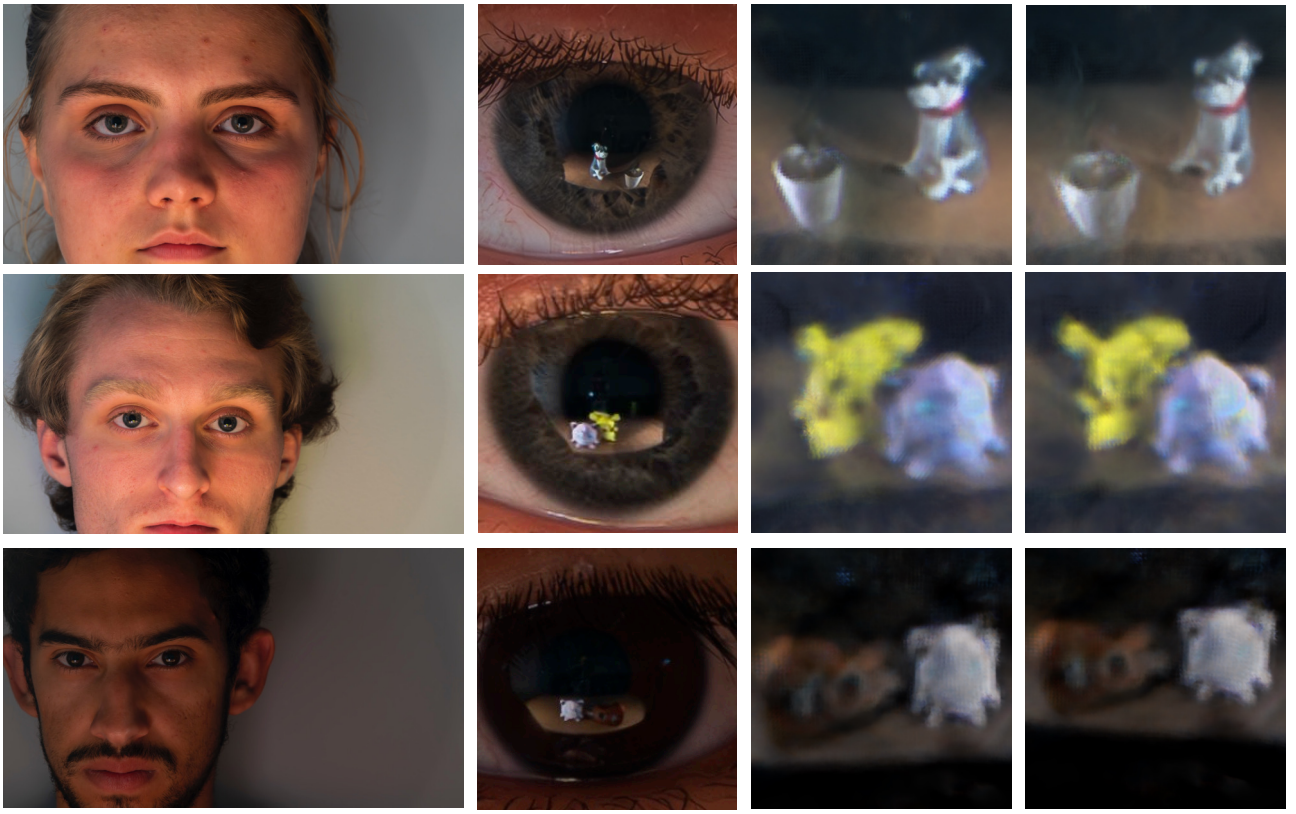


Figure 2.8: **Additional real results.** We show that our method works in a variety of capture conditions, like smaller objects as in the small plant on the top row, and varying eye colors.

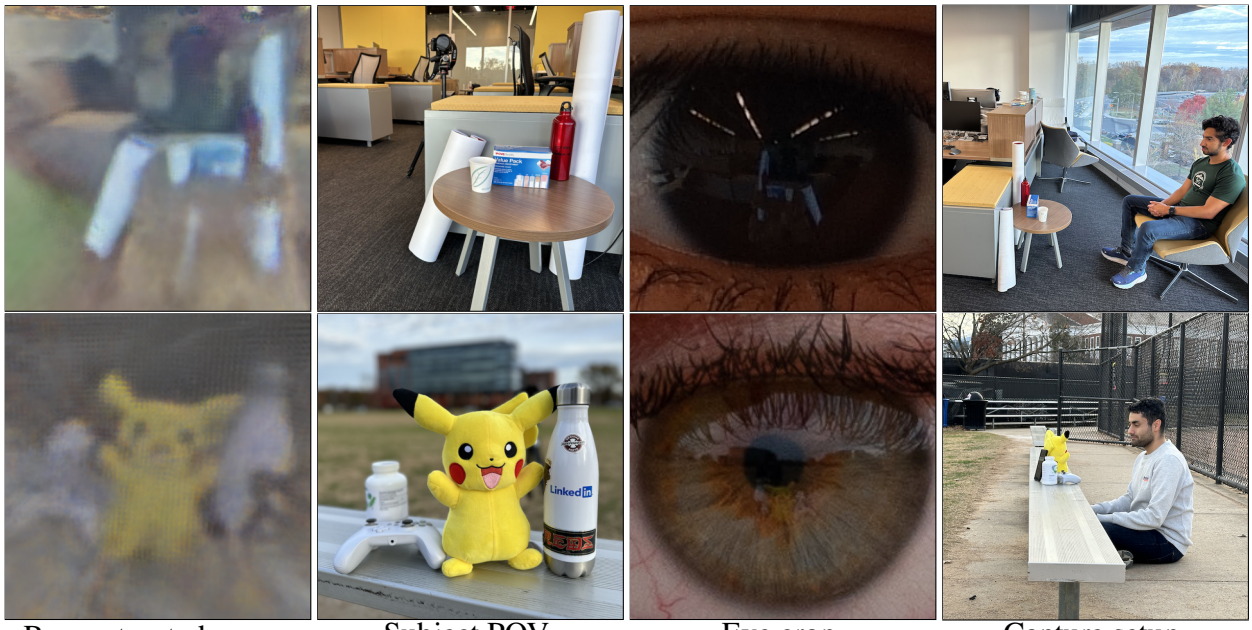


Figure 2.9: **Captures without structured lighting.** We show our method on an indoor office space scene and an outdoor scene in a field using natural light, and demonstrate that we can recover the scene from the eye reflection successfully without controlled illumination. Note that the photographer is also visible in the background.

reflected from the person's eyes. In Figure 2.4, we illustrate how we use the rays reflected from the eye. The reflected ray starts with the origin where the camera ray intersects with the cornea at O', and in the direction of the reflected ray $\overrightarrow{d'}$ instead of using O and $\overrightarrow{d}$. We compute the reflected ray explicitly using the standard reflection equation:

$$\overrightarrow{d'} = \overrightarrow{d} - 2\left(\overrightarrow{n} \cdot \overrightarrow{d}\right)\overrightarrow{n}, \tag{2.4}$$

where $\overrightarrow{n}$ is the normal at the hit point O'. Since we model the cornea geometry as an ellipsoid, we directly compute the hit points and normals using closed-form ellipsoid ray intersection formulas.

Iris texture decomposition. Since the target images are the scene reflections off the cornea, training NeRF naively leads to poor view synthesis results as the cornea images contain both reflection (scene)

and transmittance (iris texture) components. To recover only the scene in the radiance field, we jointly optimize a 2D field Φ to model the iris texture. The iris texture remains constant across the different views while the person moves, while the scene reflections vary. We thus use a texture field shared across input images. Since we model the iris as a flat, round plane behind the cornea, we can directly compute the *2D coordinate* on the iris plane by performing ray-plane intersection in 3D.

However, when a part of the scene does not display considerable motion across the training views (e.g., limited head motion of the subject), parts of the scene can be "absorbed" as part of the iris texture instead of the scene. To resolve this issue, we propose a radial regularization that encourages radial symmetry of the recovered texture. The iris is far from being perfectly rotationally symmetric. However, our observation is that as we rotate the iris, the color variance is small. We use this observation to regularize the colors we learn with the iris texture by penalizing the pixels that are rotationally inconsistent. At each step, for each point p, we empirically estimate the distribution of colors of the *ring*, by computing the average color μ_r and the standard deviation σ_r of the color ring. We then compute the range of the 10th and 90th percentile colors $[c_{10}, c_{90}]$. We only penalize the model to match the mean if a point's color falls outside of $[c_{10}, c_{90}]$ by at least a standard deviation.

$$L_{radial}(p) = \begin{cases} \ell_2(\Phi(p), \mu_r) & c_p - \sigma_p > c_{90} \\ \ell_2(\Phi(p), \mu_r) & c_p + \sigma_p < c_{10} \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

Cornea pose optimization. Due to the small cornea size in the captured images, the cornea pose and normals estimate inevitably have some errors. Training with the erroneous poses significantly affects the radiance field reconstruction's quality. To alleviate the pose errors, we optimize the 6DoF pose of

each cornea. For each cornea, we optimize for a transformation matrix $T = [R, t] \in$ SE(3), where $R \in$ SO(3) and $t \in \mathbb{R}^3$ denote the rotation and translation, respectively. We optimize the cornea poses during training similar to prior work on training NeRF with noisy poses [65, 64, 67], simultaneously with the iris texture recovery and radiance field reconstruction. We initialize the translation component of the cornea poses using Equation 2.2 as $Z = \text{depth}_{\text{avg}}$, $X = x(Z/f)$, $Y = y(Z/f)$, where x and y are the normalized image coordinates of the eye center and f is the focal length of the camera. We initialize the rotation component of the cornea poses to the identity matrix. In section 2.5.1 we analyze how the pose optimization helps improve the novel view synthesis and ground truth eye pose recovery in a synthetic setup, and analyze its sensitivity to varying noise levels.

2.5 Experiments

2.5.1 Synthetic data evaluation

We generate synthetic data in Blender with eye models placed in the scene. Since we cannot estimate the cornea poses perfectly in real captures, we evaluate the robustness of our cornea pose optimization to the noise in the estimated cornea radius/position. We study the possible noise sources: 1) the eye center estimation, 2) the iris radius estimation, and 3) the eye rotation initialization separately. To evaluate the sensitivity of our pipeline to the eye center estimation,we corrupt the observed cornea center x coordinate and y coordinate for each imaging by scaling them with varying noise levels, which results in errors in the X and Y coordinates in the 3D poses of the corneas. For iris radius estimation, we corrupt the observed iris radius for each image by scaling the estimated radius with varying noise levels, which results in errors in the Z coordinate in the 3D poses of the corneas. For the eye rotation initialization, we initialization the true rotation of the cornea in simulation with a small

amount of rotation noise. We compare the reconstruction quality and pose recovery accuracy at five noise levels for each noise source, across five different scenes, and with and without pose optimization. The quantitative results are summarized in Figure 2.6. Our experiments show that our reconstruction pipeline can tolerate about 1% error in the initial eye center estimate, 2.5% error in the initial radius estimate, or 5 degrees of rotation error, before quality starts to degrade significantly and pose recovery becomes intractable. Furthermore, we show quantitative comparisons of our method with and without iris texture decomposition in Figure 2.7. Our method performs better in terms of SSIM and LPIPS with texture decomposition than without. Notably, we do not compute PSNR because in our setting there is a drastic difference in intensities between the reflection and the scene itself. In the project webpage, we include synthesized novel views of our reconstructions of the synthetic scenes.

2.5.2 Real-world experiments

We describe capturing real-world images and demonstrate the effectiveness of our method on real captures.

Image capture. To maintain a realistic field of view, we capture images with a field of view that matches a standard portrait capture where the entire head is visible within the frame. We test our method with various illumination conditions, ranging from structured scenes with bright external lights, to unstructured indoor and outdoor settings with natural lighting. We ask the person to move within the camera's field of view and capture 5-15 frames per scene. We captured the images using a Sony RX IV camera as RAW 16-bit photos and post-processed the images using Adobe Lightroom to reduce the noise in the cornea's reflection. We vary the illumination brightness and the reflected object size for a comprehensive evaluation.

2.5.2.1 Data processing

We estimate the cornea's center and radius on images to get an initial estimate of the cornea's 3D location. Once we have the radius, we can directly approximate the cornea's 3D location using the average depth from Eq. 2.2 and the camera's focal length, and also compute its surface normals using Eq. 2.3. To automate the process, we locate the eyes bounding boxes using Grounding Dino [2] and then use ELLSeg [3] to perform ellipse fitting for the iris. While the corneas are typically occluded, we only need the unoccluded regions, so we obtain a segmentation mask for the iris using Segment Anything [4].

2.5.2.2 Results from real captures

Using our captured images, we show that our method enables the reconstruction of radiance fields from real-world portrait captures despite the cornea location and geometry inaccuracies that may arise in the real world. In Figures 2.1 and 2.8, we run our method in a controlled setting with external area lights and object-centric scenes to evaluate the best possible performance by our method and its generalization to the real-world results. In Figure 2.9 we show that our method can work even without controlled illumination. We find that pose optimization is the most critical component for a coherent reconstruction. Our finding validates sensitivity analysis of error in cornea poses shown in Section 2.5.1. In Figure 2.10, by ablating the cornea pose optimization and texture decomposition from our method, we demonstrate that both cornea pose optimization and texture decomposition are necessary for successful scene reconstruction. The initial cornea pose estimation is noisy because the blurriness of the cornea boundary makes it challenging to be localized precisely in the image, as

shown in Figure 2.11. In Figure 2.10 we show the rendered radiance field with and without the iris texture decomposition. We notice significantly more floaters when not explicitly modeling the texture. Furthermore, Figure 2.11 demonstrates that the radial regularization improves the reconstruction by preventing scene regions with limited disparity of getting absorbed into the learned iris texture.

2.5.3 Limitations

Our work demonstrates the feasibility of reconstructing the 3D world only from eye reflections. Our current real-world results are from a "laboratory setup", such as a zoom-in capture of a person's face, and deliberate person's movement. We believe more unconstrained settings remain challenging (e.g., video conferencing with natural head movement) due to lower sensor resolution, compression, and motion blur.

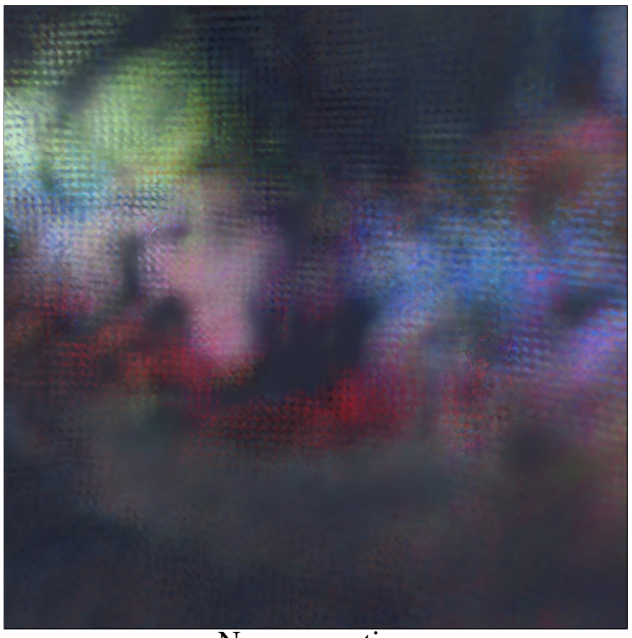
2.6 Conclusions

By leveraging the reflections of light off human eyes, we develop a method that reconstruct the scene observed by a person using monocular image sequences captured at a fixed camera position. We demonstrate that naively training a radiance field on the observed reflections is insufficient due to several factors: 1) the inherent noise in cornea localization, 2) the complexity of iris textures, and 3) the low-resolution reflections captured in each image. To address these challenges, we introduce cornea pose optimization and iris texture decomposition during training, aided by a radial texture regularization loss based on the nature of the human iris. We validate the design choices with synthetic data and showcase view synthesis results from real-world captures of varying difficulty, including cases with and without external light and outdoor setting. With this work, we hope to inspire future

explorations that leverage unexpected, accidental visual signals revealing information about the world around us, broadening the horizons of 3D scene reconstruction.



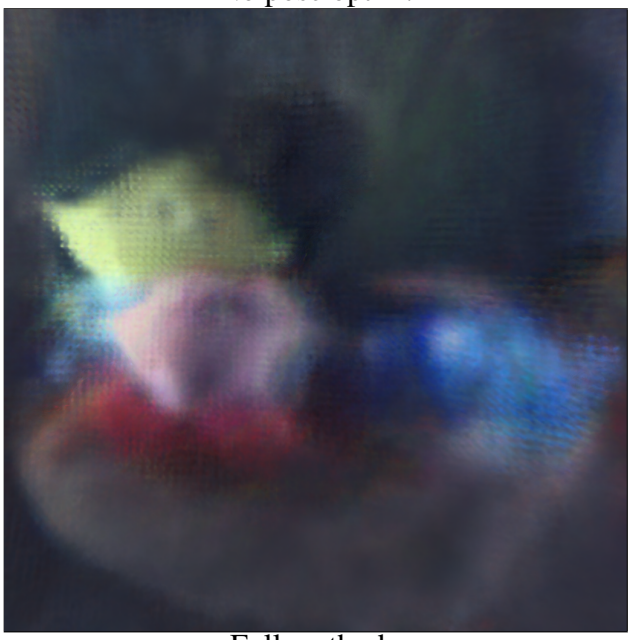
No texture field



No pose optim.

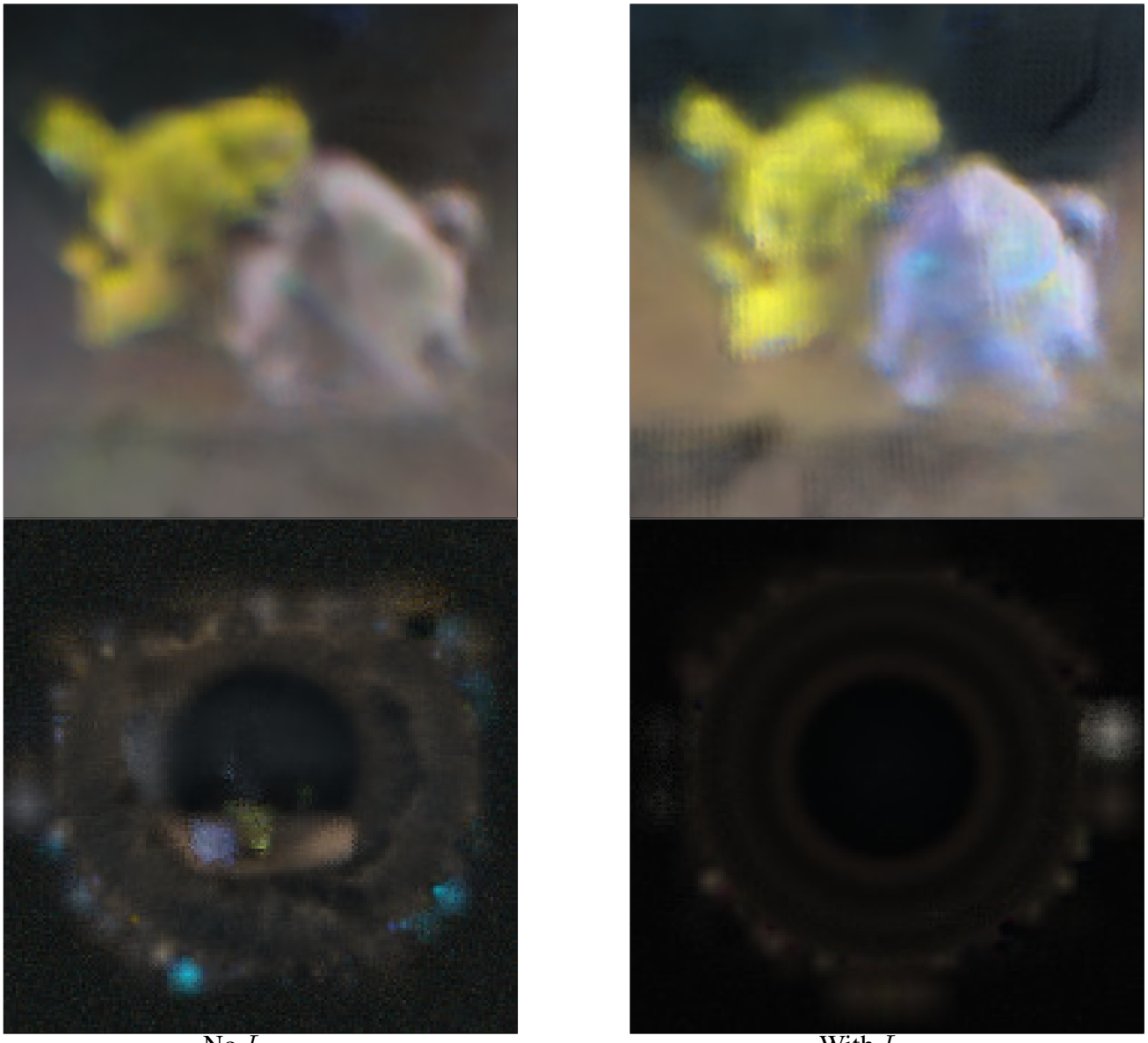


Full method



Full method

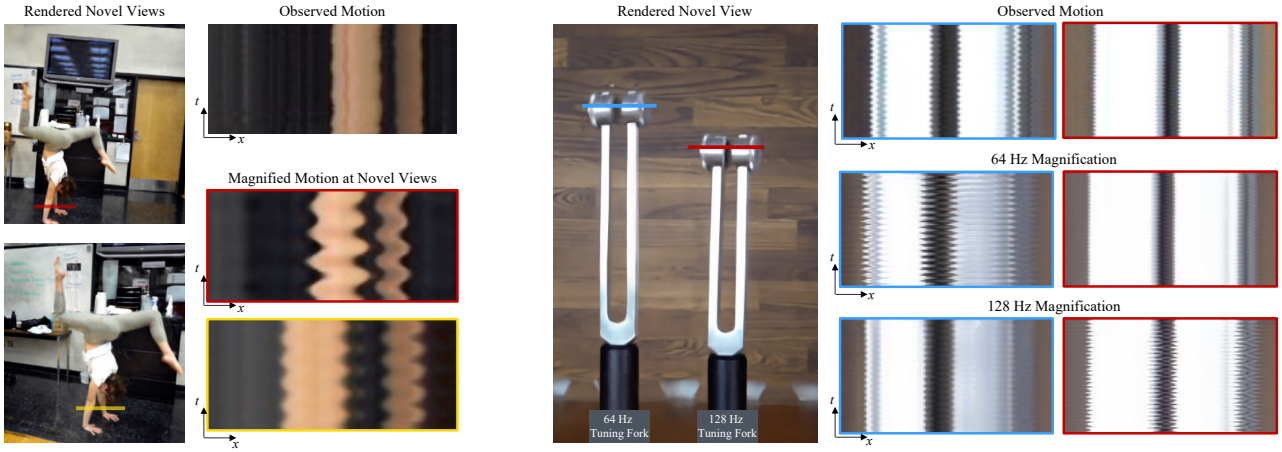
Figure 2.10: **Ablating texture decomposition and cornea pose optimization.** *Left*: not decomposing the iris texture can introduce artifacts in the reconstruction. *Right*: pose optimization is the most critical component for our method to reconstruct the scene.



No L_{radial} With L_{radial}

Figure 2.11: **Ablating radial regularization.** Without radial regularization, the reconstructed iris texture contains parts of the scene with low disparity among observed views.

Chapter 3: 3D Motion Magnification



(a) View synthesis with motion magnification

(b) Frequency-selective magnification with temporal filtering

Figure 3.1: **3D motion magnification**. (a) Novel view synthesis with a gymnast doing a handstand while magnifying the small movements of the arms needed to remain balanced. (b) Motion magnification based on targeted frequencies through temporal filtering, where the left tuning fork vibrates at 64Hz and the right at 128Hz. We visualize x-t (space-time) slices to demonstrate the motion.

3.1 Introduction

We live in a big world of small motions. These motions, such as human respiration or object vibration, are hard to perceive with our naked eyes. Video processing techniques [91, 92, 6] have been developed to extract and magnify subtle motions captured in a 2D video to highlight and visualize those motions. These motion magnification techniques empower visual analytics tools like detecting

*Equal contribution



2D Method [5]



Ours

Figure 3.2: **Motion magnification from a handheld video.** Prior 2D motion magnification approaches (e.g. [5]) cannot handle videos captured by a moving camera, producing severe artifacts. In contrast, our approach can naturally separate camera motion from object motion, allowing us to magnify only the motion of the subject of interest. See Figure 3.7 for our magnified output.

the vibrations of buildings and measuring a person's heart rate using only a video, without the need for physical contact [93–96].

However, we live in a *3D world* full of *3D motions*. Magnifying motion in 3D, as shown in Figure 3.1 allows us to perceive these motions from different views. Furthermore, modeling the motion in 3D provides a natural separation between camera motion and the motion of subjects of interest. This enables magnifying the motion from *handheld videos*, as shown in Figure 3.2. In contrast, prior 2D motion magnification methods catastrophically fail in such scenarios.

In this paper, we propose a method for *3D motion magnification* using neural radiance fields (NeRF), with minimal modifications to standard NeRF backbones and training pipelines. Prior methods designed for 2D videos often leverage the *Eulerian* perspective, which analyzes and amplifies the

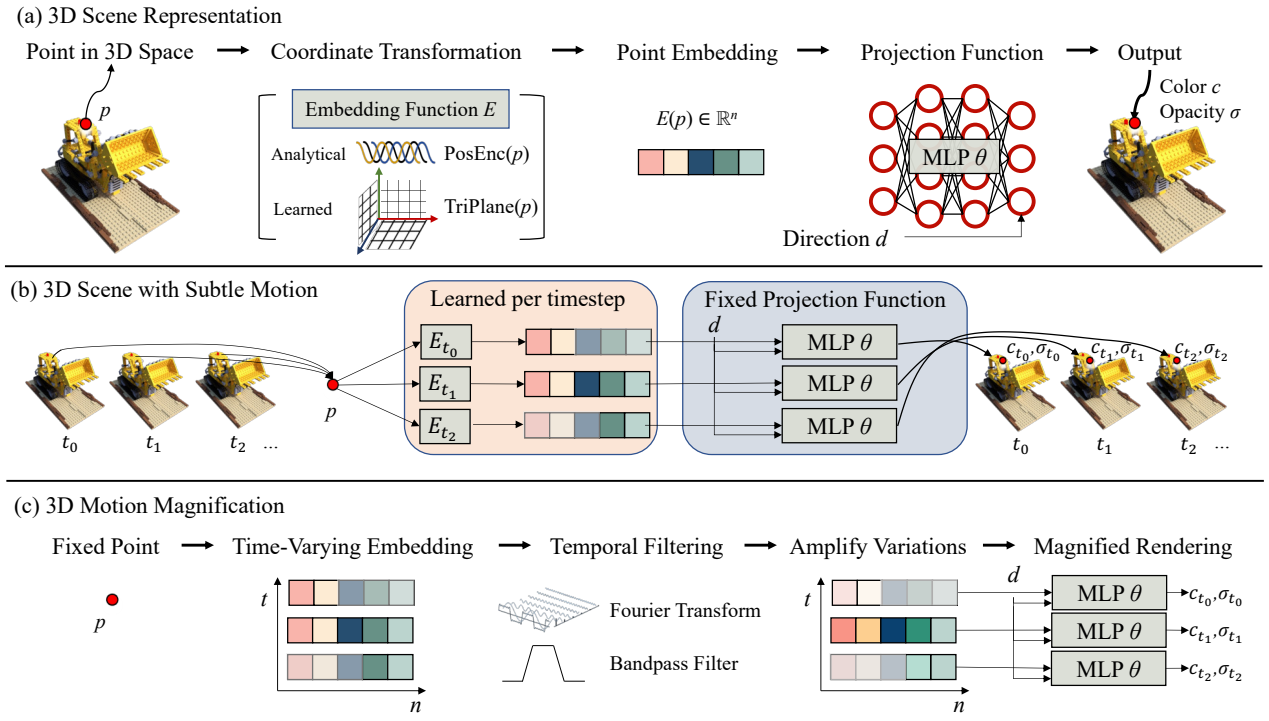


Figure 3.3: **Method overview.** (a) 3D scene representation with NeRF consists of two main components: 1) Coordinate Transformation uses an embedding function E to map the input point $p \in \mathbb{R}^3$ to a high-dimensional embedding vector $E(p) \in \mathbb{R}^n$. The embedding function can be analytical (positional encoding) or learned (tri-plane). 2) The Projection Function θ (usually an MLP) takes in the point embedding and viewing direction, and regresses them into the output color c and opacity σ at p. (b) We study scenes with **subtle motions**. To model the tiny variations with NeRF, we change E over time while fixing the projection function. (c) At a given point p, we analyze its embedding variations over time: $[E_{t_0}(p), ..., E_{t_{T-1}}(p)]$. We perform temporal filtering to isolate and amplify embedding variations within a certain frequency range and then send the amplified embedding to the MLP θ, resulting in motion-magnified 3D rendering.

color variations at each pixel location over time to magnify motion. In contrast, we bring the Eulerian analysis to a new domain beyond color space by designing a magnification method operating on the *feature embeddings* of NeRF. Our experimental results demonstrate that amplifying temporal variations in the *feature embedding of each 3D point* is highly effective in magnifying subtle 3D motion. We observe that magnifying the point embedding provides more accurate and robust magnified renderings than Eulerian magnification performed directly on rendered images.

Using images captured during a time window when *only subtle motion is visible*, we train NeRF to reconstruct the 3D scene with such subtle temporal variations. We ensure that the only element that changes *over time* is the point embedding function, while the MLP layers of NeRF *remain constant over time*. Although the linear Eulerian approach [92] is agnostic to data dimensionality and is extensible to point embeddings of NeRF, for the phase-based Eulerian approach [6], which showed superior properties over the linear approach, it remains unclear how it may be applied for NeRF as it specifically constructs a complex steerable pyramid over each 2D image frame. The recently introduced tri-plane representation for NeRF's embedding function naturally allows for 2D-specific magnification methods like the phase-based approach [97]. Instead of using the analytical *positional encoding* to generate point embeddings, we learn one feature tri-plane at each observed timestep. These tri-planes can be naturally organized as feature videos for 2D video-based magnification methods. Finally, the motion-magnified 3D scene is rendered using these motion-magnified feature triplanes as the point embedding functions.

To evaluate the performance of 3D magnification with NeRF, we first create a synthetic dataset of scenes with subtle motions and measure the magnification quality against synthetically magnified ground truth videos. The phase-based approach operating on tri-plane features leads to the best performance compared to other alternative approaches considered in our experiments. To further validate the practicality of the proposed method, we use our pipeline to process several real-world captured scenes with varying camera setups, scene compositions, and subject motions. Our results show that our proposed approach for 3D motion magnification achieves robust performance for real-world captures in the presence of image noise and camera poses.

To summarize, our contributions are:

- We introduce the problem of 3D motion magnification. We demonstrate the feasibility of applying Eulerian motion analysis for 3D motion magnification using standard NeRF backbones and training pipelines.
- We extend Eulerian analysis to a new domain beyond color space, exploring strategies to modify and filter point embedding and comparing their trade-offs.
- We demonstrate successful 3D motion magnification results on various real-world scenes with different motions, scene compositions, and even handheld videos unsupported by previous 2D methods.

3.2 Related Work

Video motion magnification. Prior approaches to video magnification fall under two categories, inspired by fluid dynamics: Lagrangian [91] and Eulerian [92, 6, 5, 98, 99]. The Lagrangian perspective tracks individual pixels as fluid particles and estimates their motion vectors to warp pixels in the image. Lagrangian-based approach to motion magnification computes the optical flow *explicitly* and uses the estimated flow to magnify the motions of the pixels [91]. The performance, however, is limited by the accuracy of flow estimation. On the other hand, the Eulerian perspective analyzes the changes at fixed pixel locations, amplifying the temporal variations at each pixel/location to magnify motion. This approach bypasses the need for explicit feature tracking or optical flow estimation, which can be inaccurate and costly. Two variants of the Eulerian approach are linear [92] and phase-based [6, 5]. Linear Eulerian [92] constructs Laplacian pyramids over the video frames and amplifies the color variation of each pixel over time. Phase-based Eulerian [6, 5] operates on the phase variations at each pixel, extracted from a complex steerable pyramid [100, 101] decomposition of each video frame.

Later work focuses on magnifying larger motion with affine transform and isolated regions of interest with matting [102], using linear-based methods instead of hand-designed filters [98], and adopting a second-order approximation (with acceleration) instead of first-order methods [99]. Video-based motion magnification has also been applied to extract signals like sound waves from videos recording objects, like a bag of chips, deform and oscillate [103, 104]. Our work builds upon classical Eulerian motion magnification but extends it 1) from 2D to 3D and 2) from color space to the point embedding space of radiance fields. Our results show that the Eulerian principle still holds in the point embedding space.

Static radiance fields. NeRF [46] has become the mainstream approach for representing 3D scenes and demonstrates high-quality view synthesis results. Various techniques have been introduced to improve NeRF in several aspects, including training and rendering acceleration [105, 72, 71, 106–108], reducing aliasing [58], unbounded scene modeling [59, 109], and optimizing poses [64, 110]. Factor Fields [111] present a unified framework summarizing various NeRF variants and other neural signal representations as mainly composed of two components: (1) a *Coordinate Transformation* that maps input coordinates into an embedding space, and (2) a *Projection Function* that maps the embeddings into a value in the field. In this paper, we adopt a similar perspective and focus on analyzing the relationship between point embedding and subtle motions. We propose magnifying subtle motions through Eulerian magnifications of point embeddings in NeRF. We demonstrate successful applications of this approach on NeRF with both positional encoding-based embedding [46, 105] and tri-plane embedding [97, 106].

Dynamic scene representations. Extensive research has been devoted to extending NeRF for modeling dynamic scenes. One line of work learns a deformation field and uses it to warp a canonical

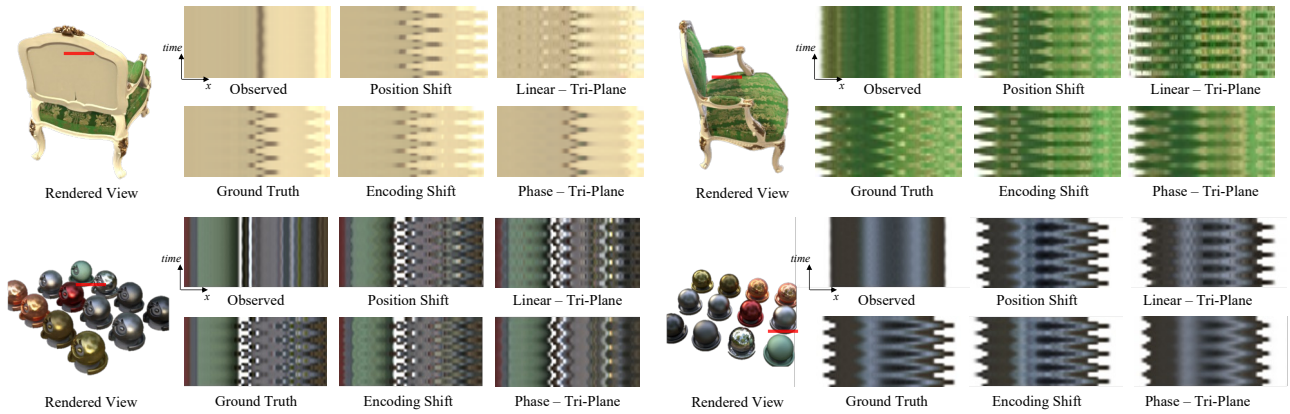


Figure 3.4: **3D motion magnification on synthetic scenes.** We generated each synthetic scene by periodically vibrating object parts. We magnify the subtle motion encoded in NeRF reconstruction using the approaches discussed in Sec. 3.4, and visualize the motion here as a 2D space-time slice image. The corresponding location of each space-time slice is indicated by a red line on the rendered view. All four approaches successfully capture and magnify the motion, although the linear Eulerian approach, *Linear - Tri-Plane*, is more prone to intensity overshooting [6], manifested as bright and dark spots in the space-time slice.

NeRF for each timestep [61, 62, 60, 112, 113]. Alternatively, one can directly learn a space-time radiance field with time as an additional coordinate [114–118, 63, 106, 119, 120]. A major challenge of reconstructing dynamic scenes is capturing time-synchronous multi-view observations. While a multi-camera setting is ideal for acquiring high-quality data [121–123], researchers have explored the more challenging but practical setting of single-camera captures, leveraging priors such as consistent depth [115, 124, 125], optical flow [114, 116, 63], or human prior [7, 126]. We demonstrate the applicability of 3D motion magnification in both multi-camera and single-camera setups.

Implicit representations. Implicit representations have emerged as powerful tools for modeling signals [127–137]. Mai and Liu [138] study *2D videos* with implicit neural representation and model motions in videos by learning *spatially invariant* phase shifts in the positional encoding function. Our experiment on magnification based on position encoding functions shares similar ideas. However, unlike Mai and Liu [138], we learn *spatially varying* phase shifts and magnify *3D motion*.

3.3 Preliminaries

Eulerian motion magnification. The Eulerian-based motion analysis focuses on the changes at a *fixed* spatial location over time instead of tracking a specific particle (pixel). The *linear Eulerian approach* converts the color variation over time at each pixel into a 1D vector, using the Fourier transform to obtain its temporal frequency components, and filters the frequencies corresponding to the desired motion. The color intensity changes within the desired frequency range are then amplified and added back to the original values to create a motion-magnified video (where subtle motions become more visible).

To offer an intuition on why amplifying per-pixel color intensity could magnify motion across the frame, let $f(x,t) = g(x + \delta(t))$ denote a signal with motion over time described by the shift $\delta(t)$. The first-order Taylor series expansion of $g(x + \delta(t))$ about the point x can be written as:

$$g(x) + g'(x)(x + \delta(t) - x) = g(x) + g'(x)\delta(t).$$

With observations at multiple timesteps t, we can easily filter out the static $g(x)$ term and keep the dynamic term $g'(x)\delta(t)$. If we multiply $g'(x)\delta(t)$ by α and add it back to the original signal, we get

$$g(x) + (1 + \alpha)g'(x)\delta(t) \approx g(x + (1 + \alpha)\delta(t)),$$

which is equivalent to magnifying the motion by α.

The *phase-based Eulerian approach* amplifies phase variations over time instead of color amplitude variations. The phase here is extracted from a complex steerable pyramid constructed from

the original frames. The connection between phase and motion can be established through the Fourier shift theorem: if a function $f(x)$ is shifted by a distance δ in its domain, it would be equivalent to multiplying its Fourier component $\mathcal{F}(k)$ by a phase factor $e^{-i2\pi k\delta}$:

$$\mathcal{F}\{f(x-\delta)\}(k) = \mathcal{F}\{f(x)\}(k)e^{-i2\pi k\delta},$$

where $\mathcal{F}$ denotes the Fourier transform operator and k denotes the frequency component. In other words, extracting the phase changes over time reveals the motion-induced pixel shift in space by δ. After amplifying the phase changes, the motion-magnified signal can be generated with an inverse Fourier transform.

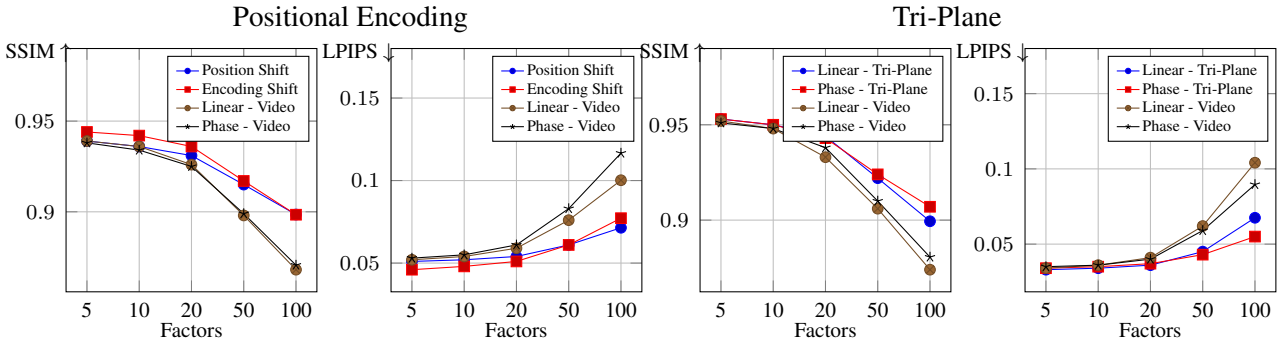


Figure 3.5: **Quantitative comparison.** We evaluate the quality of motion-magnified renderings as a function of the magnification factor used, using positional encoding (*Left*) and tri-plane (*Right*) as the point embedding function. With positional encoding, we evaluate two approaches to vary point embedding through phase shifts in the sine waves: *Position Shift* (shifting each 3D point) and *Encoding Shift* (shifting each frequency). With tri-plane, we evaluate two approaches to vary learned point embeddings: *Linear - Tri-Plane* (linear magnification on tri-plane) and *Phase - Tri-Plane* (phase-based magnification on tri-plane). For both embedding functions, we compare against two baseline methods for video motion magnification: *Linear - Video* (linear magnification on the NeRF-rendered video) and *Phase-Video* (phase-based magnification on the NeRF-rendered video). Results from two embedding functions are separated to enable better assessments of the impact of different magnification approaches and avoid confounding with the inherent performance gap between different embedding functions and MLP architectures.

Neural radiance fields as 3D scene representations. NeRF models the radiance in a scene as a continuous function, which takes as input a 3D spatial coordinate $p \in \mathbb{R}^3$ and a viewing direction $d \in \mathbb{S}^2$, and outputs the radiance color c (observed from viewing direction d) and density σ at that point. Notably, the spatial coordinate p is transformed into a feature representation through some embedding function E, before a projection function (MLP) regresses it into the final prediction:

$$f(p, d) = \text{MLP}(E(p), d) = (c, \sigma).$$

With subtle and unknown scene motions, we assume this time-varying scene can be formulated as $f(p + \delta(p, t), d)$. If MLP is fixed across time, then the unknown motion $\delta(p, t)$ can be recovered by analyzing the temporal variations of $E(p, t)$. However, where do we access $E(p, t)$? Whereas in the 2D video case, the data of interest is directly recorded by a camera and is available for analysis, here we only have access to a collection of 2D images that may have observed the 3D subtle motions during capture. In the following subsection, we discuss how to reconfigure NeRF to model subtle 3D motions by varying the function $E(p, t)$.

3.4 Method

We assume the availability of: 1) Multi-view observations to reconstruct a static NeRF, and 2) video recording of the subtle scene motions, either with a time-synchronized multi-camera setup or a single moving camera.

The general workflow of our method is as follows: 1) We train a static NeRF from image observations that can be assumed as motionless. 2) For each timestep $t \in [0, T-1]$ in the video observations, we finetune the embedding function E_t so that the NeRF rendering matches with the observations at

t. 3) After finetuning all T embedding functions E_t, we magnify motions by amplifying the temporal variations of each sampled point used in NeRF rendering.

In this section, we describe how we repurpose NeRF to capture subtle motions and perform magnification by analyzing the point embeddings learned by NeRF. We begin our discussion with the base case of the standard NeRF with positional encoding as the point embedding function. We then describe our preferred approach with tri-plane as the embedding function for NeRF, which leads to a natural integration with the phase-based Eulerian magnification technique previously designed for videos.

3.4.1 NeRF with Positional Encoding

We first describe how Eulerian magnification in the embedding space can be achieved on standard NeRF with positional encoding. Motivated by prior work on motion-adjustable neural representations for video [138], we keep the main backbone of NeRF intact and separately train a small MLP g that learns to apply phase shifts in the positional encoding functions. However, different from prior work [138], with video observations from the scene, we organize the images by their captured time and train a separate MLP g for each timestep. Effectively, g learns to adjust the embedded representation of each point so that the NeRF output from the projection function (MLP θ) is consistent with the time-varying observations. Weights of the MLP θ are shared across all time steps, and the only difference lies in the point embeddings. There are two options to induce phase shifts to the positional encoding function: position shift and encoding shift.

Position shift. To model motion exclusively through changing the point embeddings, we let g directly **predict the 3D position shift of the queried point** p: $g(p, t) = \Delta p \in \mathbb{R}^3$. We add Δp to p

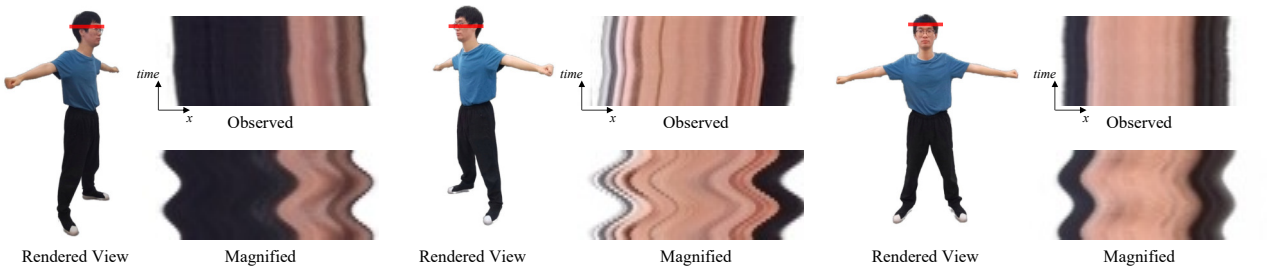


Figure 3.6: **Real-world multi-view motion magnification**. Using multi-view videos from the HumanNerf dataset [7], we can capture and magnify true 3D motion. We visualize the motion here as a 2D space-time image, where the corresponding location of each space-time slice is shown on the rendered view as a red line.

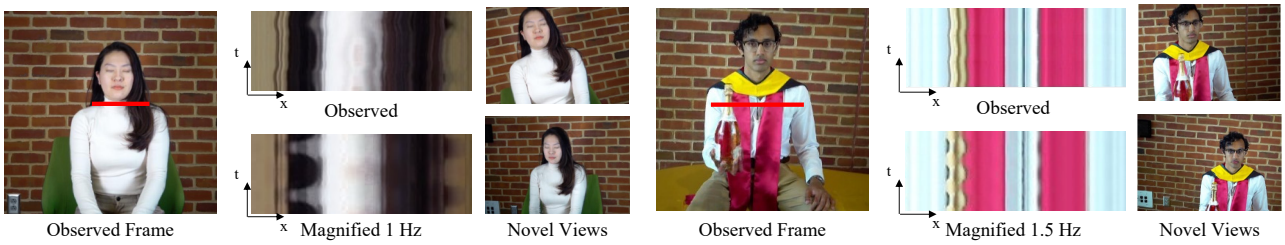


Figure 3.7: **Real-world single-camera motion magnification**. Despite the monocular view of the dynamic scene, we can render novel views while magnifying subtle motion. We visualize the motion as a 2D image slice through time.

before applying positional encoding, obtaining a time-varying point embedding function

$$E(p,t) = \text{PosEnc}(p + g(p,t)),$$

which is equivalent to applying a phase shift of $\omega \cdot g(p,t)$ within each sine wave $\sin(\omega x)$ used in positional encoding. Note that the phase shifts for all K frequencies have the same direction and only differ in magnitude, which is scaled by different $\omega \in [1, ..., K]$.

Encoding shift. Note that motion does not only lead to geometric changes in the 3D scene since it would also cause appearance changes like shadows and reflections. Therefore, attributing all the scene variation to shifts in 3D position is not sufficient. Instead, we may let g learn **a separate shift for each**

encoding frequency. In other words, with $\phi_\omega \in \mathbb{R}^3$, let $g(p,t) = [\phi_1, \phi_2, ..., \phi_K] \in \mathbb{R}^{3\times K}$, and the point embedding function becomes

$$E(p,t) = [\sin(\omega_1 p + \phi_1), ..., \sin(\omega_K p + \phi_K)].$$

This setup treats positional encoding as a feature generator that produces an embedding with $3K$ channels. As shown in later sections, this "motion-agnostic" approach still learns to capture the true motion while outperforming the approach that only accounts for position shifts.

With either of these two approaches to vary the point embeddings in a standard NeRF with positional encoding, we can render magnified motions by linearly amplifying the temporal variations of $g(p,t)$, and then rendering the point's color and opacity with NeRF.

3.4.2 NeRF with Tri-plane Learnable Embedding

We now describe our preferred approach using tri-plane as the embedding function for NeRF. Later experiments suggest that the tri-plane-based approach achieved better magnification quality than the positional encoding approach. Tri-plane [97, 106] has been recently proposed as an efficient way to obtain learnable embedding for points in NeRF rendering. Compared to NeRF with the analytical positional encoding, NeRF with tri-planes as the point embedding function can achieve similar representation capacity with far fewer MLP layers and, thus, faster inference.

In our use case of NeRF, the tri-plane formulation has a nice implication of reducing the 3D scene into a collection of 2D feature planes. Such a decomposition preserves the relative spatial relationship between points, instead of randomly hashing points into features. This observation suggests we may achieve 3D motion magnification by directly processing the feature planes and potentially

outperforming the aforementioned linear magnification within the positional encoding function.

Specifically, we train a separate tri-plane embedding function for each timestep while the MLP-based projection function is shared across time. With this setup, all subtle temporal changes in the scene (motion or appearance) would need to be encapsulated in the temporal changes of the 2D feature images of the tri-plane. The point embedding function here can be written as

$$E(p, t) = \text{Project}(p, \text{TriPlane}_t),$$

where the embedding of p is obtained by projecting p onto the tri-plane and aggregating the corresponding features.

With a separate tri-plane constructed for each timestep, we establish a key connection to prior video magnification methods: we essentially obtain a *video* for each tri-plane feature channel, on which we could either apply **linear magnification** and amplify the temporal changes of each pixel in the feature image, or **phase-based magnification** with complex steerable pyramids constructed over each channel of the 2D feature image. Our experimental results confirm the feasibility of this 2D-inspired approach. The performance comparison in Section 3.5.2 between the linear and phase-based approaches also validates the original findings from when these two approaches were applied to perform Eulerian processing of color spaces.

3.5 Experimental Results

We evaluate the performance of the proposed method using synthetic scenes in Section 3.5.2. We create ground truth sequences for the magnified motion and quantitatively compare the different approaches for 3D motion magnification. We then present our results on real-world captured data

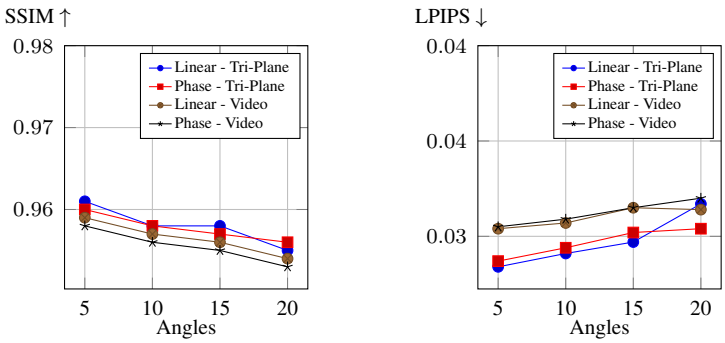


Figure 3.8: **Varying the angles of deviation from observed views**. As the deviation angle increases, magnifying through the embedding space consistently outperforms the baseline approaches that operate on the color space.

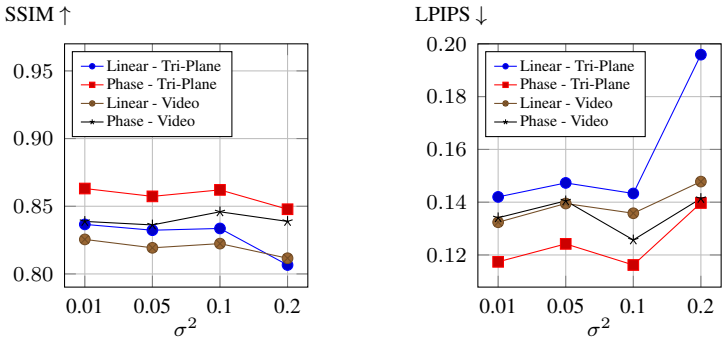


Figure 3.9: **Varying noise levels in training views**. When we perform Eulerian magnification on the tri-plane embedding space, *Phase - Tri-Plane* is more robust than *Linear - Tri-Plane* in the presence of noise. The finding is analogous to previous analysis [6] on color space magnification (validated by the two baseline results: *Linear - Video* and *Phase - Video*).

in Section 3.5.3. We first deploy our method on real-world multi-view video observations. After validating its effectiveness on real-world multi-view data, we further apply our method to real-world video sequences captured using a *single-camera*. As a result of extending motion magnification to 3D, our method successfully magnifies 3D motions from handheld-captured videos with camera shake, a scenario unattainable by prior work that focused on stabilized 2D videos.

3.5.1 Implementation Details

Positional encoding as point embedding. We train a 3-layer MLP with 32 hidden channels to predict $g(p, t)$ and apply the resulting phase shifts in positional encoding. We implement the network with nerfacc [105]. To train a static NeRF for the first timestep, we optimize for 50,000 steps, and for the remaining timesteps, we finetune the embedding function for 10,000 steps. After training, for each point p, we obtain its time-varying phase shifts $g(p, t)$. To render with magnified motion within a time window $[0, ..., T-1]$, we Fourier transform $[g(p, 0), ..., g(p, T-1)]$ along the time dimension, use a bandpass filter to isolate the motions within the frequency range of interest, amplify the components within the passband range, and then apply inverse Fourier transform to obtain the magnified predictions. The magnified predictions are added inside the positional encoding as phase shifts, followed by the standard NeRF rendering with MLP inference.

Tri-plane as point embedding. Our implementation builds on K-Planes [106]. To aggregate the embedding from different planes, we adopt concatenation instead of the default Hadamard product, and we only set the triplanes at a single scale for simplicity. To train a static NeRF for the first timestep, we optimize for 30,000 steps, and for the remaining timesteps we finetune the embedding function for 10,000 steps. After training, we compose a video for each feature plane within $[0, ..., T-1]$. Then, we apply 2D magnification methods directly on these *feature videos*. For the *linear* approach, we temporally filter and amplify the feature value variations within a frequency range. For the *phase-based* approach, we construct a complex steerable pyramid over the feature image, temporally filter and amplify the phase variations, and then collapse the pyramid back into the image space; the resulting feature image would exhibit magnified motions [6]. To produce motion-magnified rendering,

we embed each sample point with the processed triplane features and then use the MLP to project the embedding into color and density output as usual.

3.5.2 Synthetic Scenes

Data Generation. We use the standard Blender scenes [46] and simulate motions in different object parts. We render each scene for one second at 30 frames per second. The simulated motions are periodic, ranging from 3 Hz to 5 Hz. We also render sequences with ground truth magnified motions under factors 5, 10, 20, 50, and 100.

Qualitative Evaluation. We magnify motions in the Blender scenes with the approaches described in Sec. 3.4. Results in Figure 3.4 show that all four approaches successfully model and magnify tiny motions in the observations. Consistent with previous findings in video magnification [6], the linear Eulerian approach leads to more artifacts, such as clipped color intensities and noise amplification.

Quantitative Evaluation. We evaluate the results against ground truth magnified frames using structural similarity index measure (SSIM) [139], and LPIPS [140] with AlexNet [141] as the backbone. We also render a video from the trained NeRF without magnification for baseline comparisons at each test view. We then apply classical 2D methods directly on the video to magnify subtle motions. As shown in Figure 3.5, our 3D magnification methods outperform the 2D baseline methods in producing motion-magnified renderings consistent with the ground truth renderings.

Sensitivity Aanalysis. In Figure 3.8, we analyze magnification quality as the test viewpoints deviate from the observed viewpoints. In Figure 3.9, we plot magnification quality as noise levels increase in the captured frames. Previous work [6] on color space magnification has found that the phase-based approach is less sensitive to noise than the linear approach; we observe similar phenomenon during

our magnification in the embedding space.

3.5.3 Real-world Scenes

We test our methods on several real-world scenes captured with different camera setups.

Multi-Camera Setup. We first validate our method on the publicly available dataset from HumanNerf [121], comprising short videos from six cameras simultaneously capturing a scene with a person standing in the center. We extract a brief period where the person is relatively static but still exhibits subtle body movements. In Figure 3.6, we present the magnified results from different viewpoints.

Single-Camera Setup. As a multi-camera setup may be prohibitively inconvenient and expensive for many users, we further deploy our method on a single-camera setup. We design a capture procedure that consists of two stages. Step 1: we first capture a moving camera video of the static scene, which will be used to train a static NeRF. Step 2: we capture a single-view video of the dynamic scene, which is used to finetune the point embedding function in NeRF to model the time-varying scene. After the two-stage capture, we perform NeRF training and render magnified motions using the previously described pipeline. This two-step capture approach is prevalent in video-based motion magnification application scenarios, where people identify unwanted motion in static civil structures under normal conditions [95, 96]. We also highlight that our method **supports handheld capture and does not require a tripod**, unlike previous 2D-based approaches that would fail without a steady capture, as shown in Figure 3.2. This is possible since our method uses the estimated poses of each frame independently when updating the radiance fields on the dynamic sequence. Hence, an accurate pose estimation removes the need for a perfectly still capture. We show the results using monocular capture in Figure 3.1 of a gymnast holding a handstand and in Figure 3.7 on a person trying to balance on one

leg and a person breathing. We also show observed space-time slices compared to our magnified ones with NeRF reconstruction.

3.6 Limitations

Data captured in real-world environments can be blurry due to defocus and camera shake, degrading the quality of NeRF. The training of NeRF assumes the knowledge of camera poses, so its performance depends heavily on the accuracy of pose estimation, often using RGB-based structure-from-motion (SfM) algorithms [142, 143]. The estimations are mostly reliable but not flawless due to lens distortions that require specific calibrations, which may not be performed by common pipelines for NeRF-based 3D reconstruction. More importantly, inaccurate pose estimation would exacerbate the ambiguity between *camera motion* and *scene motion*, which could hinder magnifying subtle scene motions or lead to false motion magnification. Therefore, real-world data should be captured under conditions where accurate camera intrinsic and extrinsic parameters are accessible, either from *reliable* RGB-based SfM with textured surfaces in the scene, or from cameras that support 6-DoF tracking during capture.

3.7 Conclusion

We present a 3D motion magnification method that applies Eulerain processing principles to analyzing NeRF embeddings over time. While classical magnification methods (developed originally for 2D videos) process pixel colors directly, we show that processing the point embeddings of NeRF successfully generalizes those approaches and allows magnifying motions in 3D renderings. We believe our work will motivate further research towards integrating traditional signal processing techniques

into neural fields.

Chapter 4: Defocus Control using Dual-Camera



Figure 4.1: **Post-capture depth-of-field (DoF) control from dual camera.** (*Left*) Using photos captured from dual cameras with a wide field of view (FoV) and ultra-wide FoV, our method enables various DoF manipulations. (*Right*) We showcase our results of refocusing (changing the focal plane), deblurring (creating all-in-focus imagery), and synthesizing a shallower DoF (producing bokeh effects).

4.1 Introduction

Smartphone cameras are the most common modality for capturing photographs today [144]. Recent advancements in computational photography such as burst photography [145], synthetic bokeh via portrait mode [9], super-resolution [146], and more have been highly effective at closing the gap between professional DSLR and smartphone photography. However, a key limitation for smartphone cameras today is depth-of-field (DoF) control, i.e., controlling parts of the scene that appear in (and

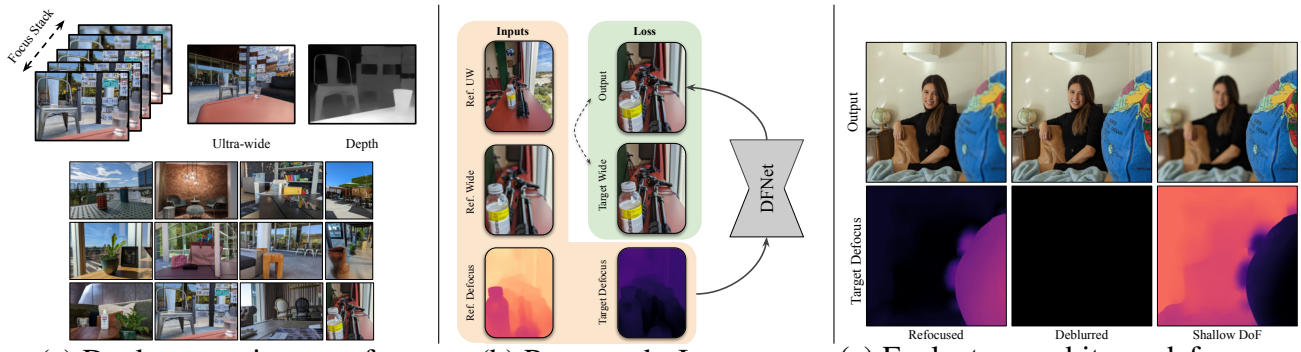
out of) focus. This is primarily an artifact of their relatively simple optics and imaging systems (e.g., fixed aperture, smaller imaging sensors, etc.). To bridge the gap, modern smartphones tend to computationally process the images for further post-capture enhancements such as synthesizing shallow DoF (e.g., portrait mode [9, 11]). However, this strategy alone does not allow for DoF *extension* or post-capture refocus. In this work, we propose Dual-Camera Defocus Control (DC^2), a framework that can provide post-capture **defocus control** leveraging multi-camera systems prevalent in smartphones today. Figure 4.1 shows example outputs from our framework for various post-capture DoF variations. In particular, our method is controllable and enables image refocus, DoF extension, and reduction.

Post-capture defocus control is a compound process that involves removing defocus blur (i.e., defocus deblurring) and then adding defocus blur selectively based on the scene depth. Defocus deblurring [147, 148, 10, 149–158], itself, is challenging due to the nature of the defocus point spread function (PSF) formation which can be spatially varying in size and shape [159, 160]. The PSF's shape and size are not only depth dependent, but also vary based on aperture size, focal length, focus distance, optical aberration, and radial distortion. Synthesizing and adding defocus blur [161, 162, 9, 163–167, 11] is also difficult and requires an accurate depth map along with an all-in-focus image. Additionally, it requires realistic blur formation and blending around the object's boundaries. Most prior work has addressed defocus deblurring and synthesizing defocus blur as two isolated tasks. There has been less work on post-capture defocus control (e.g., image refocusing [168–170]). The image refocusing literature [169, 170] has focused on light-field data captured with specialized hardware. While the results in [171, 172] are the state-of-the-art, light-field data is not representative of smartphone and DSLR cameras by lacking realistic defocus blur and spatial resolution [173].

Most modern smartphones are now equipped with two or more rear cameras to assist with computational imaging. The primary camera – often referred to as the wide camera or **W** – has a higher

resolution sensor, a higher focal length lens but a relatively shallower DoF. Alongside **W** is the ultra-wide (**UW**) camera, often with a lower resolution sensor, lower focal length (wider field of view) and wider DoF. Our critical insight is to leverage this unique camera setup and cross-camera DoF variations to design a system for realistic post-capture defocus control. Differently from prior work, we tackle the problem of defocus control (deblurring *and* adding blur) and propose using real-world data easily captured using a smartphone device to train our learning-based system. Our primary contributions in this work are as follows:

- We propose a learning-based system for **defocus control** on dual-camera smartphones. This subsumes the tasks of defocus deblurring, depth-based blur rendering, image refocusing and enables arbitrary post-capture defocus control.
- In the absence of defocus control ground-truth, we enable training our system on real-world data captured from a smartphone device. To achieve that, we re-formulate the problem of defocus control as learning to refocus and define a novel training strategy to serve the purpose.
- We collect a dataset of diverse scenes with focus stack data at controlled lens positions the **W** camera and accompanying **UW** camera images for training our system. Additionally, we compute all-in-focus images using the focus stacks to quantitatively evaluate image refocus, defocus deblurring and depth-based blurring tasks and demonstrate superior performance compared to state-of-the-art (SoTA) methods across all three tasks.
- Finally, we demonstrate creative defocus control effects enabled by our system, including tilt-shift and content-based defocus.



(a) Dual camera image refocus dataset.

(b) Proxy task: Learn to refocus.

(c) Evaluate on arbitrary defocus maps.

Figure 4.2: **Image refocus as a proxy task.** Since we cannot gather a *real* dataset for arbitrary focus manipulation, our idea is to train a model to perform *image refocus* using a target defocus map as an input. At the test time, our trained model can perform arbitrary focus manipulation by feeding it an arbitrary target defocus map.

4.2 Related Work

Defocus Deblurring. Defocus blur leads to a loss of detail in the captured image. To recover lost details, a line of work follows a two-stage approach: (1) estimate an explicit defocus map, (2) use a non-blind deconvolution guided by the defocus map [147, 148]. With the current advances in learning-based techniques, recent work perform single image deblurring directly by training a neural network end-to-end to restore the deblurred image [10, 149–153]. Due to the difficulty of the defocus deblurring task, other works try to utilize additional signals, such as the dual pixel (DP) data to improve deblurring performance [154–158]. DP data is useful for deblurring as it provides the model with defocus disparity that can be used to inform deblurring. While the DP data provides valuable cues for the amount of defocus blur at each pixel, the DP views are extracted from a single camera. Therefore, the performance of the DP deblurring methods drops noticeably and suffer from unappealing visual artifacts for severely blurred regions.

In the same vein, we aim to exploit the **UW** image as a complementary signal already available in

modern smartphones yet ignored for DoF control. By using the **UW** image with different DoF arrangements, we can deblur regions with severe defocus blur that existing methods cannot handle because of the fundamental information loss. Nevertheless, we are aware that using another camera adds other challenges like image misalignment, occlusion, and color mismatches which we address in Section 4.4.3.

Bokeh Rendering. Photographers can rely on shallow DoF to highlight an object of interest and add an artistic effect to the photo. The blur kernel is spatially variant based on depth as well as the camera and optics. To avoid the need of estimating depth, one work magnifies the existing defocus in the image to make the blur more apparent without explicit depth estimate [174]. Since recent work in depth estimation improved significantly [175, 176], many shallow DoF rendering methods assume having depth [11] or estimate depth in the process [9, 165]. Using an input or estimated depth map, a shallow DoF can be synthesized using classical rendering methods [161, 162, 9, 163], using a neural network to add the synthetic blur [177, 166, 167] or a combination of classical and neural rendering [11]. With that said, shallow DoF synthesis methods typically assume an all-in-focus image or an input with a deep DoF.

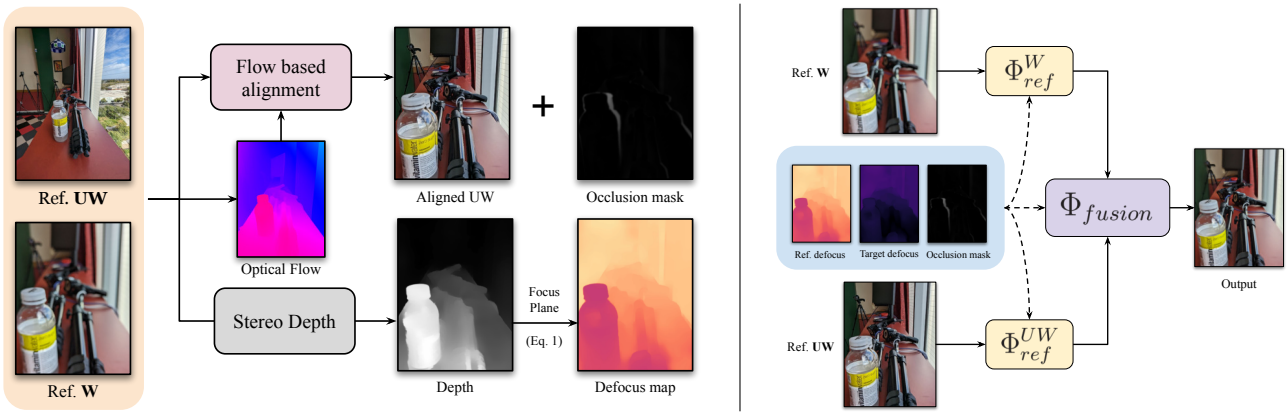
Our proposed framework learns to blur as a byproduct of learning to refocus with the insight that the refocus task involves both deblurring and selective blurring. Unlike prior work that addressed either defocus deblurring or image bokeh rendering, we introduce a generic framework that facilitates post-capture full defocus control (e.g., image refocusing).

Image Refocus and DoF Control. At capture time, the camera focus can be adjusted automatically (i.e., autofocus [178–180]) or manually by moving the lens or adjusting the aperture. When the image is captured, it can still be post-processed to manipulate the focus. Nevertheless, post-capture image

refocus is challenging as it requires both deblurring and blurring. Prior work uses specialized hardware to record a light field which allows post-capture focus control [169, 181]. However, light field cameras have low spatial resolution and are not representative of smartphone cameras. An alternative to requiring custom hardware is to capture a focus stack, and then merge the frames required to simulate the desired focus distance and DoF [170, 182–184], but the long capture time restricts using focus stacks to static scenes. Research on single-image refocus is limited due to its difficulty, but the typical approach is to deblur to obtain an all-in-focus image followed by blurring. Previous work used classical deblurring and blurring [185] to obtain single image refocus, and the most notable recent single-image-based image refocus is RefocusGAN [168], which trains a two-stages GAN to perform refocusing. The limited research on software-based image refocus is likely due to the challenging task that involves both defocus deblurring and selective blurring. In our work, we provide a practical setup for post-capture image refocus without the restrictions of inaccessible hardware or the constraint of capturing a focus stack. We do so by leveraging the dual camera that is available in modern smartphones.

Image Fusion. Combining information from images with complementary information captured using different cameras [182, 186] or the same camera with different capture settings [187, 145] can enhance images in terms of sharpness [170, 186, 182], illuminant estimation [188], exposure [187, 145, 182, 189], or other aspects [186, 190–192]. With the recent prevalence of dual-camera smartphones today, researchers have pursued works that target this setup. One line of work has used dual-camera for super-resolution to take advantage of the different resolutions the cameras have in still photos [171, 193, 194] as well as in videos [195]. The dual-camera setup has also been used in multiple commercial smartphones, e.g., Google Pixel devices to deblur faces by capturing an ultra-wide image with faster shutter time and fusing with the wide photo [196]. To our knowledge, we are the first to investigate

using the dual-camera setup for defocus control.



(a) Data processing (b) DFNet

Figure 4.3: **Data processing and high-level architecture.** (*Left*) To be able to use the reference inputs for our Detail Fusion Network, we need to align the inputs and a depth estimate to approximate the defocus map of the reference **W** and the target defocus map we would like to synthesize. We use flow-based alignment with PWCNet [8] and use the stereo depth estimated using portrait mode [9]. (*Right*) Our Detail Fusion Network (DFNet) consists of refinement modules to refine the reference inputs combined with a fusion module that predicts blending masks to combine the two refined inputs.

4.3 Learning to Refocus as a Proxy Task

As mentioned, smartphone cameras tend to have fixed apertures limiting DoF control at capture time. In our work, we aim to unlock the ability to synthetically control the aperture - by transferring sharper details where present and synthesizing realistic blur. However, to train such a model, we run into a chicken and egg problem: we require a dataset of images captured with different apertures, which isn't possible with smartphones. An alternative solution could be to generate such a dataset synthetically, but modeling a realistic point spread function (PSF) for the blur kernel is non-trivial [154]. Professional DSLRs provide yet another alternative [197] but often require paired captures smartphone / DLSR captures to reduce the domain gap. Ideally, we would like to use the same camera system for both training and evaluation. To resolve this, we observe that a somewhat parallel task is

image refocus. When we change the focus distance, the defocus radius is adjusted in different parts of the image, involving a combination of pixels getting deblurred and blurred. This suggests that image refocus is at least as hard as scaling the DoF. Motivated by this observation, we make the hypothesis that by training a model on image refocus as a *proxy task*, we can use the same model to control the DoF at test time as we show in Figure 4.2. The key idea is to provide the model with reference and target defocus maps (Section 4.4.1) as input, and at test time control the model behavior by manipulating this target defocus map.

4.4 Method

To train a model on our proxy task, we need to collect a dataset of focus stacks for the wide camera and a paired ultra-wide frame which can be used as a guide due to its deeper DoF. In Figure 4.3 we show the high-level structure of our method dubbed DC^2. The primary module that we train is the Detail Fusion Network (DFNet), which requires a reference wide frame, (aligned) reference ultra-wide frame, and estimated defocus maps. In Section 4.4.1, we describe how we collect the focus stack data and process it to obtain the inputs needed for DFNet. We then describe the architecture details of DFNet in Section 4.4.2, which is motivated by the dual-camera input setup.

4.4.1 Data Processing

Using the Google Pixel 6 Pro as our camera platform, we captured a dataset of 100 focus stacks of diverse scenes, including indoor and outdoor scenarios. For each scene, we sweep the focus plane for the wide camera and capture a complete focus stack. We simultaneously capture a frame from the ultra-wide camera, which has a smaller aperture, deeper DoF, and fixed focus. For each frame, we use

optical-flow-based warping using PWCNet [8] and following prior work [196] to align the ultra-wide frame with the wide frame. Since the alignment is imperfect (e.g., in textureless regions and occluded boundaries), we estimate an occlusion mask that can be used to express potentially misaligned regions for the model. To estimate defocus maps, we require the metric depth. We use the depth map embedded in the Pixel camera's portrait mode output which can estimate metric depth using dual camera stereo algorithms [198] with a known camera baseline. To compute the defocus map associated with each frame, we use the following formula for the radius of the circle of confusion c

$$c = A\frac{|S_2 - S_1|}{S_2}\frac{f}{S_1 - f} \tag{4.1}$$

where A is the camera aperture, S_1 is the focus distance, S_2 is the pixel depth, and f is the focal length. In Figure 4.2a, we show a visualization of a focus stack, associated **UW**, stereo depth, and a collection of sample scenes.

4.4.2 Model Architecture

Our method performs detail fusion on two primary inputs: the reference wide (**W**) and ultra-wide (**UW**) images. Since we train the model to refocus, **W** is expected to be treated as a base image, while **UW** is a guide for missing high-frequency details.

Based on this intuition, we propose **Detail Fusion Network (DFNet)** that has two refinement paths: **W** refinement path (Φ^{W}_{ref}), **UW** refinement path (Φ^{UW}_{ref}), and a fusion module (Φ_{fusion}) that predicts blending masks for the refined **W** and refined **UW**. Note that the **W** refinement path never gets to see the **UW** frame and vice versa. We use a network architecture based on Dynamic Residual Blocks Network (DRBNet) [10] for our refinement modules with multi-scale refinements. For the

fusion module, we use a sequence of atrous convolutions [199] for an increased receptive field and predict a blending mask for each scale. To preserve high-frequency details in the blending mask, we add upsampling layer and residual connections when predicting the blending mask of the larger scale. During training, we blend the outputs of Φ_{ref}^{W} and Φ_{ref}^{UW} and compute the loss for all scales for improved performance. In Figure 4.3 we show a high-level diagram of our architecture and how each component interacts with the others. By visualizing the intermediate outputs between our different modules, we observe that the network indeed attempts to maintain the low-frequency signal from**W** while utilizing high-frequency signals from **UW**.

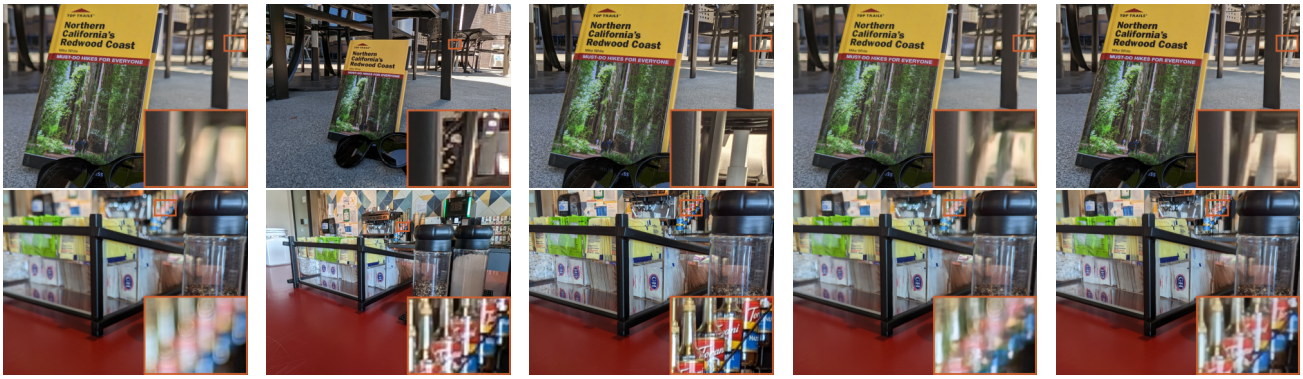
4.4.3 Training Details

We train our model by randomly sampling slices from the focus stack in our training scenes. For each element in the batch, we randomly sample a training scene, and sample two frames to use as reference and target images, respectively. While we can approximate depth from all pairs, severely blurry frames can have unreliable depth. To address that, we use the stereo pair with the greatest number of matched features to use for the scene depth to compute the defocus maps. We train on randomly cropped 256x256 patches, using a batch size of 8, and a learning rate of 10^{-4} for 200k iterations, and then reduce the learning rate to 10^{-5} for another 200k iterations using Adam [200]. Our reconstruction loss is a combination of L_1 loss on pixels and gradient magnitudes, SSIM loss [139], and perceptual loss [140]. For a target wide frame $\mathbf{W}_{tgt}$ and a model output y, the loss is

$$\begin{aligned} L_{total} = L_1(\mathbf{W}_{tgt}, y) + L_1(\nabla\mathbf{W}_{tgt}, \nabla y) \\ L_{SSIM}(\mathbf{W}_{tgt}, y)) + L_{VGG}(\mathbf{W}_{tgt}, y) \end{aligned} \tag{4.2}$$

Table 4.1: **Defocus deblurring evaluation.** Performance on generating all-in-focus images from a single slice in the focus stack. The best results are in bold numbers.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
MDP [153]	23.50	0.674	0.394
IFAN [149]	23.48	0.679	0.371
DRBNet [10]	24.27	0.681	0.377
Ours	**24.79**	**0.704**	**0.351**



Input **W** Input **UW** GT DRBNet [10] Ours

Figure 4.4: **Defocus deblurring.** We showcase the results of our method against SoTA single image defcous deblurring DRBNet [10]. Note that our method restores severely blurred regions in the background that single-image based methods often struggle with.

4.5 Experimental Results

We train our method to perform defocus control through training on the *proxy task* of image refocus. As a result, our model can perform a variety of related defocus control tasks. Specifically, we evaluate our method on defocus deblurring, synthesizing shallow DoF, and image refocus.

Evaluation metrics. We use the standard signal processing metrics, i.e., the peak signal-to-noise ratio (PSNR) and the structural similarity index measure (SSIM). We also report the learned perceptual image patch similarity (LPIPS) [201].

4.5.1 Defocus Deblurring

Task. The goal of defocus deblurring is to remove the defocus blur in the image. For our method to perform defocus deblurring, we simply set the target defocus map to all zeros. To obtain an all-in-focus image as a ground truth, we perform focus stacking using our focus stacks through commercial software provided by HeliconFocus. Then the evaluation task is deblurring individual slices from the focus stack to generate an all-in-focus image. Due to the focus magnification between the focus stack slices, we align the field-of-view (FoV) with the all-in-focus image through a combination of FoV matching and brute-force search for the best scaling and translation parameters that minimize the error. We use the same alignment method when evaluating all the methods to ensure fairness.

Methods. We compare our method with the following single-image defocus deblurring methods: Dynamic Residual Blocks Network (DRBNet) [10], Multi-task DP (MDP) network [153], and Iterative Filter Adaptive Network (IFAN) [149]. Note that these methods do not take the ultra-wide image as input, and the main purpose of the comparison is to highlight the value of leveraging an available dual-camera setup. Our dataset does not contain DP data and thus we are not able to benchmark the DP defocus deblurring methods [154–158]. As for the evaluation on other defocus deblurring datasets (e.g., [158]), our method requires dual-camera input not available in current datasets.

Evaluation. In Table 4.1, we compare the performance of our method against other defocus deblurring methods. Our method achieves the best results on all metrics with dual camera inputs. Note that our method has never seen all-in-focus outputs / zero target defocus maps during training and learns to deblur via the proxy task. Figure 4.4 shows two deblurring results of our method against DRBNet [10]. As shown in the zoomed-in insets, our method is able to restore severely blurred regions better compared to DRBNet. In general, single-image defocus deblurring methods suffer from arti-



Input **W** Input **UW** GT BokehMe [11] Ours

Figure 4.5: **Blurring results.** Our method can synthesize shallow DoF from an all-in-focus image with a performance competitive with SoTA in bokeh rendering [11].

Table 4.2: **Bokeh blurring evaluation.** performance on simulating different slices of the focus stack from the all-in-focus image.

Method	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓
BokehMe [11]	26.65	0.870	0.241
Neural Rend. [11]	27.87	0.874	0.246
Classic Rend. [11]	26.66	0.870	0.241
Ours	**29.78**	**0.898**	**0.172**

facts and tend to hallucinate when restoring severely blurred regions. Therefore, an additional signal such as the **UW** is very useful when the details are completely lost in the input image. While the main task of our proposed method is not only defocus deblurring, it achieves the SoTA deblurring results quantitatively and qualitatively. These results also demonstrate how generic and flexible our proposed defocus control framework is.

4.5.2 Shallow DoF Rendering

Task. We also evaluate our method on rendering shallow DoF images. The input to the method is an all-in-focus image, an approximate target defocus map, and the desired output is the image with a synthetic shallow DoF guided by the defocus map. We use the all-in-focus image generated from



Input **W** | Input **UW** | GT | Baseline [10, 11] | Ours

Figure 4.6: **Refocus results.** We shift the focus plane and demonstrate that we can match the desired refocused image and the target blur without completely deblurring the input. Our method outperforms the baseline that refocuses by deblurring and reblurring the image.

Table 4.3: **Image refocus evaluation.** Performance on re-synthesizing focus planes given an input with different focus plane from the same scene.

Method	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓
UW + Blur [11]	21.89	0.803	0.364
Deblur [10]+Reblur [11]	26.40	0.833	0.312
Ours	**28.58**	**0.860**	**0.217**

the focus stack as input and try to reconstruct the various slices in the focus stack using each slice's defocus map as a target.

Methods. We compare against BokehMe [11], a recent state-of-the-art in shallow DoF synthesis that relies on blending the outputs of classic blur synthesis with neural rendering methods. We also evaluate the classical scattering-based blur and the neural renderer within BokehMe in isolation.

Evaluation. In Table 4.2, we show that our method is competitive with SoTA shallow DoF rendering methods. Note that for DoF reduction, **UW** does not provide a useful signal since the task primarily involves signal removal from **W**, but the model learns to perform this task as a byproduct of training on image refocus. In Figure 4.5 we show visual results where our model synthesizes realistic blur.

Table 4.4: **Ablations on Image Input**. Comparison on different input types. Although performance increases by removing the occlusion mask, qualitative performance drops (see Figure 4.7).

Method	**PSNR** ↑	**SSIM** ↑	**LPIPS** ↓
W only	28.44	0.855	0.260
UW only	22.66	0.822	0.307
No occlusion	**28.81**	**0.864**	0.219
Full input	28.58	0.860	**0.217**

4.5.3 Image Refocus

Task. Image refocus involves shifting the focus plane and as a result, the near and far focus depths. To evaluate on image refocus, we randomly sample two frames from a focus stack, a reference frame, and a target frame, and evaluate the model performance in reproducing the target frame.

Methods. There is limited work on single-image refocus, the most notable work being RefocusGAN [168]. The idea behind RefocusGAN is to use generative models to deblur the image followed by blurring it. This approach is likely because of the difficulty of realistically switching between different defocus amounts directly [174]. However, we are not able to compare with RefocusGAN as the code and trained models are not available. As an alternative for comparison, we adopt SoTA in defocus deblurring (DRBNet [10]) and SoTA in blurring (BokehMe [11]) for image refocus. We also compare against blurring the aligned **UW** directly since it could approximate an all-in-focus image.

Evaluation. In Table 4.3 we show that our method outperforms the baseline in image refocus. Note that since we train our method to switch between the reference defocus to the target defocus, the model can implicitly learn to switch between different PSF scales from the data. We show visual results in Figure 4.6. Note that when the target image contains blurry regions like shown on the wall, our method deblurs the input just enough to match the target defocus.

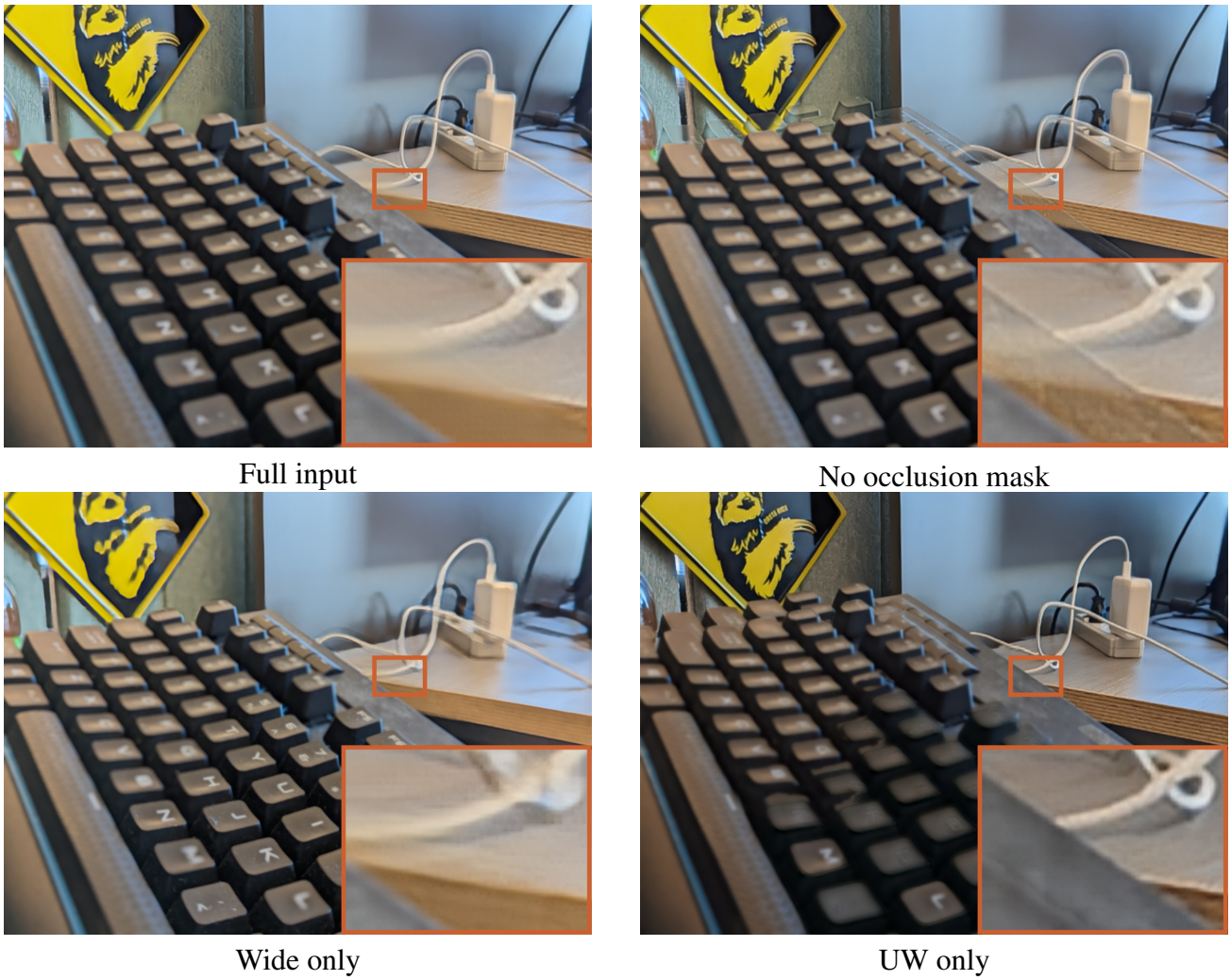


Figure 4.7: **Ablation.** Using the occlusion mask helps the model avoid transferring warping artifacts to the final image, while using **W** only hinders deblurring performance, and **UW** only suffers from warping artifacts and lower resolution.

4.5.4 Ablation Study

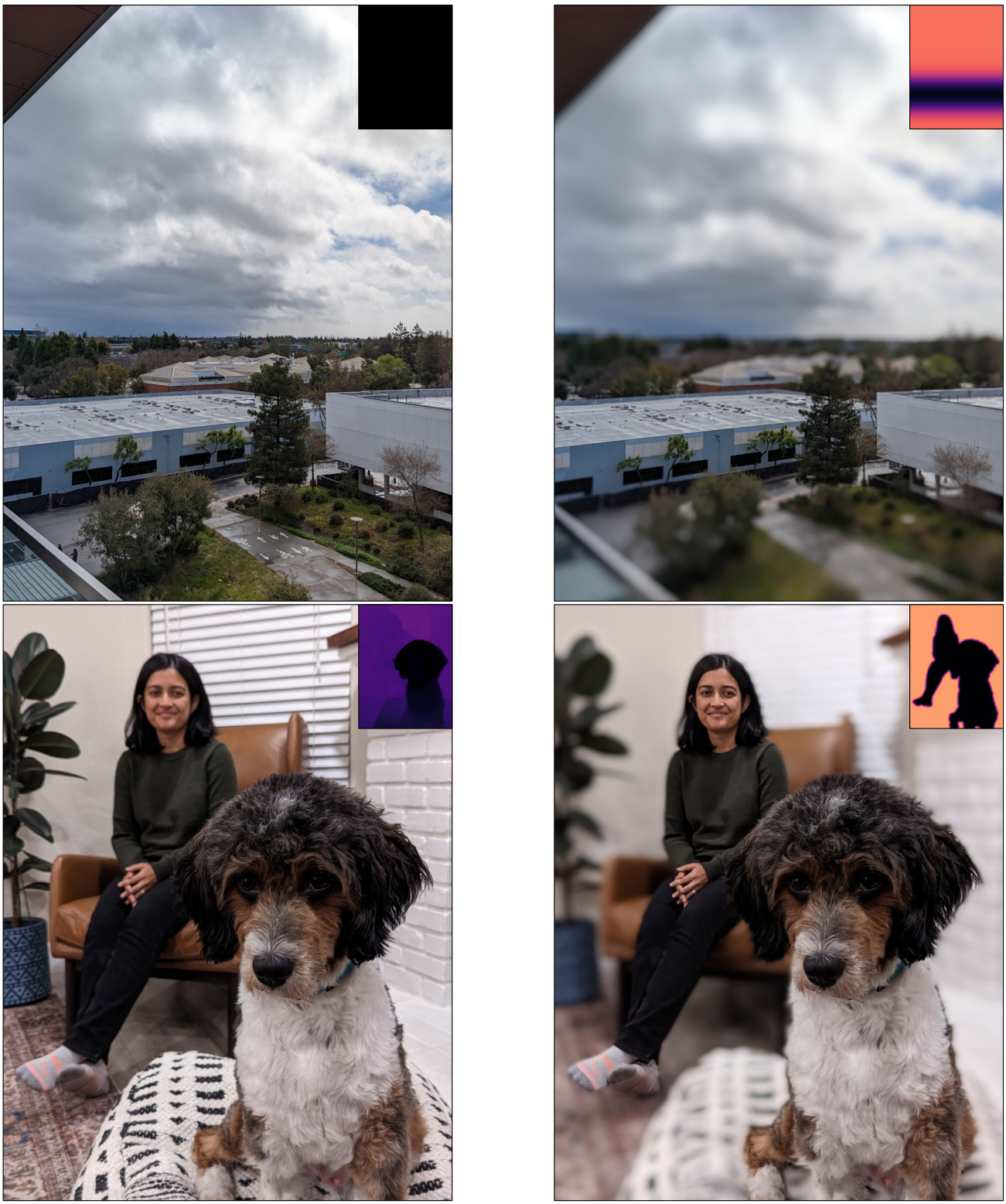
The key idea of our work is using the ultra-wide camera as a guide to performing DoF control. To evaluate the effects of using **UW**, we train a model using only W (only keeping the Wide refinement module) and similarly training a **UW** only model. We compare their performance on image refocus in Table 4.4. Note that while the wide input is sufficient when the target involves only blurring or minimal deblurring, it is an ill-posed setup when it requires considerable deblurring. On the other

hand, the warped **UW** lower quality severely limits the performance when relying on it completely. We visualize an example in In Figure 4.7. Note that when using **W** only, deblurring performance is limited. Also we note that when removing the occlusion mask, while signal-processing metrics could see slight improvements, qualitative performance drops as we can observe ghosting artifacts around occluded boundaries.

Applications. Our method allows for arbitrary target defocus maps as an input. In Figure 4.8 we demonstrate a *tilt-shift* effect, where a large scene appears smaller because of the blur, as well as using a segmentation mask to deblur objects of interest (the person) while blurring the remaining objects.

4.6 Limitations and Conclusion

We present DC^2, a novel framework for defocus control with dual-camera consumer smartphones. We bypass the need for synthetic data and domain gap issues by training with real data captured with a smartphone device. We do so by re-framing the defocus control problem as refocus and designing a learning-based solution for the same. The key idea behind our method is to use **UW** input as an additional signal to control defocus of **W**. Naturally, a limitation then is the DoF of **UW** itself; as objects outside of its DoF might not be sharper than in **W**. In general, our method benefits from asymmetry in the **W** and **UW** camera configurations and likely won't perform as well in systems with identical cameras. Another limitation is our dependence on pre-existing optical flow and stereo depth algorithms which can suffer from severe artifacts with defocus blur (Figure 4.9). A promising avenues for future work includes utilizing additional cameras to jointly model both scene depth and defocus control.



Input Output

Figure 4.8: **Creative applications**. (*Top*) we apply tilt shift effect that makes large objects appear as miniatures. (*Bottom*) we use segmentation mask to deblur objects of interest and blur the background.



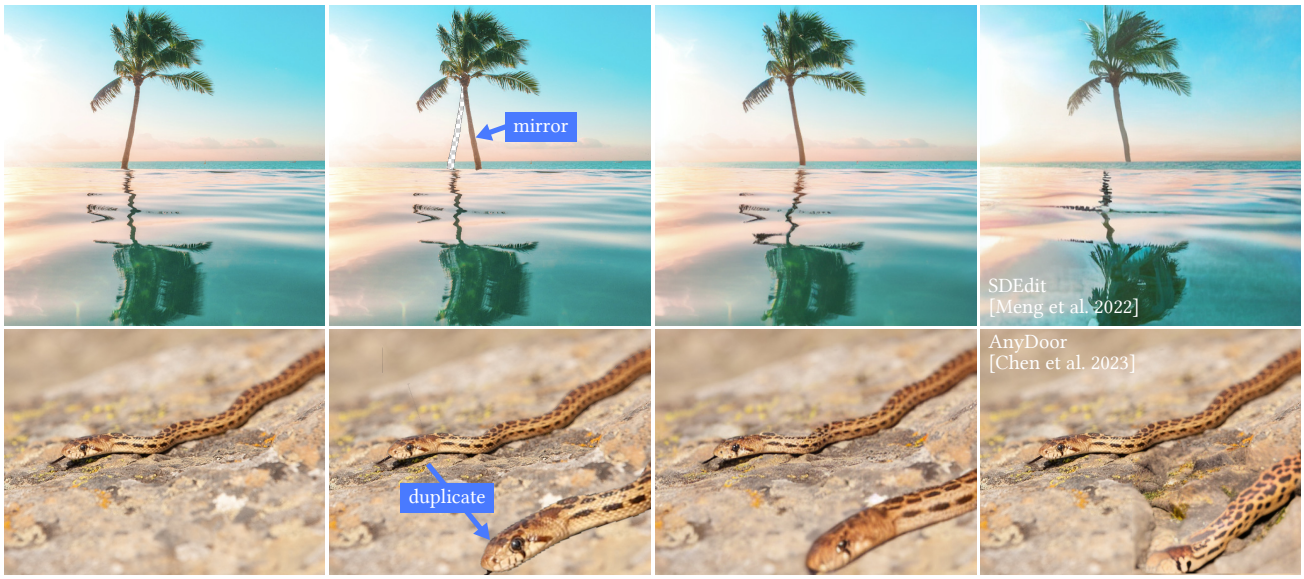
W Warped **UW**

Figure 4.9: **Failure cases.** (*Top*) In regions where **UW** is out of focus, as in the table shown above, our method would not be able to restore the sharp texture shown in the **W** image when focused on that object. (*Bottom*) Aligning the **UW** using optical flow based alignment could suffer when **W** is severly blurred like shown on the warping artifacts on the window.

Part II

Generative Priors

Chapter 5: Learning Photo Editing from Dynamic Videos



(a) reference I (b) coarse edit I_{coarse} (c) ours (d) baseline

Figure 5.1: **Applications of Magic Fixup.** We propose a diffusion model for image editing. Starting from an input image (a), a user specifies their desired changes by rearranging automatically segmented scene objects using simple 2D transforms to produce a coarse edit (b). Our model transforms this coarse edit into a realistic image (c), correctly accounting for secondary effects critical for realism, such as reflections on the water (top) or changes in depth-of-field (bottom), producing much more plausible edits than state-of-the-art methods (d). Photos sourced from ©Unsplash.

5.1 Introduction

Image editing is a labor-intensive process. Although humans can quickly and easily rearrange parts of an image to compose a new one, simple edits can easily look unrealistic, e.g., when the scene lighting and physical interactions between objects become inconsistent. Fixing these issues

manually to make the edit plausible requires professional skills and careful modifications, sometimes down to the pixel level. The success of recent generative models [202–205] paves the way for a new generation of automated tools that increase the realism of image edits while requiring much sparser user inputs [206–209]. Generative methods providing explicit spatial keypoints control have been proposed but are either limited to certain domains [210] or modest changes [18]. State-of-the-art approaches, however, regenerate pixels based on a user-specified text prompt and a mask of the region to influence [211, 212, 12, 13]. This interface is not always natural. In particular, it does not allow spatial transformations of the existing scene content, as we show in Figure 5.2, and object identities are often not fully preserved by the re-synthesis step [17, 213].

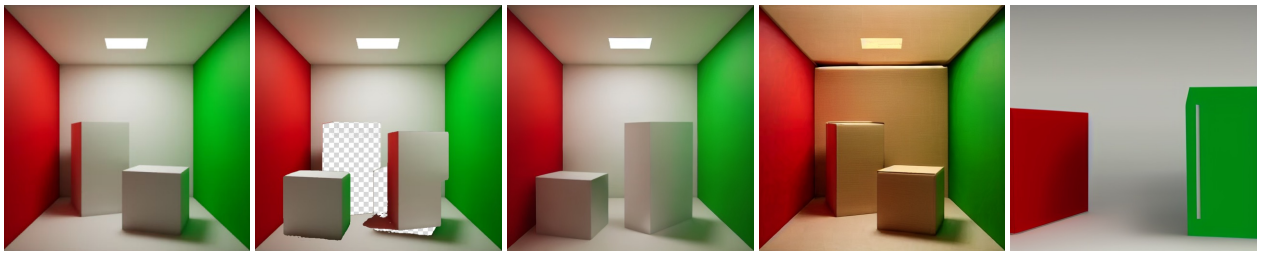


Figure 5.2: **Comparison with text based control.** Our method directly takes a coarse user edit and makes it photorealistic. Our editing is both easy and precise, and our model can harmonize the global illumination appropriately. Text-based editing methods [12, 13] on the other hand, are not able to perform such edits, resulting in global appearance changes [12] or unrealistic image [13].

In this paper, we propose a new approach to image editing that offers the controls of conventional editing methods and the realism of the modern generative model (Figure 5.1). Our method uses human inputs where it shines: users can segment the image and rearrange its parts manually in a "cut-and-transform" approach, e.g., using simple 2D transforms, duplication, or deletion to construct their desired layout, just like a collage [209]. We call our collage-like editing interface the *Collage Transform*. We then train a diffusion model to take care of the hard work of making the edit photoreal-

istic. Our model "projects" the coarsely edited image onto the natural image manifold, fixing up all the low-level image cues that violate its image prior, such as tweaking poses, blending object boundaries, harmonizing colors, adding cast shadows, reflections and other second-order interactions between the object and the environment.

Crucially, we explicitly fine-tune a latent diffusion model [202] so its output deviates as little as possible from the user's specifications and the appearance of the original objects in the scene. This is essential for photographers, as they spend significant effort capturing their images and would like to retain the content identity as much as possible. When editing an image, there is a subtle balance between being faithful to the original image and harmonizing the edited image to preserve realism. This is the regime that our work focuses on. Our insight is that videos provide a rich signal of how an edited photo's appearance should change to preserve photorealism. From videos, we can learn how objects' appearances change in the real world as they deform and move under changing light. Camera motion and disocclusions give us priors about what hides behind other objects and how the same object looks under changing perspectives.

To exploit these cues, we build a paired image dataset from a large-scale video corpus. Each pair corresponds to two frames sampled from the same video: source and target frames. We then automatically segment [14], and transform objects in the source frame to match the pose of the corresponding objects in the target frame, using two motion models based on optical flow, designed to simulate the coarse edits a user would make using our Collage Transform interface. Since the images are now roughly aligned, we can train our model to convert the coarsely edited image into the ground truth target frame in an image-to-image [214, 215] fashion. This alignment procedure encourages the model to follow the user-specified layout at test time closely. Additionally, our model is carefully designed to transfer fine details from the reference source frame to preserve the identity and appearance of objects

in the scene.

Our approach can produce plausible and realistic results from real user edits, and effectively projects coarse user edits into photorealistic images, confirming our insights on the advantages of using video data and a carefully designed motion model. Compared to the state-of-the-art, we show our outputs are preferred 89% of the time in a user study.

In short, our contributions are as follows:

- the Collage Transform, a natural interface for image editing that allows users to select and alter any part of an input image using simple transforms and that automatically turns the resulting edit into a realistic image,
- a new paired data generation approach to supervise the conversion from coarse edits to real images, which extracts pairs of video frames and aligns the input with the ground truth frame using simple motion models,
- a conditioning procedure that uses: 1. the warped image to guide layout in the diffusion generator, and 2. features from a second diffusion model to transfer fine image details and preserve object identity.
- a comprehensive analysis on the model's generalization to diverse editing tasks, like spatial editing, colorization, 3D transformation, NS perspective warping.

5.2 Related Work

Classical image editing. Classical image editing techniques offer various types of user controls to achieve diverse objectives. For instance, image retargeting aims to alter an image's size while pre-

serving its key features and content [216–219]. In contrast, image reshuffling rearranges an image's content based on user-provided rough layouts and imprecise mattes [219–221]. Image harmonization integrates objects from different images, adjusting their low-level statistics for a seamless blend [222, 223]. A common thread in these classical image editing applications is the crucial role of user interaction, which provides the necessary control for users to realize their vision. Our method aligns with this approach, allowing users to reconfigure a photograph based on their preliminary edits.

Controllable image generation. The rapid advancement in photorealistic image generation has inspired researchers to adapt generative models for image editing tasks. Early efforts focused on high-level edits, like altering age or style, by manipulating latent space of Generative Adversarial Networks (GANs) [224–226]. In a vein similar to our work, Generative Visual Manipulation [227] involves projecting user-edited images onto the natural image manifold as approximated by a pre-trained GAN. The recent introduction of CLIP embeddings [228] has further propelled image editing capabilities, particularly through text prompts [229–231, 208, 12, 232, 233]. DragGAN [210] introduces fine control in image editing by using key-handles to dictate object movement, and follow-up works extend the drag-control idea to diffusion models [18, 234, 235]. Image Sculpting [236] takes a different approach by directly reposing the reconstructed 3D model of an object and re-rendering it, providing high level of control, but time consuming editing process unlike our Collage Transform interface that is designed to increase editing efficiency. CollageDiffusion [209] guides text-to-image generation by using a collage as additional input. However, while CollageDiffusion focuses on controlling the generation of an image from scratch, we focus on using collage-like transformation to edit a reference image, and focus on preserving its identity.

Reference-based editing with generative models. To extend controllable image generation into editing

real (non-generated images), one can invert the image back to noise [237], and then guide the iterative denoising process to control the image generation[238, 19, 13]. However, naively guiding the model without any grounding can lead to a loss in image identity. Prior work [239, 240, 17] preserves the image identity through a pretrained feature extractor like CLIP [228] or DINO [241], using a ControlNet like feature-injection [242, 17], a dual-network approach [13, 243], or a combination of those approaches [17, 244]. We adopt the dual-network approach, as it allows us to fully fine-tune the model and taylor it to our photorealistic editing task using our video-based dataset. AnyDoor [17] similarly uses video frames during training, but their focus is to recompose individual objects into the scene. On the other hand, we use video data to recompose the *entire scene* and use motion models designed for a convenient photo editing interface. Closest to our work is MotionGuidance [245] that uses optical flow to guide editing the reference frame with diffusion guidance [238] for a highly user-controllable edit. However, dense optical flow is difficult to manually provide for a user, unlike simple cut-and-transform edits in our Collage Transform. Furthermore, they rely on a prohibitively time-consuming guidance that take as long as 70 minutes for a single sample. On the other hand, our approach takes less than 5 seconds to fix up the user edit, allowing for interactive editing process.

5.3 Method

We aim to enable an image editing workflow in which users can select objects in a photograph, duplicate, delete or rearrange them using simple 2D transforms to produce a realistic new image (§ 5.3.1). We leverage image priors from pretrained diffusion models to project the coarsely edited image onto the natural image manifold, so the user can focus on specifying high-level changes without worrying about making their edits plausible (§ 5.3.2). Existing diffusion models can produce impres-

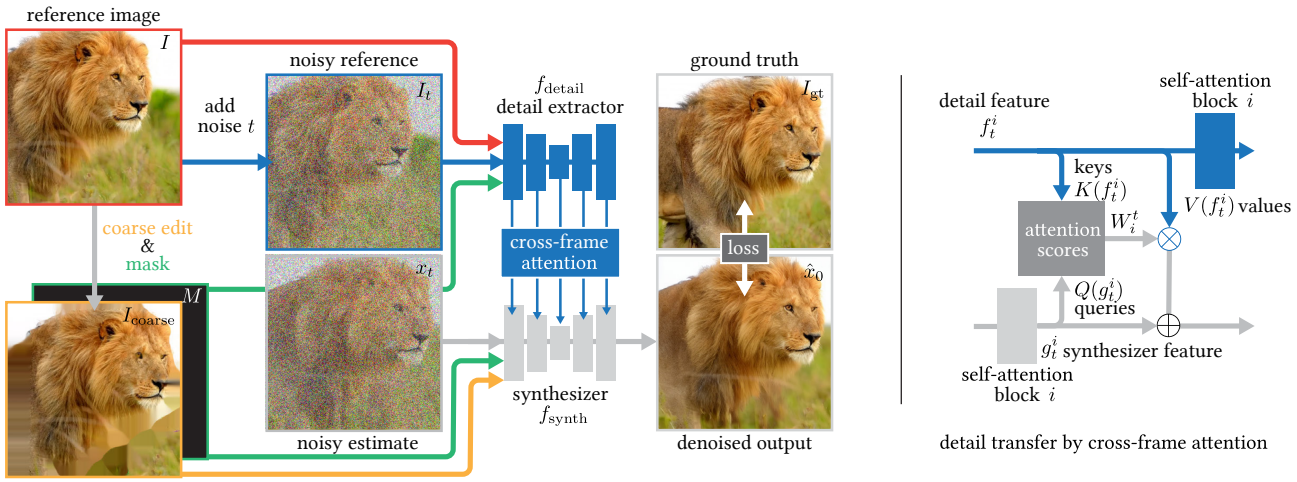


Figure 5.3: **Overview.** Our pipeline (left panel) uses two parallel models, a detail extractor (top) and a synthesizer (bottom), to generate a realistic image from a coarse user edit and a mask recording missing regions caused by the edit. The detail extractor processes the reference image, a noisy version of the reference and the mask, to produce a set of features that guide the synthesis and allow us to preserve the object appearance and fine details from the reference image. The synthesizer generates the output conditioned on the mask and coarse edit. The features from the detail extractor are injected via cross-attention at multiple stages in the synthesizer, in order to transfer details from the input. Both models are finetuned on our paired dataset. The right panel shows a detailed view of our cross-attention detail transfer operator. Photos sourced from ©Adobe Stock.

sive results but often do so at the expense of control and adherence to the user input [19]. In particular, they tend to "forget" the identity and appearance of the edited object [239], and often only loosely conform to the user-specified pose [17]. Our method addresses these issues using two mechanisms. First, our synthesis pipeline is a conditional diffusion model (§ 5.3.4) that follows the coarse layout defined by the user, and transfers fine details from the reference input image (§ 5.3.3) to best preserve the original image content. Second, we construct a supervised dataset exploiting object motion from videos to finetune the pretrained model to explicitly encourage content preservation and faithfulness to the input edit (§ 5.3.5).

5.3.1 Specifying coarse structure with simple transforms

Starting from an image $I \in \mathbb{R}^{3hw}$, $h = w = 512$, we run an automatic segmentation algorithm [14] to split the image into non-overlapping semantic object segments. The user can edit this image by applying 2D transformations to the individual segments (e.g., translation, scaling, rotation, mirroring). Segments can also be duplicated or deleted. Figure 5.1 illustrates this workflow. We keep track of holes caused by disocclusions when moving the segment in a binary mask $M \in \{0, 1\}^{hw}$, and inpaint them using a simple algorithm [246]. We denote the resulting, coarsely edited image by $I_{\text{coarse}} \in \mathbb{R}^{3hw}$.

We operate in an intermediate latent space for efficiency, but our approach also applies to pixel-space diffusion. With a slight abuse of notation, in the rest of the paper $I, I_{\text{coarse}} \in \mathbb{R}^{3hw}$, with $h = w = 64$ refer to the input and coarse edit after encoding with the latent encoder from Stable Diffusion [202], and M the mask downsampled to the corresponding size using nearest neighbor interpolation. The latent triplet $(I, I_{\text{coarse}}, M)$ forms the input to our algorithm.

5.3.2 From coarse edits to realistic images using diffusion

We want to generate a realistic image that (1) follows the large-scale structure defined by the coarse user edit, and (2) preserves the fine details and low-level object appearance from the unedited image, filling in the missing regions. Our pipeline, illustrated in Figure 5.3, uses 2 parallel models.

The first, which we call *synthesizer* f_{synth}, generates our final output image. The second model, which we name *detail extractor* f_{detail}, transfers fine-grained details from the unedited reference image I to our synthesized output during the diffusion process. It modulates the synthesizer by cross-attention at each diffusion step, an approach similar to Masa-Ctrl [13] and AnimateAnyone [243]. Both models

are initialized from a pretrained Stable Diffusion v1.4 model [202], and finetuned on our paired dataset (§ 5.3.5). Since we have a detailed reference image I to guide the synthesis, we do not need the coarse semantic guidance provided by CLIP, so we remove the CLIP cross-attention from the model.

Let $T \in \mathbb{N}^*$ be the number of sampling steps, and $\alpha_0, \ldots, \alpha_T \in \mathbb{R}^+$ be the alphas of the diffusion noise schedule [204]. Starting from an image $x_0 \in \mathbb{R}^{3hw}$, the forward diffusion process progressively adds Gaussian noise, yielding a sequence of increasingly noisy iterates:

$$x_t \sim \mathcal{N}\left(\sqrt{\alpha_t} x_{t-1}; (1-\alpha_t)\mathbf{I}\right). \tag{5.1}$$

The base diffusion model f is trained to reverse this diffusion process and synthesize an image iteratively, starting from pure noise $x_T \sim \mathcal{N}(0, I)$. The synthesizer and detail extractor in our approach make a few modifications to this base model, which we describe next.

5.3.3 Extracting details from the reference image

During inference, at each time step t, we start by extracting a set of features F_t from the reference image using f_{detail} (Figure 5.3, top). These features will guide the synthesis model and help preserve realistic image details and object identity. Since we use a pretrained diffusion model as a feature extractor, we start by adding noise to the reference unedited image:

$$I_t = \sqrt{\bar{\alpha}_t} I + (1-\bar{\alpha}_t)\epsilon, \tag{5.2}$$

with $\epsilon \sim \mathcal{N}(0, \mathbf{I})$, $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$. We extract the feature tensors immediately before each of the $n = 11$ self-attention blocks in the model:

$$F_t := [f_t^1, \ldots, f_t^n] = f_{\text{detail}}([I_t, I, M]; t), \tag{5.3}$$

where $[\cdot]$ denotes concatenation along the channel dimension. Our feature extractor also takes as input the clean reference image since it is always available for detail transfer and mask, so the model knows which regions need inpainting. Since the pretrained model only takes I as an input, we modify the first layer at initialization by padding its weight with zeros to accept the additional channel inputs. Using a noisy version of the reference ensures the extracted features are comparable to those in the cross-attention operators of the synthesis model.

5.3.4 Image synthesis by detail transfer to the coarse edit

The synthesizer f_{synth} generates the final image, conditioned on the detail features F_t. Unlike standard diffusion sampling, we do not start from pure Gaussian noise. Instead, inspired by SDEDit [19], we start from an extremely noisy version of the coarsely edited image:

$$x_{T-1} = \sqrt{\bar{\alpha}_{T-1}} I_{\text{coarse}} + (1 - \bar{\alpha}_{T-1})\epsilon, \tag{5.4}$$

so we effectively bypass the first denoising step with adding noise to the edit rather than pure noise. This initialization circumvents a commonly observed issue where diffusion models struggle to generate images whose mean and variance deviate from the normal distribution. This is particularly important in our setup as the user input can have arbitrary color distribution, and we need the model to match the

user input. This has been shown to stem from a domain gap between training and sampling [15, 16]:

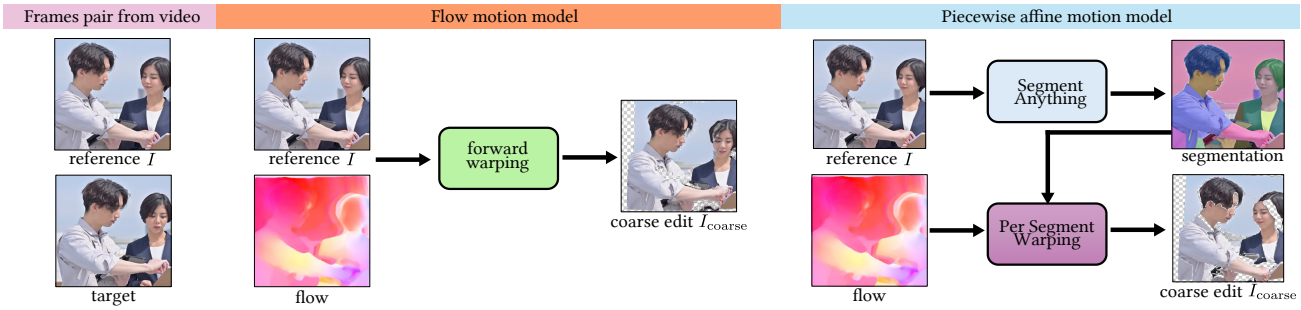


Figure 5.4: **Dataset synthesis.** Our dataset synthesis pipeline starts by sampling two frames from a video. We set one frame to be the reference, which we warp by one of our motion models, and set the other frame to be the target that we warp the reference towards. To generate aligned training pairs, we use 2 motion model to warp the reference frame towards the ground truth (target frame). The first model uses optical flow (left). It provides the most accurate alignment but does not correspond to what the user would provide during inference. This motion model encourages adherence of our model's output to the layout specified using the coarse edit. To generate training pairs closer to the collage-like user inputs, we use a second motion model (right). For this, we segment everything in the image [14] and apply similarity transforms to each segment, estimated from the flow within the segment. Figure 5.10 analyses the impact of these motion models on the final result. Photos sourced from ©AdobeStock.



Figure 5.5: **Effects of Latent Initialization.** Starting from pure noise, as is standard practice, the model struggles to generate images with deep blacks and synthesizes nonsensical content to keep the image's mean and standard deviation close to the starting Gaussian noise. This is a known issue with current diffusion models [15, 16]. Instead, during inference, we initialize the latent to the warped image with a very large amount of Gaussian noise before running the diffusion. This simple change makes a drastic difference and lets the model preserve the image content. Photo sourced from ©Unsplash.

the model never sees pure noise during training, but a sample from the normal distribution is the starting point for inference. Our latent initialization addresses this issue by directly bridging the gap between training and inference. In Fig. 5.5 we show how initializing with pure noise leads to a low contrast image, while our initialization allows the model to preserve the input color range well. For subsequent steps during inference, we update the current image estimate x_t at each time step t, using the following update rule:

$$x_{t-1} = f_{\text{synth}}([x_t, I_{\text{coarse}}, M]; t, F_t). \tag{5.5}$$

We provide the mask and coarse edit as conditions by simple concatenation, but because we need to extract fine details from the reference, we found passing the reference information by cross-attention with the features F_t provided richer information. Again, we extend the weight tensor of the first convolution layer with zeros to accommodate the additional input channels.

Detail transfer via cross-attention We use the intermediate features $F_t = [f_t^1, \ldots, f_t^n]$, extracted *before* the detail extractor's self-attention layers to transfer fine image details from the reference image to our synthesis network by cross-attention with features $[g_t^1, \ldots, g_t^n]$ extracted *after* the corresponding self-attention layers in the synthesis model. See the right panel of Fig. 5.3 for an illustration, where Q, K, V are linear projection layers to compute the query, key, and value vectors, respectively, and W_i^t is the matrix of attention scores for layer i, at time step t. The feature tensors g_t^i, f_t^i are 2D matrices whose dimensions are the number of tokens and feature channels, which depend on the layer index i.

5.3.5 Training with paired supervision from video data

We train our model on a new dataset obtained by extracting image pairs from videos to reconstruct a ground truth frame given an input frame and a coarse edit automatically generated from it. Our insight is that motion provides useful information for the model to learn how objects change and deform. Videos let us observe the same object interact with diverse backgrounds, lights, and surfaces. For example, skin wrinkles as a person flexes their arm, their clothes crease in complex ways as they walk, and the grass under their feet reacts to each step. Even camera motion yields disocclusion cues and multiple observations of the same scene from various angles.

Concretely, each training sample is a tuple $(I, I_{\text{gt}}, I_{\text{coarse}}, M)$, where I and I_{gt} are the input and ground-truth frames, respectively, extracted from the video with a time interval sampled uniformly at random from $\{1, \ldots, 10\}$ seconds between them. However, if the computed flow between the two frames was too large (at least 10 percent of the image has a flow magnitude of 350 pixels), we resample another pair. This is to ensure that the warping produces reasonable outputs. We construct the coarse edit I_{coarse} and corresponding mask M using an automated procedure that warps I to approximately match I_{gt}, in a way that mimics our Collage Transform interface. For this, we use one of 2 possible editing models: a flow-based model and a piecewise affine motion model (Fig 5.4).

Flow-based editing model We compute the optical flow using RAFT-Large [247] for each consecutive pair of frames between I and I_{gt} and compose the flow vectors by backward warping the flow to obtain the flow between the two frames. We then forward warp I using softmax-splatting [248], to obtain I_{coarse}, which roughly aligns with the ground truth frame. The forward warping process creates holes in the image. We record these holes in the mask M. Our model needs to learn to inpaint these re-

gions and those we have no correspondence (e.g., an object appearing in the frame). Using flow-based warping helps the model learn to preserve the identity of the input, rather than always hallucinating new poses and content.

Piecewise affine editing model Optical flow warping can sometimes match the ground truth too closely. As we discuss in Section 5.4 and Figure 5.10, training the flow-based editing model only can limit the diversity of our outputs, leading to images that do not deviate much from the coarse edit. Flow-warping is also reasonably distinct from our expected test-time user inputs (§ 5.3.1). Our second editing model addresses these issues by transforming the reference frame as a *collage*. We compute a depth map for the image using MiDaS [249, 250] and automatically segment the image using SegmentAnything [14].

We then transform each segment using the affine transformation that best matches the optical flow for this segment, compositing them back to front according to each segment's average depth. For the image regions that are not segmented, we use the optical flow warping scheme described above. Due to the coarser alignment, the model learns how different parts of the scene interact in a realistic setting, like associating objects and their shadows and reflections.

We use a dataset consisting of 12 million 5-10 second clips of stock videos, and we filter out keywords that indicate static scenes or synthetic/animated videos, as we are only interested in photo-realistic videos and also highly dynamic scenes where the motion is too large (like car racing). For each valid clip, we sample one pair and compute the warping using both motion models. After filtering for desired motion, we use 2.5 million clips, creating a dataset consists of 2.5 million samples for each motion model, making a total of 5 million training pairs.

5.3.6 Implementation details

We finetune both models jointly for 120,000 steps with a batch size of 32, using Adam [251], with a learning rate of 1×10^{-5} on 8 NVIDIA A100 GPUs, which takes approximately 48 hours. Note that this is considerably more efficient than recent compositing work [239] that uses 64 NVIDIA V100 GPUs for 7 days. We hypothesize that the stronger input signal helps the model converge faster. We use a linear diffusion noise schedule, with $\alpha_1 = 0.9999$ and $\alpha_T = 0.98$, with $T = 1000$. During inference, we sample using DDIM for 50 denoising steps.

5.4 Experimental Results

We evaluate our method qualitatively on a set of user edits to demonstrate real-world use cases, as well as on a held-out validation dataset created in the same way as our training set (§ 5.3.5) for quantitative evaluation. *In the appendix, we show additional applications of the model on editing tasks beyond spatial recomposition, like colorization, perspective editing, and 3D transformation.*

Our model is trained on a synthetically-generated dataset. We validate that it generalizes to real user edits using a prototype interface illustrating our segment-based editing workflow. The user can segment any part of the image and transform, duplicate, or delete it. To the best of our knowledge, no previous work focuses exactly on our use case (photorealistic spatial edits), so we adapt closely related techniques to our problem setting for comparison. Specifically, we compare to the following baselines:

1. SDEdit [19]: a general text-based editing method that trades off the adherence to the input image and the faithfulness to the text. This is the most general method we compare against, as we can

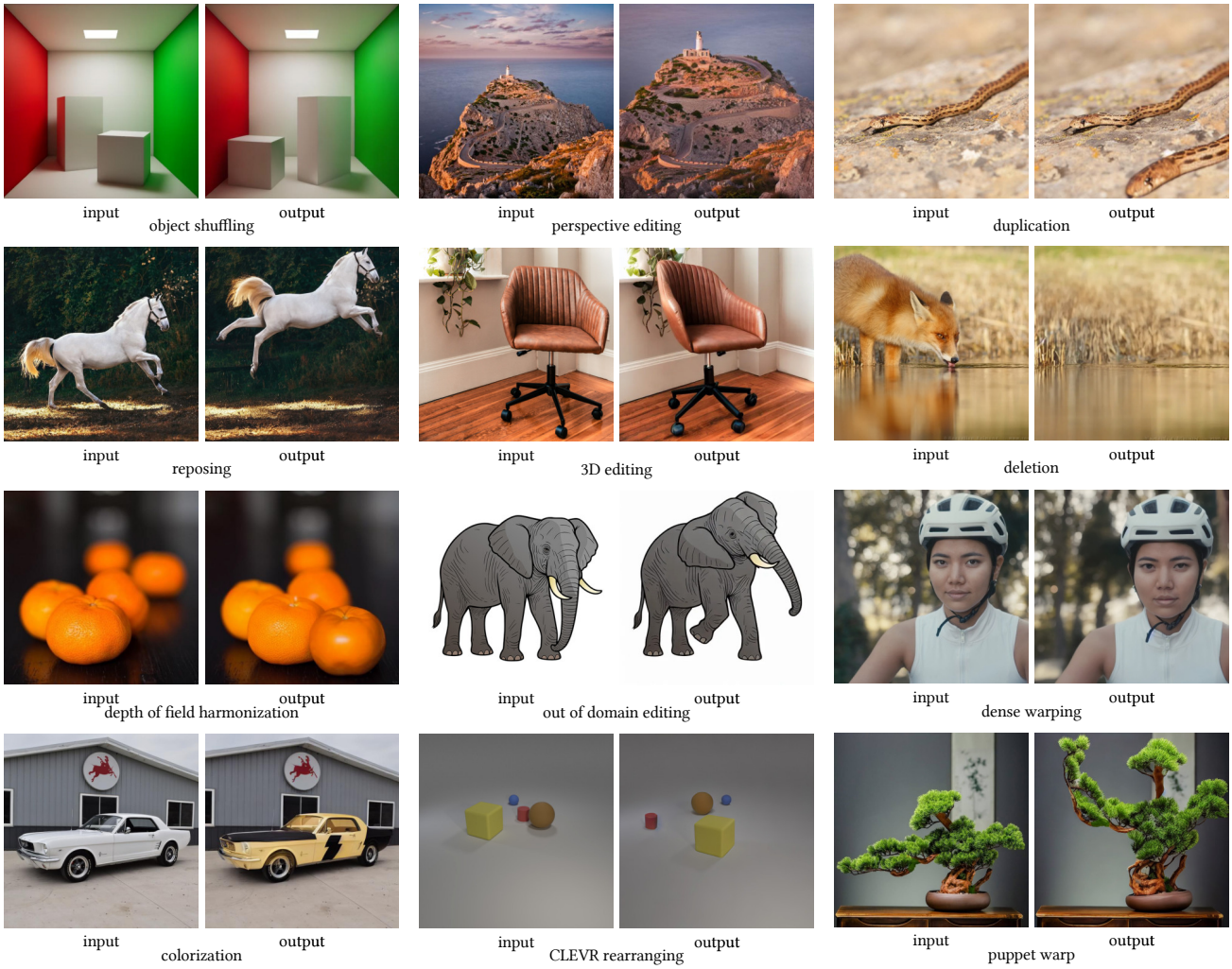


Figure 5.6: **MagicFixup applications.** Our method can generalize to a large number of different applications. We just need to create a coarse edit, and MagicFixup realistically clean up the edit. Here we show a selection of tasks we used MagicFixup to achieve, like image recomposition, perspective editing, reposing, 3D transformation, and colorization. We show the coarse edits for these examples throughout the appendix. Photos sourced from ©Unsplash and CC licensed images.

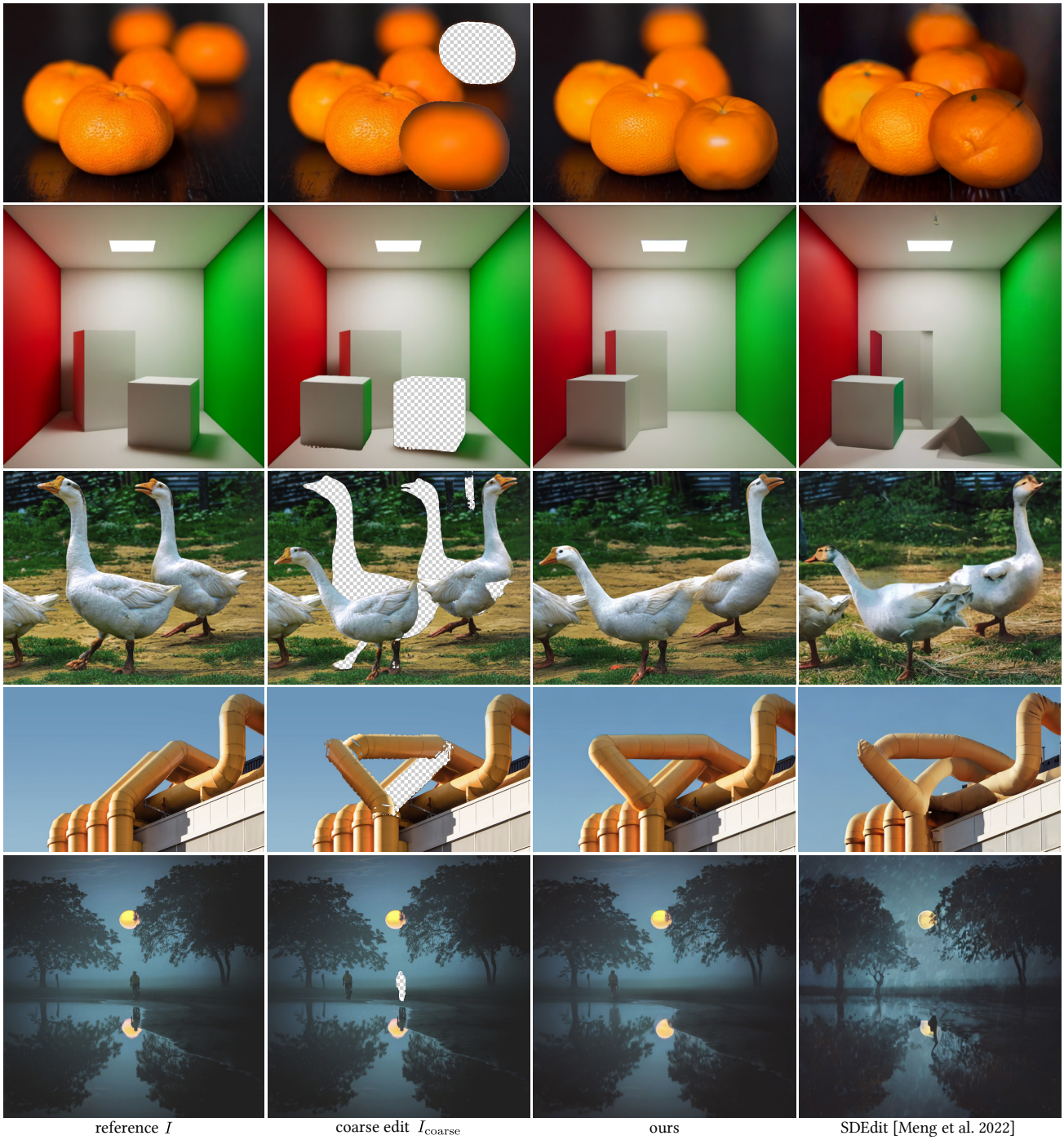


Figure 5.7: **Spatial editing.** We show example of scene recompositing. Our model is capable of synthesizing compelling effects that harmonize realistically with the rest of the image such as: changing the depth of field (row 2), adjusting the global illumination (green reflection on the cube, row 3), and removing or adding reflections (row 6). Photos sourced from ©Unsplash.



Figure 5.8: **Comparison to Anydoor [17].** Anydoor was trained to insert objects from one image to another. We can repurpose their approach for our image editing task by using the same image as source and target. Their approach does not preserve the dog's identity in this example. AnyDoor also does not harmonize the lighting properly (the sun direction and shadows are wrong), the image is too bright, and some blending seams are visible. On the other hand, our output shows natural shadows and plausible contacts with the ground, adding realistic moving sand consistent with the pose. Photo sourced from ©Unsplash.

directly provide it with the coarse user edit and a generated caption. Since SDEdit can take the coarse edit directly as an input, we emphasize it the most in our comparisons.

2. DragDiffusion [18]: a drag-based editing model that takes source-target key-handles to move parts of the object for re-posing.

3. InstructPix2Pix [12]: a text-based editing method that follows an instruction-like style captions.

4. MasaCtrl [13]: a text-based editing method that achieves a consistent identity preservation through a self-attention mechanism. However, it relies on DDIM inversion as an essential step, which can compromise the method's robustness.

Adapting the baselines. We convert our inputs to the interface expected by these baselines for comparison. SDEdit requires choosing a strength parameter dictating the amount of noise added to the input and trades off between faithfulness and unconstrained synthesis. We set the strength to 0.4 in all experiments, i.e. we start at 40% of the way through the diffusion process, adding the corresponding level of noise to I_{coarse}. Unlike ours, their model expects a text input, which we automatically compute

using BLIP [252]. We use the same generated caption with the other text-based methods (MasaCtrl and InstructPix2Pix) for the quantitative evaluation on our large corpus. For the qualitative evaluation on user edits, we choose a caption that describes the user edit to the best of our abilities. However, we note that text description of spatial edits is inherently ambiguous (which is the motivation of our proposed method).

To compare with DragDiffusion [18], we record the segment motion in our user interface, compute the motion vectors for each pixel, and use this information to automatically create the keypoint-handles input needed by DragDiffusion.



Figure 5.9: **Comparison with DragDiffusion.** We use the Drag Diffusion [18] to generate the results in the right column. We seed dragging control points this method expects for each of the modified image segments, and displace them using the same affine transform used to produce our coarse edit (second column). DragDiffusion generates fairly conservative image edits, and collapses with more drastic reposing edits. However, our method successfully handles wide range of reposing levels. Photos sourced from ©Unsplash.

5.4.1 Quantitative evaluation

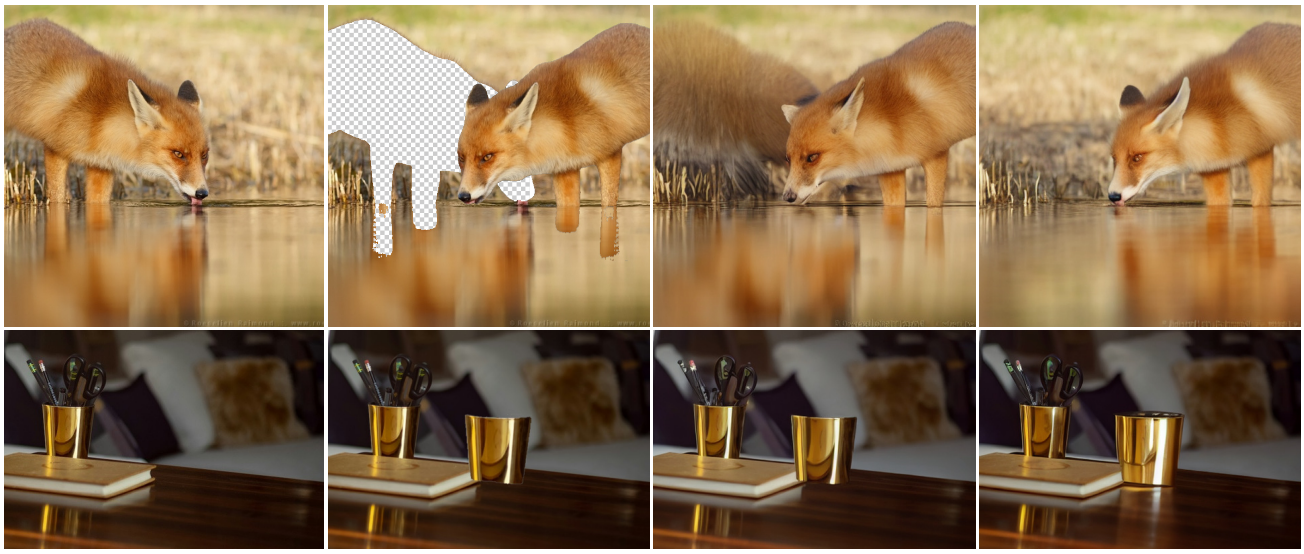
While the task of image editing is inherently subjective, a natural task is to evaluate our method on edits generated from videos using the motion models we discuss earlier. We use a held-out split of our dataset, and evaluate our method against the baselines on the performance at reconstructing

the target frame. In Table 5.1 we show that our method significantly outperforms the baselines on reconstruction metrics. We also computed the flow error, by comparing the RAFT optical flow between the reference and target, and the flow between the reference and the output. Intuitively, the smaller the flow error, the more that the method is faithful to the user edit. We show that our method is significantly outperforming all the baselines across all metrics. The second best method is DragDiffusion, which is likely due to the fact that it accepts a form of spatial edit as an input (through drag handles), and the LoRA optimization on each image to preserve identity. Across all text-based methods, SDEdit performs the best in general likely due to the fact that it can directly accept the user edit as an input rather than purely relying on the caption to perform the edit.



Figure 5.10: **Motion models ablation.** We compare how the 2 motion models we use to create our coarse edits (column 2) during training affect the model's behavior. If we warp the reference frame (column 1) using the flow only (column 3), the model learns how to harmonize the edges of the edited regions, but remains very conservative and does not add much details to increase realism. On the other extreme, if we only use the piecewise affine motion model (column 4), the model learns to hallucinate excessively, losing its ability to preserve object identity. Our full solution trains with both motion models (column 5) to increase the model versatility, allowing the model to generate realistic details while still maintaining good adherence to the user input. Photos sourced from ©Unsplash.

[1]We find that DragDiffusion fails to produce an output on edits with large motion, so we restricted this evaluation to the subset where it is able to produce an output. Note that the ordering of the methods remains the same if we restrict all methods to the same subset.



reference I | coarse edit I_{coarse} | without detail extractor f_{detail} | ours

Figure 5.11: **Architecture ablation.** Without the detail extractor branch and using CLIP to extract the reference features (3rd column), the model struggles with spatial reasoning as it cannot access the grounding of the original reference image (1st column). This ablation's outputs are overly conservative, not steering too far away from the coarse edit (2nd column). Our full model produces much more realistic edits (4th column), with harmonious shadows and object-background contact. It refines object boundaries and synthesizes plausible reflections. Photos sourced from ©Roeselien Raymond and ©Unsplash.

5.4.2 Evaluation on user edits

The training dataset we use for MagicFixup allows for a host of different applications. In Fig. 5.6 we highlight 12 different example applications we generated using MagicFixup, which includes edits outside of the training data, like perspective edits, 3D transformations, and even colorization. In this section, we discuss edits generated using our Collage Transform interface and compare against pose-editing baselines. We further highlight additional applications beyond image recomposition in the appendix.

Collage transform editing. Using our user interface, we created a collection of edits that spatially recompose photos. In Fig. 5.7 we show how our model adds realistic details to objects moved to a region of sharper focus, snaps disconnected objects together, and resynthesizes shadows and reflections

Table 5.1: **Quantitative evaluation.** Perceptual loss, SSIM, and flow error on a held-out validation subset of our video dataset. Our method significantly outperforms all the baselines across all metrics.

Method	Motion Model	LPIPS ↓	SSIM ↑	Flow (px) ↓
MasaCtrl [13]	-	0.44	0.49	23.1
IP2P [12]	-	0.47	0.42	24.9
DragDiff* [18] [1]	Piecewise affine	0.26	0.58	16.9
	Flow-based	0.26	0.59	15.8
SDEdit [19]	Piecewise affine	0.40	0.57	24.7
	Flow-based	0.37	0.61	22.4
Ours	Piecewise affine	**0.21**	**0.70**	**5.33**
	Flow-based	**0.16**	**0.78**	**3.06**

as needed. Another natural baseline for spatial recomposition is inpainting an object and reinserting it in the image. We use AnyDoor [17] as the insertion method, and compare the recomposition result. In Fig. 5.8, we used our model to delete the dog (and automatically remove the shadow), and then re-inserted the dog using AnyDoor. The dog's identity underwent significant changes, and AnyDoor does not harmonize the composite with the ground. It also does not completely remove the halo caused by the inpainting mask in the destination region. In contrast, our model synthesizes a coherent output without discontinuity artifacts. We also used AnyDoor in duplicating the snake in Fig. 5.1, and we show that AnyDoor has a loss of identity on the snake, while our method correctly introduces some defocus blur to adjust for the reference's shallow depth of field. We compare against text-only editing methods in Fig. 5.2, and show that InstructPix2Pix [12] only alters the apperance without following the spatial edit instruction prompt, and MasaCtrl [13] completely loses the input identity due to the failure of DDIM inversion. In the appendix, we also show additional comparisons against text-only methods.

Image reposing. Since we allow the user to edit the image by selecting segments of arbitrary size, the user can re-pose objects by selecting sub-parts and applying an affine transformation on them,

effectively animating the object. We compare our method to DragDiffusion [18] that uses drag handles, and Motion Guidance [245] that uses flow to guide the diffusion sampling to follow the user edit. To ensure a fair comparison, we keep track of dense pixel correspondence in our Collage Transform user interface for the user edit. Using the dense correspondence maps, we can directly generate the drag handles and flow inputs these baselines require. In Fig. 5.9, DragDiffusion moves the lion's body higher up, which loosely aligns with the user edit, but is inconsistent with the user's intent of only moving the head. This example highlights how a non-interactive point-dragging interface can be at odds with the user's desired output, because it does not provide a good preview of what the model would generate before running it. Our Collage Transform interface is more immediate, and our coarse edit aligns with the final output. On the other hand, despite having dense flow, Motion Guidance completely fails to follow the user edit as the test time optimization process is unreliable. In the second example, DragDiffusion collapses, likely because the user input is complex and goes beyond a minimal displacement of the subject that it can handle, and Motion Guidance lifts the horse up in the air but fails to keep it in one piece.

Note that both DragDiffusion and Motion Guidance require a costly test-time optimization for each input. On NVIDIA A100, DragDiffusion takes approximately 2 minutes, and Motion Guidance takes nearly an hour for a single input. In contrast, our method only requires the feed-forward sampling, taking approximately 5 seconds.

Perceptual user study. To evaluate the realism of our editing, we conducted a user study comparing the quality of our edits against three baselines: SDEdit [19], InstrictPix2Pix [12], and Masa-Ctrl [13]. We used 11 diverse photo edits, with 21 students participating and voting for all pairs of images. For each pair, we provided the users with the reference image as well as the *intended* user edit, and asked

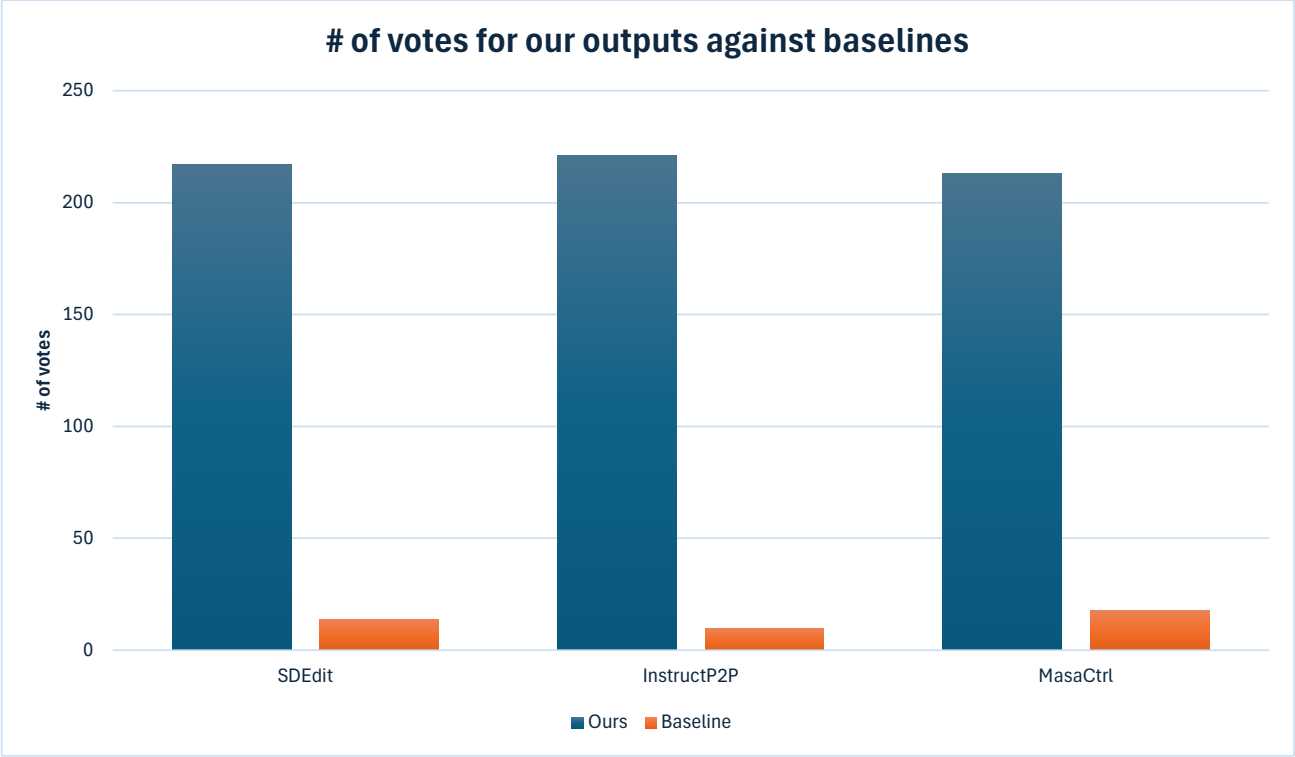


Figure 5.12: **User study results.** We asked 21 users to compare the editing results from our method and the outputs using baselines. For each baseline, the user directly compares between our method's output and the baseline's. Overall, the users overwhelmingly preferred our method. Out of 231 votes, only 14 votes were for SDEdit [19], 10 votes were for InstructPix2Pix [12], and 19 votes for MasaCtrl [13]



Figure 5.13: **Ablating reference input.** When passing an arbitrary image as a reference to the detail extractor network independent of the input of the synthesis network, we find an effect similar to style transfer. The model preserves the spatial structure of the "coarse edit" input while maintaining the style and global appearance of the "reference." This behavior likely contributes to the model robustness in generalization to new domains and processing new types of edits. The scream painting is now in public domain, and remaining photos were sourced from ©Unsplash.

for each sample the following "For the following edit, which of those images do you find a more realistic result?" in a 2-alternative forced-choice (2AFC) format. In Fig. 5.12 we show the aggregated votes for our method against each baseline. We see that our method is overwhelmingly preferred. For each baseline comparison we have 231 total votes, and only 14 votes were for SDEdit [19], 10 votes were for InstructPix2Pix [12], and 19 votes for MasaCtrl [13]. We show comparison results against those baselines in Appendix 5.6.3, and we encourage the reader to compare the results directly.

5.4.3 Ablation studies

In this section, we evaluate the role that different motion models play, as well as the importance of cross-reference attention.

Motion models ablation. Intuitively, training the model only on flow-warped images would prevent

the model from learning to synthesize drastic changes, since flow-warping tends to be well-aligned around the edges. On the other hand, using the piecewise-affine motion model requires the model to adjust the pose of each segment (and learn to connect them together nicely), which forces the model to only use the input as a coarse conditioning. In Fig. 5.10, we show that the behavior of the model trained on different motion models is consistent with our intuition, where the model trained on flow-only preserves the content and refines the edges, while the model trained only on the piecewise-affine model struggles with preserving identity. On the other hand, the model trained on different motion models falls in the sweet-spot where it addresses user edits faithfully while adding content as needed.

Architecture ablation. Prior work relies on using Image-CLIP embeddings or DINO features to encode the information of the content being inpainted or inserted into the image [239, 17]. The CLIP features are a reminiscent of the way Stable-Diffusion is trained with cross-attention with text CLIP embeddings. However, as we believe that CLIP features only carry semantic features that are too weak to pass useful information about the reference structure. We use a cross-reference mechanism, similar to Masa-Ctrl [13], and unlike prior work, we completely remove the CLIP cross-attention layer. To validate our design decision, we compare using CLIP image embedding of the reference for cross-attention as opposed to the cross-reference-attention we propose. We observe that when relying only on CLIP embeddings, the model struggles in harmonizing the edited regions as shown in Fig. 5.11. We find that the ablated model is conservative, and cannot address secondary effects like reflections.

Quantitative comparison. We evaluate our ablations on a held-out validation dataset from our video dataset. In the table on the right, Table 5.2, we show that the model trained with flow-data and affine-motion are the top performers on perceptual loss on both types of test and that dropping the cross-reference attention and replacing it with CLIP embedding causes a severe drop in performance.

Table 5.2: **Quantitative ablations.** Perceptual loss on a held-out validation set from our video dataset.

Model & Training Data	Test Data	LPIPS ↓
Piecewise affine	Piecewise affine	$\mathbf{0.231 \pm 0.007}$
	Flow-based	0.220 ± 0.007
Flow-based	Piecewise affine	$\mathbf{0.229 \pm 0.007}$
	Flow-based	$\mathbf{0.190 \pm 0.007}$
Both motion models (no cross-ref attn)	Piecewise affine	0.327 ± 0.007
	Flow-based	0.269 ± 0.008
Both motion models (Full method)	Piecewise affine	$\mathbf{0.231 \pm 0.007}$
	Flow-based	$\mathbf{0.196 \pm 0.007}$

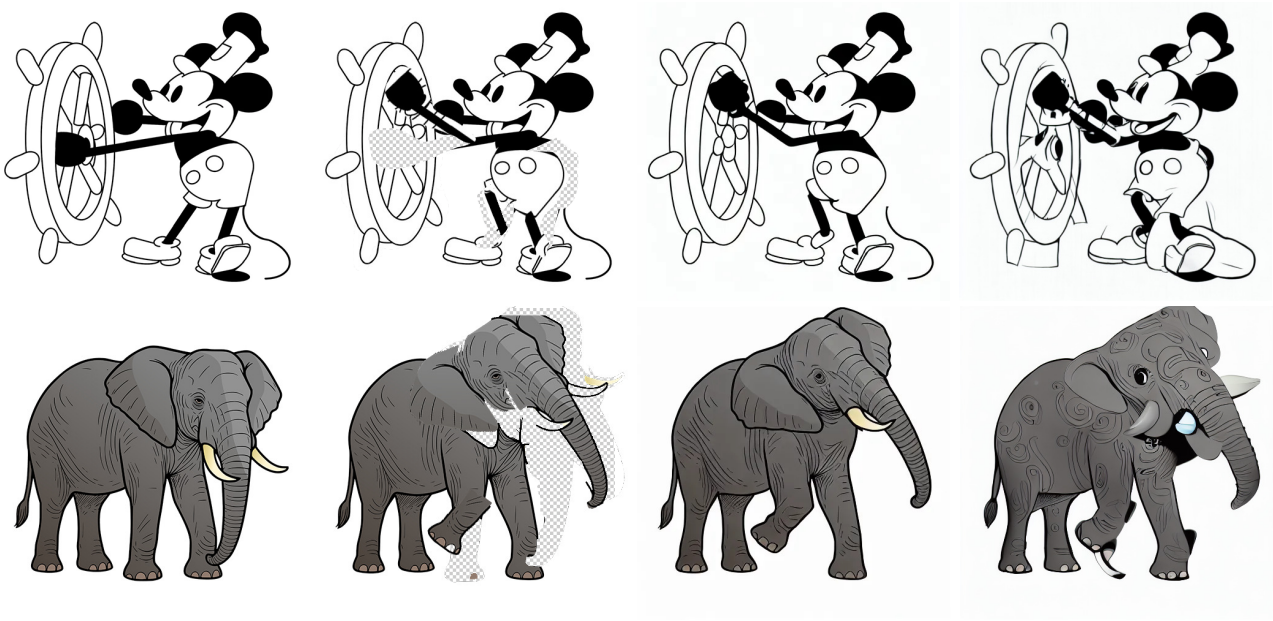


Figure 5.14: **Beyond real photos.** Despite training the model on exclusively real videos, we find that the model can generalize to new domains beyond the training data, like cartoons and vector art. Photos are using public domain materials and CC licensed images.

Detail extractor inputs. To better understand how the model is utilizing the reference image in preserving the image details, we ask, what would happen if the provided "reference image" is completely independent from the provided "coarse edit?" Note that this case never occurs in training, but by modifying the inputs, we can gain insights on the inner workings of the model. In Fig. 5.13 we show how the model behaves when provided an unrelated reference and edited images, and highlight diverse



Figure 5.15: **Limitations**. Since the model can was only trained on spatial edits by rearranging the parts of a single image, the model struggles in inserting objects from outside the original image. Here we see that the model attempt to stylize the bunny to have an appearance similar to the teddy bear, but struggles to preserve the bunny's identity. Photos sourced from ©Unsplash.

samples from the model. We notice that the model preserves the "style" and global appearance of the reference image, while preserving the spatial structure of the coarse edit. Intuitively, the reference image provides a sample of a clean image is supposed to look like, while the coarse edit provides the guidance of the spatial structure that the user intends to keep. This also touches on recent style transfer work [253] that achieves style transfer through shared attention layers, which is similar to the effect we see here where the model transfers the style of the reference to the coarse edit. In Appendix 5.6.4 we provide additional quantitative analysis ablating the inputs of the detail extractor and synthesis UNets, like the disocclusion masks and the denoising timestep embedding.

5.4.4 Generalization beyond real photos

While we only use video data of real videos, and filter out the majority of non-photorealistic videos in our training data, we explore the model's ability to generalize to completely new domains. In Fig. 5.14 we show the model's ability to generalize to new domains, like cartoons and vector art. We find that the model smoothly re-connects any disconnected parts of the image, and correctly re-synthesizes the art's outline that was lost in the editing process. On the other hand, we find that SDEdit

fails on all of those domains.

5.5 Limitations and conclusions

We present a method of assisting artists in photo editing through generative models while retaining a large level of control that traditional editing pipelines provide. We observe that with the appropriate motion model, we can use videos to train a model that can serve as a direct plugin in the editing process. We hope that our work inspires future editing research that can simply remove the cumbersome last-mile work by the press of a button.

Our generative model is trained for spatial compositions using video data. It can spatially recompose parts of the image but would struggle to insert objects from a completely different image as shown in Figure. 5.15. Furthermore, we inherit the limitations of Latent Diffusion Models, which we use as our base models, especially for generating hands, faces, and small objects.

5.6 Appendix

5.6.1 Reproducibility

To ensure the reproducibility of our results, we plan to release our code along with a version of the model trained on a public video dataset. We use the public Moments in Time dataset (MiT) [254] for our open-source model, due to the similarity of the types of videos in our dataset, as our dataset contains stock-like clips similar to the ones in Moments in Time [254] and WebVid10M [255]. We use 700k pairs of frames from MiT in contrast to 2.4M in the main model to train the model. We avoid using the larger WebVid10M dataset as it was recently taken down and the legality of using it is unclear. In Figure 5.21 we show that the open source model can achieve similar effects of addressing

secondary artifacts like shadows and reflections.

5.6.2 Samples of our training data

In Fig. 5.23 we show samples processed from our internal dataset that we used to train the model (show ref frame, target frame, flow warped, affine warped frames). Our videos come from stock-like internal dataset that is free of watermarks, unlike the commonly used public video datasets like WebVid10M [255].

5.6.3 Additional comparisons with text based methods

While text based editing methods lack the spatial control required to recompose photos precisely, we include additional qualitative comparisons for a comprehensive evaluation in Figure 5.22. We compare against InstructPix2Pix [12] and MasaCtrl [13], and include our main baseline, SDEdit [19] for reference. To preserve input identity, MasaCtrl relies on a DDIM inversion step to reconstruct the input. However, inversion is not always reliable and can result in images that are similar in the high level appearance but with a different content from the input, as we show in the fox example. In contrast, we pass the input in a feed forward manner that allows the network to reliably preserve the input identity. On the other hand, InstructPix2Pix either leaves the input image intact with minimal changes, or severely alter the image identity, making it unreliable for spatial editing. We find that SDEdit is the most reliable baseline as it can take the user edit directly as an input, improving the spatial controllability. As a result, we use SDEdit as our primary baseline in the main paper.

5.6.4 Ablation on models inputs

For a comprehensive ablation study of the inputs to the detail extractor and synthesis UNets. We analyze the role of the mask to the different UNets, and we experiment with dropping the timestep embedding from the detail extractor UNet. Dropping the time embedding in the detail extractor is equivalent to doing a one-time feature extraction, which would make it similar in style to using CLIP or DINO features rather than doing a feature extraction that depends on the current denoising step. We finetune our main model for each of these configurations on images generated using WebVid-10M as we no longer have access to the original internal data. For fairness, we also finetune the main model on the same dataset. We show the results in Tab. 5.3. Overall, timestep embedding in the reference UNet is essential. Providing the mask to the reference UNet is not needed, but the performance difference is within the standard error.

We find that providing the timestep embedding is essential for performance, which indicates that the detail extractor network extracts different features from the reference throughout the denoising process. This supports the intuitive understanding that the image generation process requires different levels of details for each step, as diffusion model generally starts by generating the coarse structure of the output and then synthesizing the higher frequency details later on. For the disocclusion mask, we find that the providing the mask is essential to the synthesis network, but provides no additional information to the detail extractor network. We also ablate including the noisy reference in the detail extractor UNet, and find that its effect on performance is negligible as expected.

Table 5.3: **Ablation on models inputs**. Here we ablate the inputs to the detail-extractor and synthesis UNets. We find that the providing the mask is essential to the synthesis network, but provides no additional information to the detail extractor network. We also find that providing the timestep embedding is essential for performance, which indicates that the detail extractor network extracts different features from the reference throughout the denoising process.

Motion model	Flow motion model			Piecewise affine motion model		
Method	**LPIPS** ↓	**SSIM** ↑	**Flow (px)** ↓	**LPIPS** ↓	**SSIM** ↑	**Flow (px)** ↓
w/o mask in either UNet	0.277 ± 0.01	0.615 ± 0.01	$\mathbf{35.554 \pm 2.14}$	0.294 ± 0.01	0.596 ± 0.01	$\mathbf{34.066 \pm 2.19}$
w/o mask in detail ext.	$\mathbf{0.187 \pm 0.01}$	$\mathbf{0.715 \pm 0.01}$	$\mathbf{35.682 \pm 1.88}$	$\mathbf{0.220 \pm 0.01}$	$\mathbf{0.667 \pm 0.01}$	$\mathbf{36.051 \pm 2.09}$
w/o timestep in detail ext.	0.504 ± 0.01	0.498 ± 0.01	71.769 ± 3.58	0.579 ± 0.01	0.457 ± 0.01	73.004 ± 3.49
w/o noisy input in detail ext.	$\mathbf{0.207 \pm 0.01}$	$\mathbf{0.699 \pm 0.01}$	$\mathbf{37.654 \pm 2.03}$	$\mathbf{0.242 \pm 0.01}$	$\mathbf{0.655 \pm 0.01}$	$\mathbf{37.126 \pm 1.99}$
ours	$\mathbf{0.194 \pm 0.01}$	$\mathbf{0.708 \pm 0.01}$	$\mathbf{35.716 \pm 1.93}$	$\mathbf{0.232 \pm 0.01}$	$\mathbf{0.657 \pm 0.01}$	$\mathbf{35.785 \pm 1.93}$

5.6.5 Quantiative evaluation using CLEVR

We used evaluation dataset generated through our dataset construction pipeline, as it provides a natural test set for spatial editing. It is challenging to construct large scale evaluation datasets without developing a novel motion model that we avoid training on. However, to substantiate our results further, we rely on the CLEVR dataset. CLEVR places objects with varying materials on a surface board, and includes multiple lights that showcase interesting shadows and shading. To levarage CLEVR to evaluate our model, we generate 50 random collection of objects, and rearrange each collection 3 times. Then, for an image in a given collection, we warp it to match the two other arrangements. This way we can automatically construct coarse edits, and have access to ground truth data for quantitative evaluation at the same time. In total, the test dataset consists of 300 edits. In Fig. 5.16, we show two samples and the outputs of MagicFixup against SDEdit and DragDiffusion. We find that our method synthesizes new shadows and harmonizes the objects layering. On the other hand, while SDEdit can preserve the target arrangements, and DragDiffusion struggles to spatially relocate the objects. Since DragDiffusion cannot rearrange the objects, we restrict our quantiative evaluation in Table 5.4 to our

method and SDEdit, and we find that our method outperforms the baseline in all metrics. While the CLEVR dataset is an imperfect test set, and out of distribution for our method, we find that the results further corroborate the robustness of our method and support our qualitative results.

Table 5.4: **CLEVR rearranging evaluation.** We generate a version of the CLEVR dataset where synthesize 50 random collections of objects, and rearrange each collection in three different ways. We evaluate the method's performance in realistically rearranging the objects against the ground truth.

Method	**LPIPS** ↓	**SSIM** ↑	**Flow (px)** ↓
SDEdit	0.156 ± 0.007	0.886 ± 0.002	26.02 ± 2.57
Ours	$\mathbf{0.078 \pm 0.007}$	$\mathbf{0.913 \pm 0.003}$	$\mathbf{8.03 \pm 1.20}$

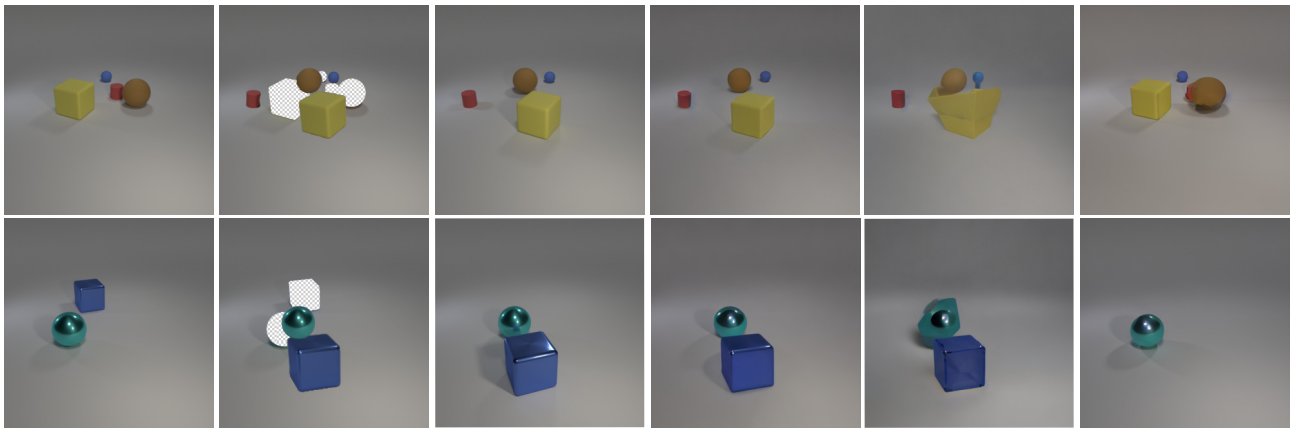


Figure 5.16: **CLEVR rearranging.** We modify the CLEVR dataset generation pipeline to generate random sets of objects in different spatial arrangements, and synthesize a coarse edit that corresponds to re-aligning one arrangement to the other. Compared to the baselines, our method can better preserve the objects and harmonizes it with the environments. We find that Drag based methods like DragDiffusion struggle in generating motion beyond reposing, and SDEdit drastically morphs the objects.

5.6.6 Collage transform interface

To facilitate creating user edits quickly, we created our own interface that supports the user selecting any object or parts they would like to edit, and make the edit by apply an affine transformation,

duplication, or deletion. We show our user interface with an example demonstrating its usage in Figure. 5.24 The interface allowed our users to create edits smoothly without any prior editing experience. Several of the edits used in this paper were created by novice users with no editing background. Beyond the simplicity of the interface, we maintain a dense correspondence map between the pixels in the original image and the edit. The correspondence maps are critical to fairly compare against the baselines that take drag handles or dense flow as an input, as we can directly use the correspondence to compute the needed input.

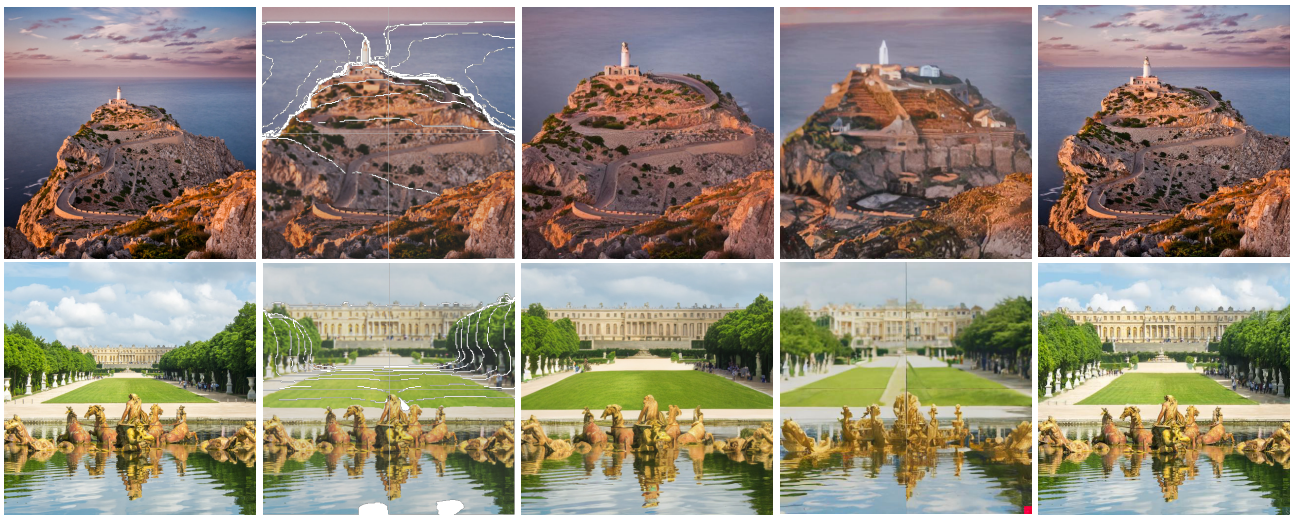
5.6.7 Expanded user study with SDEdit

Since SDEdit [19] is our primary baseline, we conduct an additional study only comparing our method with SDEDit, and significantly more user edits. We used 30 diverse photo edits, with 27 students participating and voting for all pairs of images. We conducted the study similar to the user study described in Section 5.4, in a 2-alternative forced-choice (2AFC) format. For 80% of the edits, at least 75% of the users preferred our method. For the remaining images, except for one image, users preferred our method $65-80\%$ of the time. For one image in out of domain edit (editing a non-realistic artistic painting), users preferred both edits almost equally likely (52 $\%$ of users preferred SDEdit).



reference I | coarse edit I_{coarse} | ours (sample 1) | ours (sample 2) | SDEdit [Meng et al. 2022]

Figure 5.17: **Colorization.** Even though our motion models only included spatial transformations, we experiment with running the model on out of domain coarse edits. Surprisingly, we find the model to synthesize realistic colorized outputs. The model also cleans up uneven coarse edges. For example, the lightning drawn on the Mustang contains uneven curvy edges, and the model cleans it up nicely. We also show multiple samples to highlight the diversity of the outputs the model can generate to address these edits. Photos sourced from ©Unsplash.



reference I | coarse edit I_{coarse} | ours | SDEdit [Meng al. 2022] | ZoomShop [Liu et al. 2022]

Figure 5.18: **Perspective editing.** By unprojecting the scene, then reprojecting different regions using variable camera parameters, we can manipulate perspective and make distant objects appear larger. While it is time consuming to create a high quality perspective edit (ZoomShop [20] takes as long as 4 hours of manual labor), by using MagicFixup we can take a coarse reprojection and make it realistic. Here we attempt to reproduce the results from ZoomShop with our method, and include their results as a point of reference. Photos sourced from ZoomShop [20].



Figure 5.19: **3D transformations.** By unprojecting the image, applying 3D transformations on the unprojected point cloud, and reprojecting, we can achieve coarse 3D edits. We show that MagicFixup can addresses the artifacts generated from the reprojection, while preserving the image identity. On the other hand, we find DiffusionHandles [21] to alter the background identity on the right wall of the first example, the number of cars in the second example, and the shading of the plate and altering the identity of the spoon next to the white mug. Photos sourced from ©Unsplash.



Figure 5.20: **Photoshop's puppet warp.** We experiment with finer grain deformation using Photoshop's puppet warp feature. We deform the bonsai tree to resemble a dancing figure, and introduce a rough smile to a person's face. Our model then improves the realism of the edit. Note that in the second row, the model introduced a natural smile along with wrinkles around the mouth and eyes to display a more natural smile. Photos sourced from ©Unsplash and CC materials.



reference I | coarse edit I_{coarse} | output of internal model | output of open source model

Figure 5.21: **Comparison with model trained on public data.** To maximize reproducibility, we train a version of the model on public video datasets that we plan to publicly release and open source. Here we show that the model trained on public data can similarly address secondary artifacts like reflections, and clean up artifacts due to coarse selection and editing as shown in the first row example with a coarse segmentation of the fox. Photos sourced from ©Roeselien Raymond ©Unsplash.

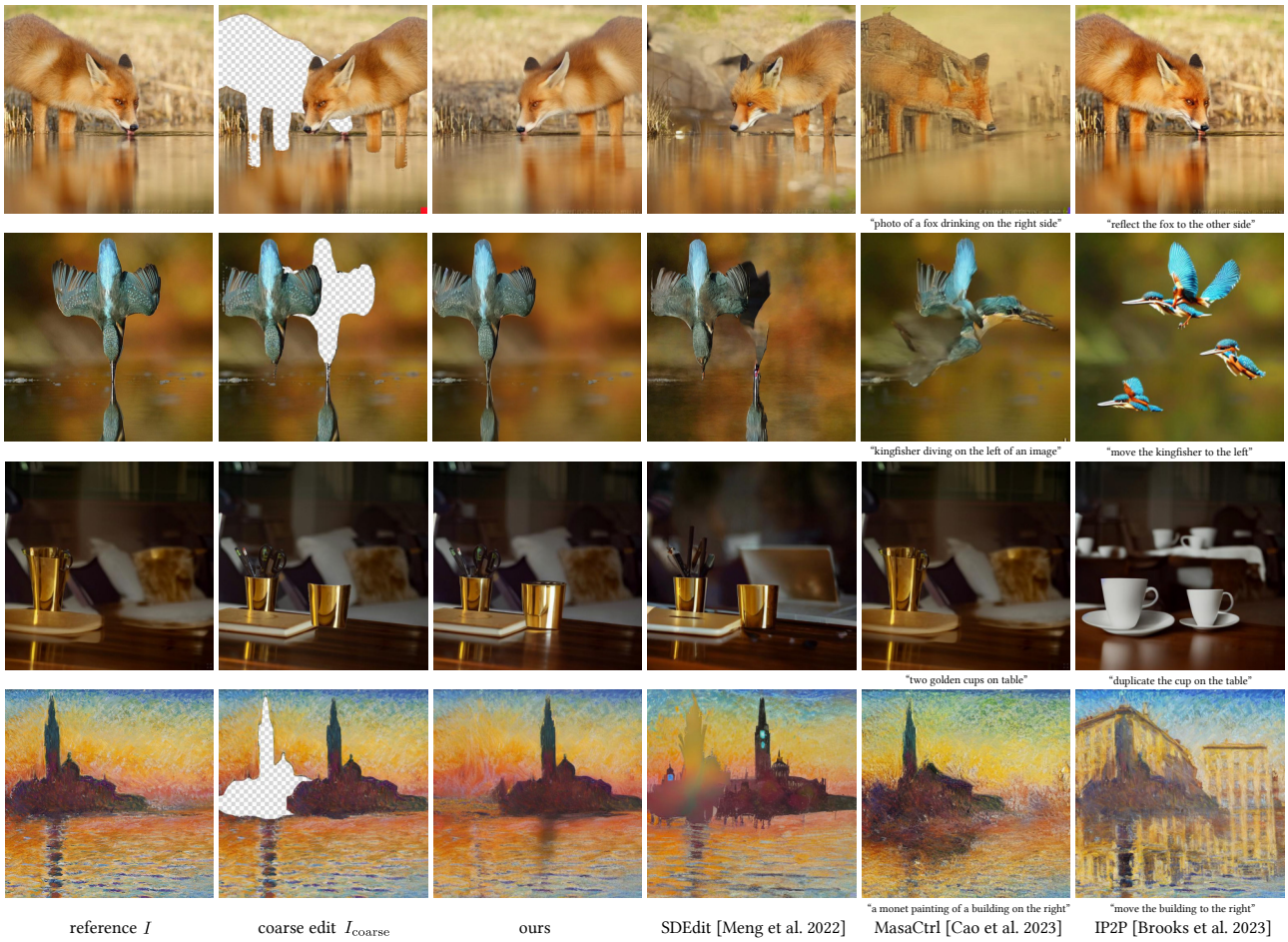
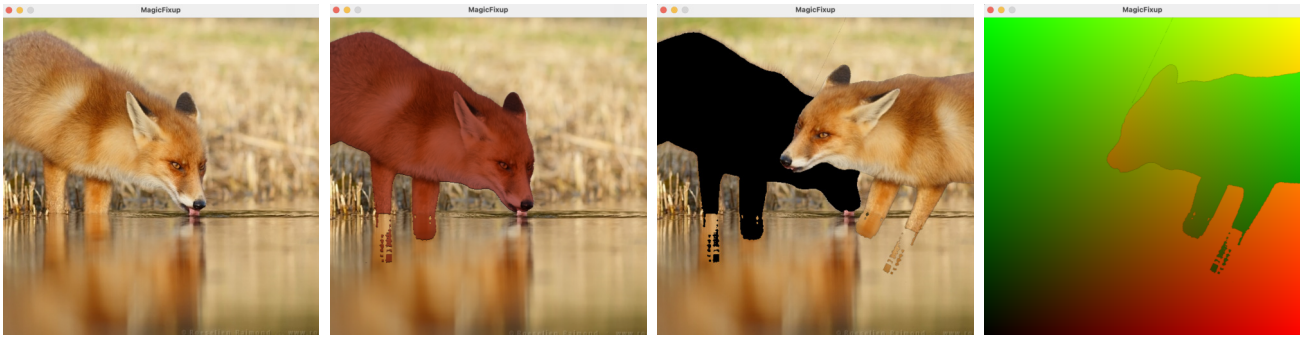


Figure 5.22: **Additional text comparisons.** We compare our method against text conditioned baselines MasaCtrl [13] and InstructPix2Pix [12]. To preserve the input identity, MasaCtrl requires a DDIM inversion step to the input image, which is prone to failing in reconstructing the input (as we show in the first two rows, the output identity is completely different from the input due to DDIM inversion failure), and in the cases where it succeeds in DDIM inversion, it is not possible to convey the user intent through a text prompt. Similarly, InstructPix2Pix either completely changes the identity of the image, or fails into editing the image to follow the text instruction. We show the text captions we used for both baselines under the image. We show the SDEdit [19] output as a reference, and we see that it is much more effective in following the user edit than the text baselines, which is why we rely on it as our main baseline. Photos sourced from ©Roeselien Raymond, ©Unsplash, and public domain data.



Figure 5.23: **Dataset samples.** We highlight some examples from our dataset along with the outputs of the flow and affine motion models. Note that the flow model densely aligns the reference image to the target, while the affine transformations provide much coarser alignment. For example, notice in the last row that in the flow warped image, the woman's smile and facial expression is aligned with the target, while in the affine warped we only see an alignment through scaling and shifting the person's segmentation mask. Media sourced from ©Adobe Stock.



(a) input image (b) segment any part (c) edit with affine transformations (d) pixel correspondence maps

Figure 5.24: **Collage transform interface.** To create user edits while maintaining correspondences between the original image and the edit, we created the Collage transform interface. The user can select any object or part they would like to edit, apply the desired affine transformation, duplication, or deletion. The correspondence map that we maintain allow us to accurately and fairly compare against the baselines by computing flow or drag keyhandles using the correspondence maps. Photo sourced from ©Roeselien Raymond.

Chapter 6: Generative Multi-View Relighting for 3D Reconstruction



Figure 6.1: **3D reconstruction under extreme illumination variation.** We propose a method for 3D reconstruction from a set of images captured under strongly varying illumination. Our method recovers high-fidelity appearance details including specular highlights that prior state-of-the-art approaches cannot recover (*top baseline*: NeRF-Casting [22] with appearance embeddings, *bottom baseline*: NeROIC [23]).

6.1 Introduction

A common strategy for performing novel view synthesis is to recover a 3D representation from a collection of photographs of a scene and then use it to render novel views from unseen viewpoints. Most view synthesis approaches assume that the input images are all taken under the same illumination conditions, *i.e.*, the scene's appearance is assumed to be *static*. However, this assumption is frequently violated during capture: moving clouds may cause the intensity of the sun to vary, artificial light sources may turn on or off, and even the shadows cast by the photographer may interfere with the scene.

This issue becomes even more severe when dealing with unstructured photograph collections such as those scraped from the Internet, where an object may be imaged in environments with extremely different illumination conditions, such as the case shown in Figure 6.1.

One strategy for addressing illumination variation is to use a view synthesis model capable of representing appearance changes across captured images. A simple and popular approach is to parameterize the view-dependent radiance of the scene as a function of a per-image "latent code" [256, 257]. Such an approach is effective for modeling changes (*i.e.* the "base" color and texture of the object). However, the additional expressivity comes at a cost because it tends to "explain away" *all* view-dependent appearances (not just those due to variable illumination). For non-diffuse (shiny) objects, we find that methods using per-image latent code often produce unrealistic, mostly-diffuse reconstructions.

Other physics-based approaches explicitly recover the spatially-varying material properties of the scene in addition to a per-image representation of illumination [258, 23]. While inverse rendering provides a more physically meaningful image formation model than a view-dependent latent encoding, it suffers from the same ambiguities: appearance variation can be attributed to either a change in lighting or a change in viewing direction, and these models often mix up the two.

Recent works [259, 24] attempt to circumvent these ambiguities by training image diffusion models that relight an individual image to appear as if it were lit by a specified illumination condition. Though diffusion models can leverage strong priors on natural image appearance to assist in material/lighting disambiguation, single-image relighting is fundamentally ill-posed. As such, independently applying these models to relight multiple images to a single reference illumination often yields samples corresponding to completely different, mutually-incompatible explanations of object materials.

In this work, we first use a *multiview* image diffusion model to jointly relight all input images to match the lighting of a reference image selected from the set of input images. This produces relit images that are significantly more consistent than those from prior single-image relighting diffusion models. We then design a novel 3D reconstruction model based on NeRF [46, 22] to fit a 3D representation to those relit images, which enables rendering from novel viewpoints. Our 3D reconstruction model is designed to be robust to inconsistencies that may still be present in the output of the relit samples. Since the diffusion model has to implicitly estimate 3D shape to relight the input images, any errors in its shape estimation would translate to specular highlights not appearing at the correct locations. To resolve this, we use a per-image vector which we call "shading embedding". This vector is used to encode per-image perturbations to the surface normal vectors used by the shading model of the NeRF, which allows the model to account for errors in the model's underlying estimated shape.

We validate the method on both synthetic (Objaverse [260]) and real (NAVI [261]) datasets. We showcase the capability of reconstructing accurate geometry and accurate view-dependent appearance from images captured under extreme illumination variation. Extensive quantitative and qualitative evaluation shows that the proposed approach significantly outperforms state-of-the-art methods both qualitatively and quantitatively.

6.2 Related Work

3D Reconstruction and View Synthesis. Neural radiance fields (NeRF) [46] is a successful approach for reconstructing a 3D scene under a fixed illumination. Many improvements to NeRF have been developed to improve the modeling of effects such as refractions and reflections [262, 71, 263, 264, 22, 265, 266], but all of these approaches also rely on static lighting. NeRF-W [256] achieved high-quality

3D reconstructions from unconstrained "in the wild" images taken from the internet, which tend to exhibit a variety of inconsistencies in lighting, appearance, and geometry. One aspect of this approach was using per-image appearance embeddings within the learned mapping from spatial coordinates and viewing direction to color (but not volumetric density). This parameterization forces optimization to recover a single consistent 3D geometry model while allowing appearance to vary across input images. This allows per-image appearance variation to be explained away, but also allows many view-dependent effects to be incorrectly attributed to per-image variation, often resulting in diffuse and erroneous reconstructions. Due to its simplicity (in that it does not contain a physically-based image formation model), per-image appearance embeddings are frequently used in settings where lighting only varies slightly [59, 267, 268].

Inverse Rendering. Decomposing an image into its constituent physical components has been a central problem in computer vision for nearly half a century [269]. Most modern approaches leverage the success of the computer graphics community, which has produced accurate and efficient models for the "forward" problem of rendering an image from an underlying 3D model [270]. Many modern inverse rendering techniques use radiance fields that, rather than mapping a 3D location and viewing direction to an outgoing color, map a 3D location to material properties and surface normals, which are then rendered according to some estimate of incident illumination [271, 258, 272–281]. Because this problem is ill-posed, these techniques often critically rely on analytical priors to regularize the estimated scene decomposition. Although these models fully decompose the scene into its components and thereby enable editing and relighting, their rigidly specified rendering models tend to constrain the solution space of the recovered model and result in lower rendering quality than models designed solely for view synthesis.

Many prior approaches to inverse rendering depend critically on modeling illumination and material properties using low-frequency parameterizations such as spherical harmonics [23], spherical Gaussians [282], or MLPs that are strongly biased to produce smoothly-varying outputs [283]. All of these parametrizations necessarily limit the model's ability to recover and render high-frequency (*i.e.*, non-smooth) color variations caused by glossy surfaces. This limitation is avoided by NeRF-Casting [22], which models incident illumination by explicitly reflecting secondary rays into the volumetric representation, thereby enabling the reconstruction of extremely specular objects. The 3D representation used by our model builds upon NeRF-Casting.

Lighting Estimation. Because estimating every aspect of a 3D scene is a challenging task with a broad scope, many prior works have instead attempted to only reconstruct a model of illumination from input images [284–287]. Though this problem is still fundamentally underconstrained, it can be made tractable by learning a model from datasets containing light probes [288] or by relying on accidental light probes in the scene [289]. Modern generative approaches in this vein have shown that diffusion models are capable of developing an internal understanding of illumination [290], which is further validated by our work.

Generative Relighting. Diffusion-based generative models have recently been used for relighting individual images [291] and, closer to our approach, for 3D relighting [259, 24, 292]. This process circumvents the need to decompose the scene into physically-meaningful components and instead relies on a diffusion model to re-synthesize an input image as if it were illuminated by some reference illumination. Unlike our method, these approaches require the set of input images to have constant illumination, which is used in the first step to recover the 3D geometry of the object before relighting it. IllumiNeRF [24] samples a diffusion model multiple times to generate a collection of plausible

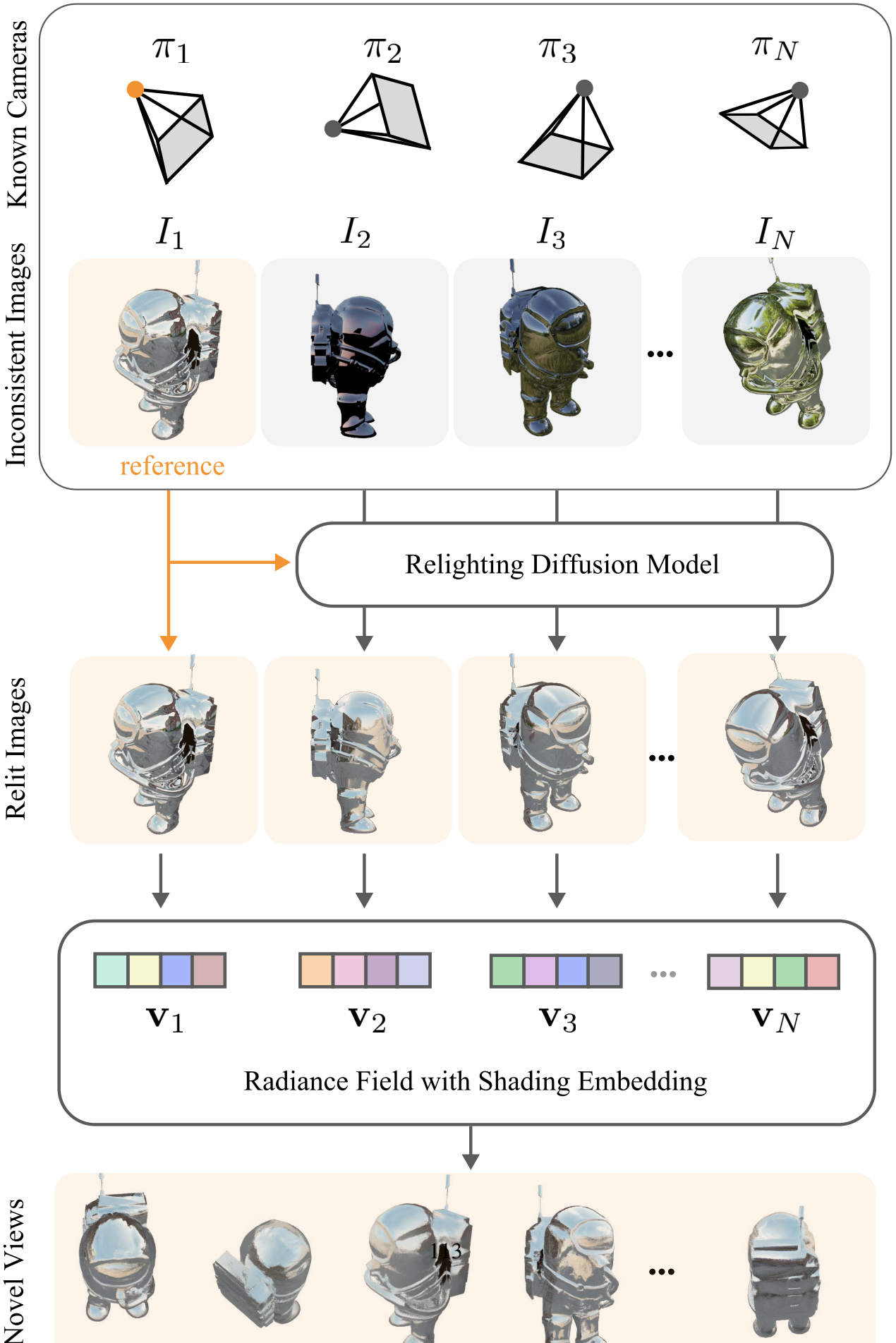
results, while Neural Gaffer [259] iteratively refines a set of relit images alongside a NeRF. Crucially, both models *independently* relight each input image given a reference illumination, which can lead to inconsistent relit images. Concurrent to our work, RelitLRM [293] uses a large reconstruction model to jointly reconstruct and relight an object from multiple images, but it also assumes that the input images are consistently lit. Additionally, all of these methods require a reference environment map as input, while we address a more general problem where we only have access to images. Our model works by relighting a set of differently-lit input images *simultaneously* to match the appearance of one of the input images, and it does not require any additional information, including constantly-lit input images or a reference environment map. We show that our approach not only yields significantly more consistent 3D reconstructions than prior work, but is also more practically applicable to a broader variety of inputs and capture settings.

6.3 Method

We aim to recover a 3D model from an collection of (posed) images of an object, where each image is illuminated by arbitrarily-varying lights. We use a two-step solution: in Section 6.3.1 we describe a relighting diffusion model that "harmonizes" the input images by making their illumination constant, and in Section 6.3.2, we describe a NeRF-based approach to convert the relit images produced by that diffusion model into a consistent 3D model. Our full pipeline is illustrated in Figure 6.2.

6.3.1 Relighting model

Our relighting model takes a set of N input images $I_1, ..., I_N$ of an object from multiple known camera poses $\pi_1, ..., \pi_N$, each of which is assumed to have been taken under a different illumination

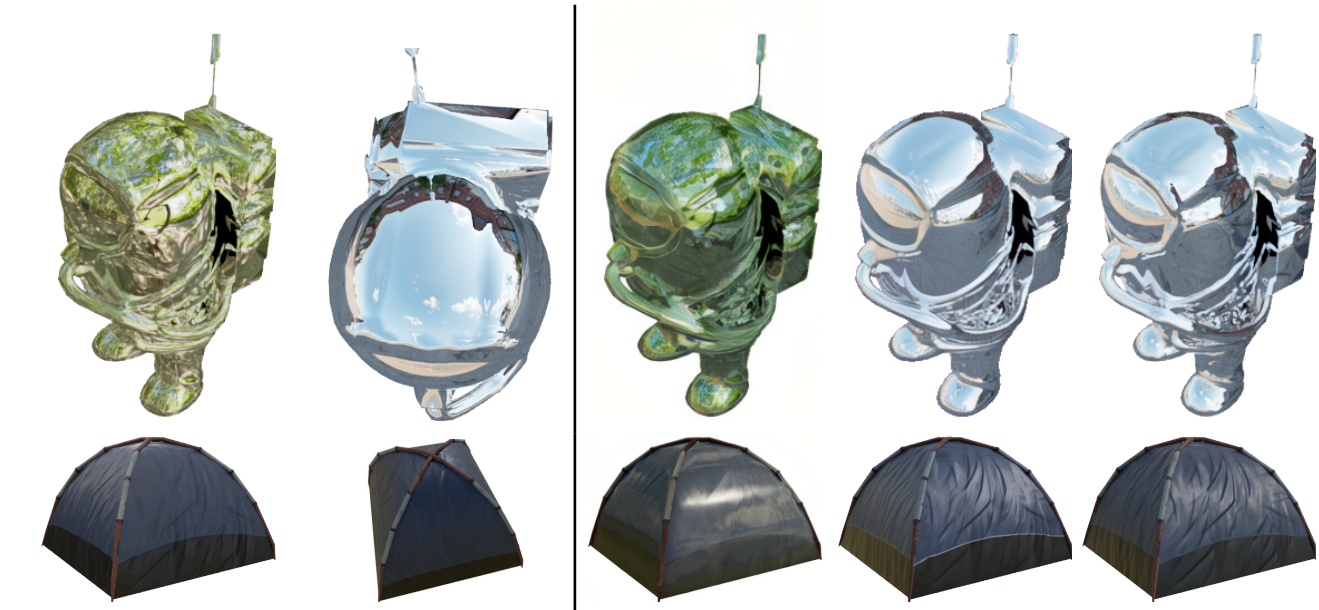

Known Cameras
π_1
π_2
π_3
π_N
Inconsistent Images
I_1
I_2
I_3
I_N
reference
Relighting Diffusion Model
Relit Images
$\mathbf{v}_1$
$\mathbf{v}_2$
$\mathbf{v}_3$
$\mathbf{v}_N$
Radiance Field with Shading Embedding
Novel Views

(see Figure 6.2, leftmost pane). The relighting model's aim is to generate a set of images of the same object from the same viewpoints, relit under the illumination from a designated reference image in the input set, which we arbitrarily select as the first image, I_1.

We build our relighting model using image diffusion, which can leverage strong natural image priors from large amounts of data. While recent works have used image diffusion models to relight individual input images based on a reference environment map [291, 24, 259], we argue that it is essential to relight the entire set of input images jointly. As demonstrated by the success of photometric stereo [294], leveraging multiple observations of the same material under varying lights reduces the inherent ambiguity between material properties and illumination, and it provides a strong cue for estimating shape. Using this same insight in the context of modern diffusion models leads to significantly more consistent relit images, as each sample from our model represents a unified interpretation of the object's material. Consequently, our method does not require ad-hoc solutions to address the issue of extremely inconsistent relit images, such as the use of appearance embeddings [24] or refinement of the relit images during reconstruction [259].

Our model's architecture follows prior work on view synthesis that extends image generative models to the multiview or video case [295–297] — we will analogously extend a single-image relighting model to multiple views, each with a distinct light.

We construct our relighting diffusion model as a multiview latent model that denoises a set of N latent codes $\mathbf{z}_1, ..., \mathbf{z}_N$, one for each input image. Unlike generative models that synthesize images from only random noise, the observed images $I_1, ..., I_N$ contain a significant amount of valuable information about geometry and illumination. As such, when denoising the ith latent code, $\mathbf{z}_i$, the denoising network should use the geometry encoded in image I_i as well as the lighting of the reference image I_1. To accomplish this, we pass the "clean" latent corresponding to the ith image I_i and an encoding of its



(a) Input Image to be Relit (b) Reference Image (c) IllumiNeRF [24] (d) Our Output (e) Ground Truth

Figure 6.3: **A comparison of our multiview relighting with prior work on single-image relighting.** Our method first relights a set of inconsistently-lit images (one of which is shown in (a)) to match the illumination of a selected reference image (b) in that set. Single-image relighting techniques such as IllumiNeRF [24] (c) struggle to disambiguate geometry, lighting, and materials, leading to an inaccurate relighting. In contrast, our model jointly relights a set of inconsistently-lit frames, which reduces ambiguities and results in a significantly more accurate result (d) when compared with the ground truth (e).

pose π_i into the network that denoises the ith latent $\mathbf{z}_i$. To jointly reason about the reference lighting, we use 3D self-attention blocks [297] for the latent codes, and we use cross-attention in between them and the camera poses. This structure allows the geometry and camera pose of the ith image and the lighting of the reference image to be directly observed by the network denoising the ith latent $\mathbf{z}_i$.

Because our input is an unordered set of images, we follow CAT3D's approach [297] of replacing temporal embeddings with camera poses which we encode as raymaps [298, 299]. To inform the network which image should be used as the reference, we concatenate the reference image with a "reference map" consisting of a single-channel image of ones, and we concatenate all other images with a single-channel image of zeros. Results from this relighting model component are shown in Figure 6.3.

6.3.2 3D Reconstruction

Applying the relighting model from Section 6.3.1 to the input images yields a set of "harmonized" input images that appear to have been lit under a single consistent illumination. With these images we can recover a 3D model without solving the difficult problem of handling variable illumination within a 3D reconstruction pipeline. We build our reconstruction model on NeRF-Casting [22], a view synthesis technique that achieves state-of-the-art results for reconstructing highly specular objects under constant illumination.

Although the relit images produced by our relighting diffusion model appear largely consistent when visually inspecting individual images, they still exhibit slight inconsistencies. As we will demonstrate, if the outputs of our diffusion model are used to naïvely train a NeRF that is not robust to this subtle variation (such as the common appearance-embedding based approach of NeRF-W [256]), these inconsistencies may be absorbed into the view-dependent parameterization of the NeRF, which results in novel view synthesis results with highly unrealistic and distracting "flickering" temporal artifacts as the camera moves. We must, therefore, be careful to parameterize appearance such that optimization is robust to this variation — resolving these inconsistencies is a primary goal of 3D reconstruction.

We address this problem by observing that the subtle inter-image errors made by the diffusion model are usually due to the specular highlights being slightly "warped" relative to the ground truth. This suggests that the diffusion model has small errors in its implicit estimate of the surface normals of the object, which causes the specular reflections to be tilted in slightly incorrect directions. We therefore resolve this by using a per-image "shading embedding": an embedding vector $\mathbf{v}_i$ is used to warp the normal vectors of the corresponding ith training image, and those warped normal vectors are provided as input to the network that models the shading (and therefore, appearance) of the image.

More concretely, the normals of the ith image at a 3D coordinate $\mathbf{x}$ are computed as:

$$\mathbf{n}_i(\mathbf{x}) = \text{normalize}(\text{MLP}(\mathbf{f}(\mathbf{x}), \mathbf{v}_i)) \,, \tag{6.1}$$

where $\text{MLP}(\cdot)$ is a multi-layer perceptron with 3 layers each with 128 hidden units, $\mathbf{f}(\mathbf{x})$ is a spatially-varying feature parameterizing the normals (identical to the one used by NeRF-Casting), and the $\text{normalize}(\cdot)$ operator turns the output of the MLP to a unit-length vector.

The per-image surface normals defined in Equation 6.1 can slightly alter the directions in which secondary rays are reflected by the model, which allows optimization to account for errors in the output of the relighting diffusion model. We prevent the normals from changing significantly between different images by tying them to the normals corresponding to the underlying NeRF density field, as in NeRF-Casting [22]. As we will demonstrate later, this shading embedding yields more accurate reconstructions and renderings than the standard appearance embedding approach [256] used in prior work for relighting [259, 24].

6.4 Architecture and Training

Training Data. The relighting model's training dataset is key to its ability to generalize across different illumination conditions, materials, and geometries. Due to the ease of rendering a large set of synthetic data, we rely solely on synthetic data for training, and we experimentally verify that our method generalizes to real captured photographs. We render our dataset using assets from a dataset of ~ 300K high-quality objects, similar in appearance to Objaverse but with more diverse materials. We additionally augment this dataset by rendering another copy after replacing all materials with perfect mirrors, which we found improved reconstruction quality for both highly-specular objects and (perhaps

surprisingly) for standard mostly-diffuse objects. We render the objects using about 700 environment maps taken from Poly Haven [300], which we augment by randomly rotating azimuthally.

Diffusion Model. We initialize our model using an image generation latent diffusion model similar in architecture to Stable Diffusion 1.5 [301] that had been trained on a large dataset of images. The image input of our diffusion model is a $512 \times 512 \times 3$ resolution image that is encoded into $64 \times 64 \times 8$ latents. To enable classifier-free guidance (CFG) [302], we randomly mask out the attention to the reference image to force the model to do "unconditional relighting" and relight the input views with an arbitrary lighting. For all our experiments, we use a CFG value of 3. Because the speed and memory requirements of training the model are inversely proportional to the number of synthesized views, we train our model in stages: We first train the model to relight 8 frames for 200K steps, then 16 frames for 100K steps, then 32 frames for 50K steps, and finally 64 frames for another 50K steps. Note that our model parameters are independent of the number of input views, since the cross-attention operation can scale to an arbitrary number of images. Our model was trained on 64 5th-generation TPUs with a total batch size of 64, which took approximately two weeks. At inference, we set the number of images that our diffusion model simultaneously relights to $N = 64$

3D Reconstruction. The optimization of each radiance field uses a similar number of parameters to those used in NeRF-Casting [22]. Optimizing a model from a set of harmonized images takes roughly 30 minutes on 16 NVIDIA A100 GPUs, and rendering a single 512×512 image takes about 0.5 seconds on the same hardware.

6.5 Results

We evaluate our method on two datasets: synthetic objects from Objaverse [260] and real captured objects from NAVI [261]. The synthetic dataset comprises two components: 8 standard textured assets whose materials range from mostly-diffuse to glossy, and 12 shiny assets containing objects with perfectly-reflective materials, designed to highlight the challenge of recovering accurate view-dependent effects. We render the synthetic Objaverse dataset using environment maps from HDRIMaps [303]. Each scene contains 64 training images and 36 testing images, all with different random poses. Each training image was rendered with distinct illumination, and the test set was rendered using the illumination of the reference image from the training set.

6.5.1 Comparisons to State-of-the-Art Methods

We compare our approach with two methods designed for 3D reconstruction "in the wild": NeROIC [23] and NeRF-Casting [22] with standard appearance embeddings for each image (which we call "NeRF-Casting + AE" or "NeRFCast + AE" for short). We also compare our approach with the recent diffusion-based relighting method IllumiNeRF [24]. IllumiNeRF requires a target environment map for relighting and we provide it with the ground truth illumination, which is a significant advantage over other techniques as the full environment map contains more information than the reference image. For methods with appearance embeddings (ours and NeRF-Casting + AE), we render novel views using the same embedding as the reference image. Similarly, when rendering test images using NeROIC, we use the estimated illumination of the reference image.

Synthetic Results. In Figure 6.4, we qualitatively compare our method's novel view synthesis re-



Figure 6.4: **Visual comparison of novel view renderings on the Objaverse dataset**. (b) We show sample input images under extreme illumination variation. (c) Adding a per-image latent code to NeRF-Casting [22] ("NeRFCast + AE") cannot accurately explain away the variations, leading to erroneous reconstruction. (d) Due to the ill-posed nature of the problem, inverse rendering-based methods such as NeROIC [23] tend to produce lower-quality renderings with mostly diffuse appearance. (e) IllumiNeRF [24] leverages diffusion prior for single-image relighting but produces inconsistent output samples, resulting in excessive blur in rendered novel views. Note that IllumiNeRF requires access to the target illumination's environment map as input. We provide IllumiNeRF with the ground truth environment map corresponding to the reference image (a), while other methods only have access to the reference image itself. (f) Our method renders accurate appearance with specular highlights close to those in the ground truth images (g).

sults with prior state-of-the-art approaches. Our results appear significantly more accurate, and all baselines struggle to recover specular highlights. This is particularly apparent in the first and third rows, demonstrating the recovery of fine details in the reflections of perfectly specular objects. The quantitative results in Table 6.1 show that our method significantly outperforms all baselines on both splits of the dataset, even when we provide the IllumiNeRF [24] baseline with true environment maps corresponding to the reference image.

Table 6.1: **Novel view relighting on Objaverse.** Our approach of relighting and robustly optimizing a radiance field outperforms previous state-of-the-art methods in rendering novel views under target lighting. We outperform prior work despite giving the baseline a significant advantage by providing it the true lighting.

	Standard Assets			Shiny Assets		
Method	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**
NeROIC [23]	26.13	0.935	0.088	22.14	0.880	0.113
NeRFCast [22] + AE	27.53	0.941	0.067	21.80	0.874	0.108
IllumiNeRF [24] w/ known light	29.22	0.958	0.057	23.46	0.881	0.095
Ours	**31.34**	**0.966**	**0.053**	**26.54**	**0.911**	**0.090**

Real Photographs. The top row of Figure 6.1 shows photographs we captured of a chrome-plated figurine in different environments. The resulting synthesized novel views show that our method can render accurate images corresponding to the selected reference illumination. In the bottom of Figure 6.1 and in Figure 6.5 we show reconstruction results on captured photographs from NAVI. The results show that our method can preserve shadows and specular highlights. Since NAVI also provides multiview data for each illumination, we can quantitatively evaluate the novel view synthesis performance on held-out views under the same illumination as the reference. We evaluate the method on all of the 18 NAVI scenes with at least 64 frames, and we show that our method also achieves the best quantitative results across all prior methods, as shown in Table 6.2. Since NAVI does not provide ground truth environment maps, we could not evaluate IllumiNeRF on this dataset.

Table 6.2: **Novel view relighting on real photos from NAVI.** We show that we outperform prior work on synthesizing novel views from "in-the-wild" images.

Method	**PSNR↑**	**SSIM↑**	**LPIPS↓**
NeRF-Casting [22] + AE	23.59	0.916	0.066
NeROIC [23]	24.01	0.918	0.079
Our Model	**25.55**	**0.929**	**0.060**



(a) Reference (b) Input Samples (c) NeRFCast + AE (d) NeROIC (e) Ours (f) Ground Truth

Figure 6.5: **Comparison on real world photos.** We use our method to reconstruct objects from in-the-wild photos taken in different environments. Our method can render novel views under the illumination conditions of any input image we select as the reference. Unlike prior work, our technique accurately preserves shadows (*e.g.* the bunny's ear shadow in the first row) and reflections (*e.g.* the box's handle in the second row and the car's specularities in the last row) that appear in the reference image we would like to match.

6.5.2 Ablation Studies

Here, we provide ablations designed to test the different components of our model.

Number of Frames. A major benefit of our approach is that it jointly relights all frames. Table 6.3 shows our view synthesis results when relighting the images using different numbers of simultaneously-predicted frames N. Increasing the number of predicted frames improves 3D reconstruction, hence our choice for predicting $N = 64$ images.

Dataset. Next, we demonstrate the importance of including purely-specular objects in our training data. To do this, we train two 16-frame versions of our model for 70K steps, with and without the

Table 6.3: **The effect of the number of frames N output simultaneously by our model.** Increasing the number of frames simultaneously processed by our model improves its performance, therefore we use $N = 64$.

	Standard Assets			Shiny Assets		
# frames N	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**
1 frame	28.39	0.943	0.079	23.85	0.889	0.105
8 frames	30.43	0.956	0.067	25.62	0.901	0.095
16 frames	30.49	0.958	0.063	25.52	0.900	0.097
32 frames	30.72	0.961	0.060	26.14	0.908	0.090
64 frames (ours)	**31.34**	**0.966**	**0.053**	**26.54**	**0.911**	**0.090**

purely specular data augmentation described in Section 6.4. Table 6.4 shows that including the specular materials in the training set improves performance on specular objects and, perhaps surprisingly, that it is also beneficial for our general less-specular (but not perfectly-diffuse) dataset.

Table 6.4: **Training dataset ablation.** We show that training on assets with standard materials as well as objects with highly reflective materials improves performance for both types of assets.

	Standard Assets			Shiny Assets		
Training data	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**
Standard assets	29.16	0.938	0.066	22.34	0.874	0.113
Standard + shiny assets	**29.26**	**0.939**	**0.066**	**22.91**	**0.877**	**0.111**

Shading Embeddings. Table 6.5 demonstrates the effect of our shading embeddings on the 3D reconstruction of the shiny synthetic scenes harmonized by our relighting model. As explained in Section 6.3.2, the shading embeddings allow the model to slightly shift the normal vectors used for computing appearance separately for each image, which enables absorbing small inconsistencies in the relit images. This improves upon the same NeRF model with no per-image embeddings ("No embeddings"). In contrast, using standard embeddings as in NeRF-W [256] ("Shading embeddings") has an

adverse effect: the per-image codes allow the model to encode specularities as per-image diffuse color, which results in unrealistic and poor reconstructions.

Table 6.5: **Per-image embedding ablation.** We show that our shading embeddings improve performance compared to NeRF-Casting without any per-image code and that standard appearance embeddings have an adverse effect on reconstruction quality.

Method	PSNR↑	SSIM ↑	LPIPS↓
No embeddings	26.02	0.907	0.094
Appearance embeddings	24.38	0.896	0.096
Shading embeddings (ours)	**26.54**	**0.911**	**0.090**

6.6 Discussions and Conclusions

Limitations. Our approach requires object masks and accurate camera poses as inputs. Camera poses, in particular, can be challenging to compute from images of highly reflective objects that do not have reliable features for matching, but are required by most 3D reconstruction methods. However, there has been significant recent progress in camera pose estimation using learning-based techniques [304–307] which do not rely on explicit feature matching. We believe that leveraging strong generative priors for joint pose estimation and 3D reconstruction is an exciting direction for future research.

Conclusions. In this paper, we address the challenge of reconstructing 3D objects from images captured under extreme illumination variation. Our core insight is to simultaneously relight all input images to match the illumination of a chosen reference image, thereby harmonizing lighting conditions among input images. We achieve this by training a diffusion-based multiview relighting model. Using these relit images, we apply a 3D reconstruction model that is robust to residual inconsistencies, enabling accurate reconstruction of the object's shape and view-dependent appearance. Our method effectively reconstructs objects with complex shapes and materials from images taken under drastically different lighting conditions. Our work highlights a promising direction for leveraging strong

generative priors to tackle the ill-posed problems of 3D reconstruction.

Chapter 7: Composing Pretrained Generative Models at Inference

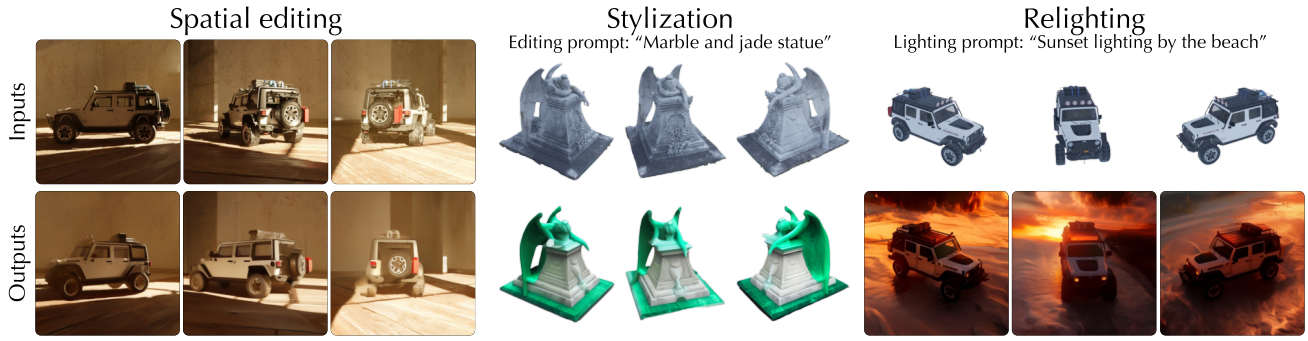


Figure 7.1: **Applications of coupled diffusion sampling.** Our approach enables lifting off-the-shelf 2D diffusion editing models into multi-view by combining the sampling process of 2D models with multi-view generation models to produce consistent edits. Here we showcase example view-consistent results using three 2D editing models: spatial editing, stylization, and text-based relighting.

7.1 Introduction

Diffusion-based image editing models have demonstrated unprecedented realism across diverse tasks via end-to-end training. Simply by curating paired editing datasets, we can perform relighting [259, 308, 309], spatial structure editing [310, 311, 34, 312], and stylization [313]. However, collecting and curating 3D data is significantly more challenging than 2D data. As a result, recent research has explored test-time optimization methods for multi-view editing that leverage pre-trained 2D image diffusion models [314, 315].

When editing a collection of photos of a 3D scene with a 2D model, we encounter the issue where each image is edited inconsistently, as shown in Fig. 7.2. To resolve those inconsistencies,

most existing methods [315, 259] rely on explicit 3D representations, *i.e.,* NeRF [46] or 3D Gaussian Splatting [316]. The advantage of using explicit representations is to avoid the need to train new 3D editing models. However, these methods typically require time-consuming optimization and dense coverage of the input view, limiting their applicability to real-time, real-world scenarios. Furthermore, the lack of a data-driven prior can lead to inconsistencies that result in blurring or variants of the Janus problem.

We follow a different approach. Instead of using 2D models to supervise and train an explicit 3D representation, we use 2D models to *guide* implicit 3D generative models. Recent multi-view diffusion models [317, 297] were proposed to perform novel view synthesis and generation using a data-driven prior. However, when naively attempting to edit a scene by editing a single image and synthesizing the remaining views, the outputs would lose the identity of the input, as shown in Fig. 7.2, because this process is under-constrained and many solutions exist. We address this by *steering* the multi-view model using the 2D editing model to sample a valid edit that is *faithful* to the input, and *consistent* across views. We call our novel diffusion sampling process *coupled diffusion sampling*. As shown in fig. 7.1, our approach enables multi-view consistent image editing across diverse applications, including multi-view spatial editing, stylization, and relighting.

While prior work [25, 26] explored combining diffusion models within a modality, we observe that such approaches do not maintain multi-view consistency and can stray from the editing objective as shown in Fig. 7.2. Our approach is motivated by the observation that any sequence of images generated by a pre-trained multi-view image diffusion model inherently exhibits multi-view consistency. To this end, we embrace an implicit 3D regularization paradigm by leveraging scores estimated from multi-view diffusion models during the diffusion sampling process. Specifically, for any multi-view image editing task leveraging a pre-trained 2D model, we couple it with a foundation multi-view dif-

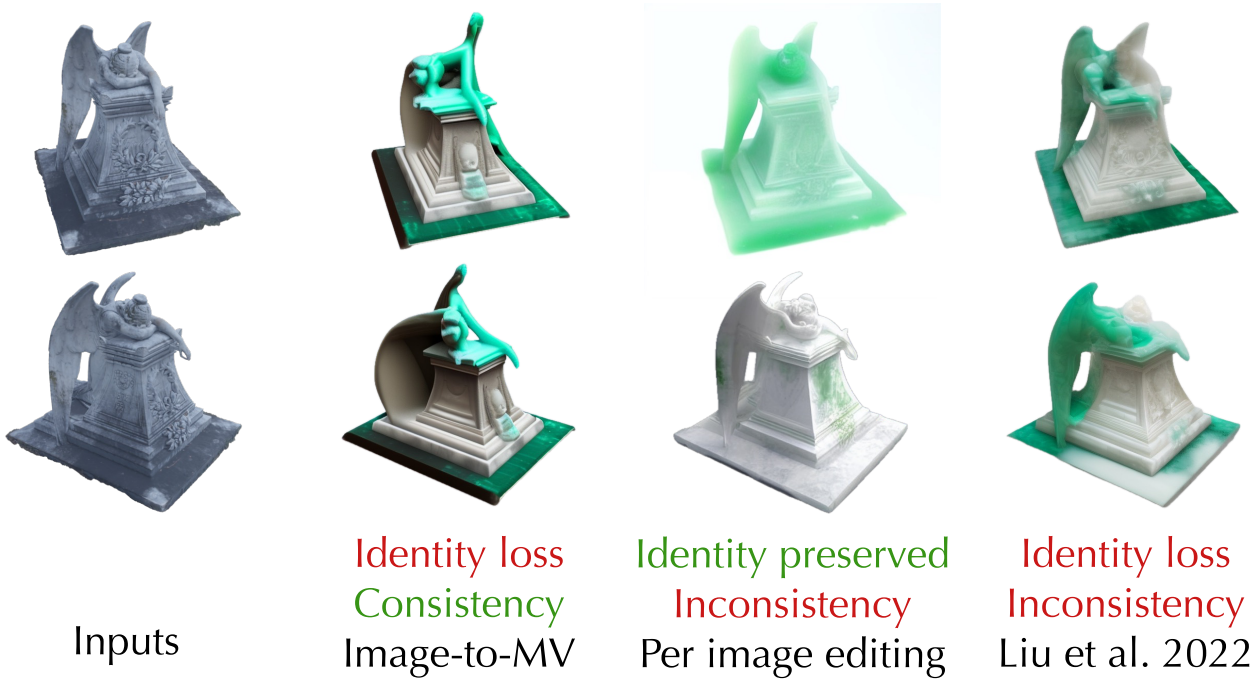


Figure 7.2: **Limitations of baselines.** Using image-to-multiview model conditioned on a single edited image cannot be faithful to the remaining input views, while editing each image individually with a 2D model produces inconsistent results. While prior work [25] proposes a method to compose diffusion models, their approach produces inconsistent and flickering results.

fusion model and perform sampling under dual guidance from both models. This process ensures that the resulting samples satisfy both the editing objective and multi-view 3D consistency, yet without any additional explicit 3D regularization or training overhead.

We propose a practical sampling framework to achieve the above-mentioned goal by steering the standard diffusion sampling trajectory with an energy term coupling two sampling trajectories. This method ensures that each sample from a given diffusion model remains within its own distribution while being guided by the others. In particular, samples from the multi-view diffusion model maintain multi-view consistency while being steered by the content edits from the 2D model. Conversely, the 2D model is steered so that its edits remain faithful to the identity of the inputs while being consistent across independently edited frames.

Our solution is conceptually simple, broadly applicable, and adaptable to a variety of settings.

We showcase its effectiveness across three distinct multi-view image editing tasks: multi-view spatial editing, stylization, and relighting. Through comprehensive experiments across each task, we demonstrate the advantages of our method over the state of the art. We further validate the generalizability of our approach to diverse backbones and latent spaces, and illustrate its effects on image and video generation, underscoring its promise as a general multi-view image editing engine with potential extensions to broader generative tasks.

7.2 Related work

Test-time diffusion guidance. Test-time scaling methods [318–320], such as rejection sampling or verifier-based search. In contrast, optimization-based guidance actively steers diffusion trajectories, offering a more efficient alternative. The diffusion sampling process can be steered with a classifier guidance [205], a differentiable objective function [321, 245], or by enforcing additional assumptions in the forward process, e.g., as in linear inverse restoration problems [322–324]. In cases where the edit can be characterized by a "prompt" edit, the model can be used to guide itself to edit the input [325, 13, 326]. In our method, we use diffusion models that capture complementary distributions to inform one another. Unlike prior work, this allows us to capture complex edits beyond what can be formulated by a differentiable function and to achieve multi-view editing without paired data.

3D and multiview editing. As diffusion models capable of producing high-quality 2D image edits [325, 13, 326, 21, 327–329], a natural question has been how to leverage those capabilities for 3D editing. One common approach is to optimize an explicit 3D representation, like NeRF, either by modifying the training dataset during the optimization loop [315, 330] or through score distillation sampling [314, 331–334]. However, both approaches are prone to visual artifacts, which is funda-

mentally caused by the fact that 2D diffusion models lack 3D consistency awareness. To address this fundamental challenge, prior work has directly trained multi-view diffusion models [335, 35, 336–338] for consistent editing. However, training a multi-view diffusion model for each individual editing task is computationally expensive, and suitable training datasets are scarce. In our approach, we propose reusing existing multi-view *generation* models [297, 317, 339] for multi-view *editing* by combining them with a 2D editing model, thereby incurring no additional training cost. In contrast to NeRF-based approaches, our method does not require a costly optimization process, as it relies solely on feed-forward sampling.

Compositional diffusion sampling. Compositional sampling methods for diffusion models have been proposed to combine the priors of multiple models. Examples include product-of-experts sampling [340, 341], which samples from the product distribution of individual models. However, this approach imposes a strict requirement that valid samples lie in the intersection of the supports of the models and fails when no such joint support exists. MultiDiffusion [342] and SyncTweedies [343] apply score composition for out-of-distribution scenarios like stitching panoramas or large images. On the other hand, our work emphasizes remaining within each model's prior distribution while steering generation toward satisfying cross-model constraints. Prior works [25, 26] address inference-time composition for diffusion models, but these works focus on the same data modality. In contrast, our work bridges 2D and 3D modalities to tackle the practical challenge of 3D data sparsity.

7.3 Method

7.3.1 Background

Diffusion Models. Let $x_0 \sim p_{\text{data}}(x_0)$ be a data sample and consider the forward noising process:

$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t \mid \sqrt{1-\sigma_t}x_{t-1}, \sigma_t I), \tag{7.1}$$

with a variance schedule $\{\sigma_t\}_{t=1}^T$. [204] proposes to train a neural network $\epsilon_\theta(x_t, t)$, where θ denotes network parameters, such that when starting with an initial noise $x_T \sim \mathcal{N}(0, I)$, it allows one to gradually denoise the sample to $x_0 \sim p_{\text{data}}(x_0)$ via

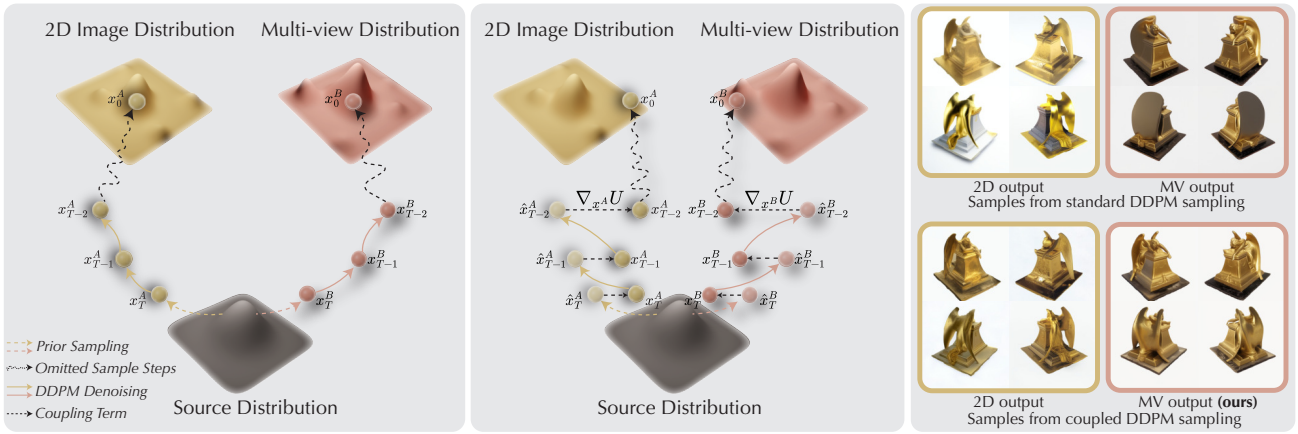
$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(x_t)) \tag{7.2}$$

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_t) + \sigma_t z, \tag{7.3}$$

where $\alpha_t = 1 - \sigma_t, \bar{\alpha}_t := \prod_{s=1}^t \alpha_s$. The next-step prediction x_{t-1} is obtained by computing the *clean* image estimate $\hat{x}_0$ and re-injecting a decreasing amount of random noise $z \sim \mathcal{N}(0, I)$.

7.3.2 Coupled DDPM Sampling

Problem. Given two diffusion models ϵ_{θ^A} and ϵ_{θ^B} for a shared data domain $\mathbb{R}^d$ and with a shared DDPM schedule, our goal is to obtain two samples $x^A, x^B \in \mathbb{R}^d$ such that they follow the data distribution prescribed by the pre-trained models $p^A_{\text{data}}(x)$ and $p^B_{\text{data}}(x)$, respectively, while staying close to each other. This objective can be interpreted as tilting the distribution $p^A_{\text{data}}(x)$ to be close to a sample $x^B(x) \sim p^B_{\text{data}}(x)$, and vice versa. We introduce a coupling function $U : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ that measures



(a) Standard DDPM Sampling (b) Coupled DDPM Sampling (c) Samples from Two Methods

Figure 7.3: **Overview of the proposed coupled sampling method.** Given two target statistical distributions modeled with diffusion models: (a) standard DDPM sampling generates two instances independently, using scores from each distribution, which leads to samples without spatial alignment; (b) in contrast, the proposed coupled DDPM sampling introduces coupling terms ∇U that pull the two sample paths together, producing spatially and semantically aligned outputs; and (c) as illustrated, the 2D samples get increasingly consistent with coupling, and the coupled multi-view samples (which we set to be our final output) are faithful to the input views.

the closeness of two samples. A natural choice is the Euclidean Distance, and in this work, we use $U(x, x') = -\frac{\lambda}{2}\|x - x'\|_2^2$ with a constant coefficient $\lambda \in \mathbb{R}$. Formally, our objective is written as

$$\max_{x^A, x^B} \mathcal{J}^A(x^A, x^B) + \mathcal{J}^B(x^A, x^B), \quad \text{where} \tag{7.4}$$

$$\mathcal{J}^A(x; x') := p^A_{\text{data}}(x) \exp U(x, \text{sg}(x')), \tag{7.5}$$

$$\mathcal{J}^B(x; x') := p^B_{\text{data}}(x) \exp U(\text{sg}(x), x'), \tag{7.6}$$

where sg denotes stop grad. Taking the gradients:

$$\nabla_x \mathcal{J}^i(x, x') = \nabla_x \log p^i(x) + \nabla_x U(x, x'), \tag{7.7}$$

for $i \in \{A, B\}$. Here, the additional term $\nabla_x U(x, x')$ biases the sample trajectory $\{x^i_t\}_t$ from the standard diffusion trajectory following $p^i(x)$ to satisfy the goal. Tilting diffusion model sampling

Algorithm 1 Coupled DDPM Sampling

θ_{2D}: Text2Image diffusion model
θ_{MV}: Text2MultiView diffusion model
$x_{T,2D}, x_{T,MV} \sim \mathcal{N}(0, I)$: initial latents
$x_{T,2D}, x_{T,MV}$ shapes: $N \times H \times W \times C$ where N is # of views
for $t \in T, ..., 0$ **do**
x_0 prediction:
    $\hat{x}_{0,MV} \leftarrow \frac{1}{\sqrt{\bar{\alpha}_t}}(x_{t,MV} - \sqrt{1-\bar{\alpha}_t}\epsilon_{\theta,MV}(x_{t,MV}))$
    $\hat{x}_{0,2D} \leftarrow \frac{1}{\sqrt{\bar{\alpha}_t}}(x_{t,2D} - \sqrt{1-\bar{\alpha}_t}\epsilon_{\theta,2D}(x_{t,2D}))$
DDPM step:
    $\hat{x}_{t-1,MV} \leftarrow \sqrt{\bar{\alpha}_{t-1,MV}}\hat{x}_{MV,0} + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_{t,MV}) + \sigma_t z$
    $\hat{x}_{t-1,2D} \leftarrow \sqrt{\bar{\alpha}_{t-1,MV}}\hat{x}_{MV,0} + \sqrt{1-\bar{\alpha}_{t-1}}\epsilon_\theta(x_{t,2D}) + \sigma_t z$
Coupled guidance step:
    $x_{t-1,2D} \leftarrow \hat{x}_{t-1,2D} - \sqrt{1-\bar{\alpha}_{t-1}}\lambda(\hat{x}_{0,2D} - \hat{x}_{0,MV})$
    $x_{t-1,MV} \leftarrow \hat{x}_{t-1,MV} - \sqrt{1-\bar{\alpha}_{t-1}}\lambda(\hat{x}_{0,MV} - \hat{x}_{0,2D})$
end for

towards inference-time reward functions or constraints has been widely studied for preference alignment [344] and inverse problems [324, 345], with gradient likelihood of a form similar to eq. (7.7), although typically under a fixed target. In contrast, in this work, the optimization target depends on another variable. The coefficient λ is a hyperparameter dictating the coupled guidance strength. A value of zero corresponds to standard DDPM sampling, and increasing it strengthens the coupling between the two models. Similar to prior guidance-based methods [346, 321, 205], applying too strong a guidance can cause the sampling process to collapse and produce out-of-distribution results.

Algorithm. Let $x_t^A, x_t^B \in \mathbb{R}^d$ be two data samples to couple, and denote $\hat{x}_{t-1}^A, \hat{x}_{t-1}^B$ to be the denoised output in DDPM using eq. (7.3). We compute the coupled samples by:

$$x_{t-1}^A = \hat{x}_{t-1}^A + \sqrt{1-\bar{\alpha}_{t-1}}\nabla_{\hat{x}_0^A} U(\hat{x}_0^A, \hat{x}_0^B) \tag{7.8}$$

$$x_{t-1}^B = \hat{x}_{t-1}^B + \sqrt{1-\bar{\alpha}_{t-1}}\nabla_{\hat{x}_0^B} U(\hat{x}_0^B, \hat{x}_0^A). \tag{7.9}$$

Notice that by fixing $\hat{x}_0^B$,

$$\exp U(\hat{x}_0^A, \hat{x}_0^B) \propto \exp -\frac{1}{2}\frac{\|\hat{x}_0^A - \hat{x}_0^B\|_2^2}{1/\lambda} = \mathcal{N}(\hat{x}_0^B, 1/\sqrt{\lambda}I) \tag{7.10}$$

we find the interpretation that $\exp U(\hat{x}_0^A, \hat{x}_0^B)$ assigns low energy to $\hat{x}_0^A$ close to $\hat{x}_0^B$ in during the sampling process, and similarly for x_t^B. This term effectively serves as a soft regularization that encourages two samples to stay close. The gradient term $\nabla_x U(x, x') = -\lambda(x - x')$ is easy to compute with minimal computation overhead. The sampling algorithm is summarized in algorithm 1. Our choice of scaling the guidance by $\sqrt{1 - \bar{\alpha}_{t-1}}$ is to limit the KL divergence between the training and inference distribution across timesteps. More details can be found in the appendix.

Coupled sampling for multi-view editing. Given a collection of N posed multi-view images $\{I_k, P_k\}_{k=1}^N$, we condition the 2D editing model on each image independently I_k. On the other hand, we condition the multi-view model on a single edited image $I_{1,\text{edited}}$ and on the poses of the remaining views $\{P_k\}_{k=2}^N$ to synthesize as novel views. We perform the coupling between the 2D latent conditioned on image I_k, and the multi-view latent associated with its corresponding pose P_k. In fig. 7.3, we illustrate our sampling process and show how the samples of each model change with coupling. We set the coupled multi-view sample as our final output, since we want it to lie within the multi-view distribution.

7.4 Experiments

To demonstrate the versatility of our method, we select tasks that highlight various editing aspects. 1) *Spatial editing*: We use Magic Fixup [34] to highlight the ability of making geometric changes in a scene. 2) *Stylization*: We perform stylization using Control-Net [313] with edge control, demonstrating how we can alter the general appearance of the input while preserving its overall shape. 3) *Relighting*: We perform relighting using Neural-Gaffer [259], which takes an explicit environment map as input. For each of these tasks, we begin with a collection of input images and additional task-specific conditioning. The 2D model edits each image individually, but cannot enforce consistency

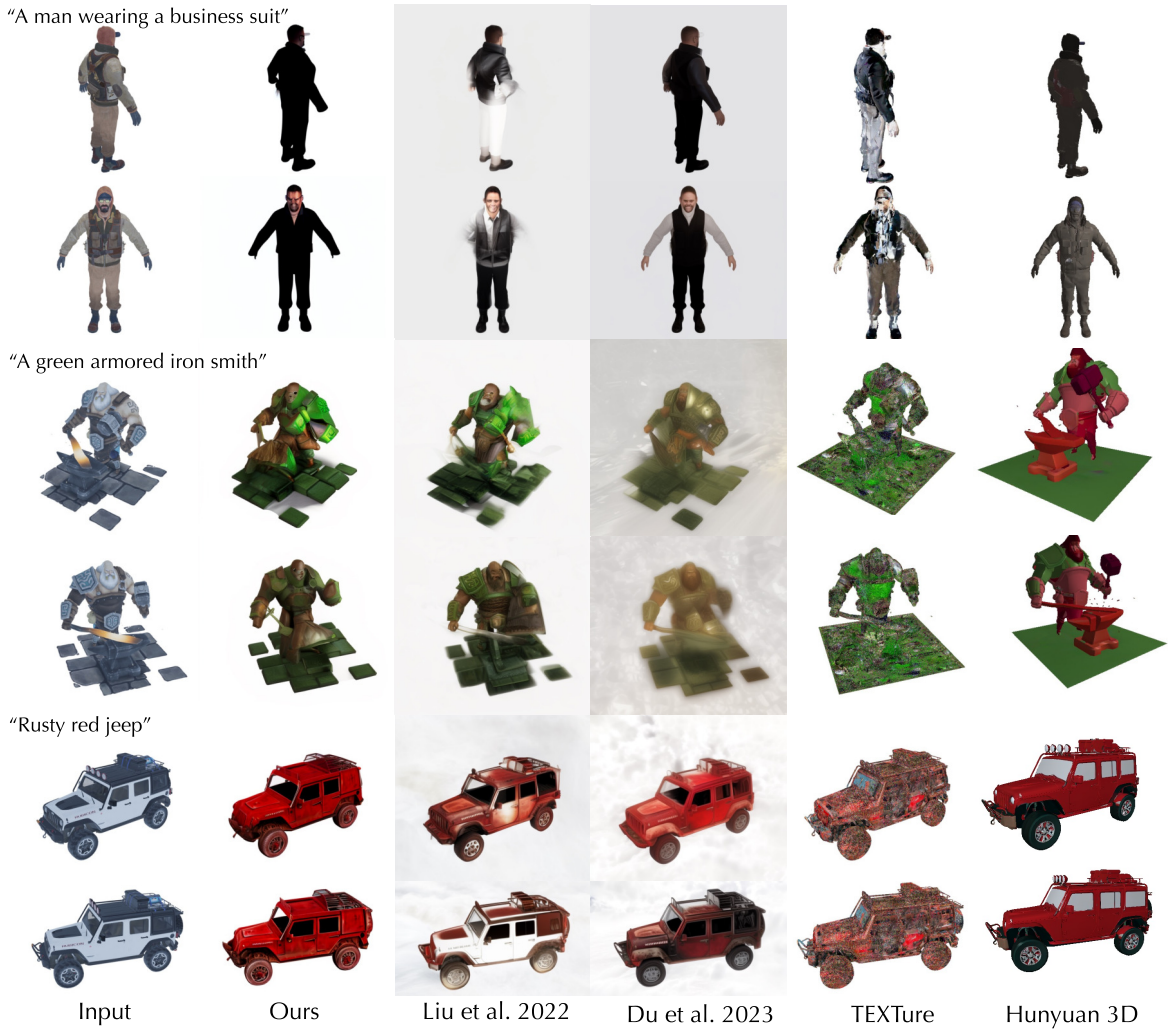


Figure 7.4: **Multi-view stylization.** We show three examples of multi-view stylization of our method against the baselines. Prior work on combining diffusion models [25, 26] suffer from inconsistencies across frames. When provided with a GT mesh, SDS based methods [27] suffer from severe artifacts while Hunyuan 3D's results follow the prompt loosely when doing retexturing.

across multiple images. In contrast, the multi-view model [317] is a novel view synthesis model that takes a set of consistent images and generates novel views. Our pipeline is first edits a single image using the 2D model and then uses it as a reference for the multi-view model. Then, to enforce faithfulness to the remaining input views, we couple the two models, enabling the multi-view model to maintain the input identity while ensuring consistency across views. We perform the coupling in the latent space, and both the 2D models and the multi-view model in this section operate in the latent

Table 7.1: Quantitative comparison on spatial editing. We evaluate against GT renders of the target, and use MEt3r for consistency.

	Per-image metrics			MV metric	
Method	PSNR ↑	SSIM ↑	LPIPS ↓	MEt3r ↓	Users ↑
Per-image [34]	16.5	0.550	0.253	0.353	-
Image-to-MV [317]	12.84	0.400	0.556	0.417	-
Liu *et al*. [25]	16.5	0.530	**0.354**	0.368	9%
Du *et al*. [26]	16.7	0.548	0.411	0.344	1%
SDEdit [19]	15.4	0.458	0.468	0.393	11%
Coupled Sampling (Ours)	**17.0**	**0.550**	0.421	**0.335**	**80%**

space of Stable Diffusion 2.1[347].

Baseline selection. For each task, we adopt Liu *et al*. [25] and Du *et al*. [26] as general-purpose baselines, as they allow composing two diffusion models, and can be directly compared against our framework. We further adopt strong task-specific baselines tailored to each scenario. We also compare against baselines of using the 2D editing model per image and of editing a single image, followed by image-to-MV generation.

User study. To provide a comprehensive evaluation, we conduct user studies using Prolific with 25 participants for each task, comparing our approach to all baselines using best-of-n preference questions. In each task, we conduct 9 comparisons with different scenes and edits.

7.4.1 Multi-view spatial editing

Spatial editing is challenging because it requires accurately harmonizing the scene, including object interactions and changes in shadows and reflections resulting from edits. No large-scale datasets are available for training spatial editing models. As a result, previous work on 2D spatial editing has relied on large-scale video datasets [310, 349, 34] to learn natural object motion. However, such data

Table 7.2: Quantitative comparison on stylization. We evaluate the temporal and subject consistency, and MEt3r score for geometric consistency. CLIP score is computed against the edit prompt. The best metrics across methods equivalent to ours are bolded.

	Per-img metric	MV metrics				
Method	CLIP ↑	Temporal ↑	Subject. ↑	MEt3r ↓	Users ↑	Mesh-free
Per-image [313]	30.0	0.922	0.740	0.546	-	✓
Image-to-MV [317]	29.5	0.927	0.787	0.382	-	✓
TEXTure [27]	28.4	0.967	0.748	0.426	14%	X
Hunyuan3D [348]	29.9	0.952	0.754	0.391	8%	X
Liu et al. [25]	30.1	0.934	0.759	0.461	19%	✓
Du et al. [26]	**30.2**	0.926	0.762	0.461	12%	✓
Coupled Sampling (Ours)	29.68	**0.946**	**0.807**	**0.392**	**47%**	✓

sources do not exist for multi-view datasets, as 4D datasets are extremely scarce and typically created only for evaluation. Our coupled sampling paradigm addresses this gap.

We use Magic Fixup [34] for the 2D editing model. This model takes the original image and a coarse edit that specifies the desired spatial changes. For multi-view editing, it is necessary to apply the edit consistently across all views. In our experiments, we unproject the target object in each image using a depth map. We then apply a 3D transformation to the object and reproject it into the image. As a baseline, we follow the proposed baseline in Magic Fixup, using SDEdit [19], which allows coarse edit inputs. Figure 7.5 presents three different coarse edits, with two frames from each edit shown to illustrate consistency. In the first example, we find that our method correctly translates and rotates the car while preserving the identity of the input. By contrast, the baselines struggle to maintain the back view of the scene. In the final edit, our method produces smooth shadows that match the ground truth, whereas the baseline results in highly irregular shadows.

To quantitatively evaluate performance, we construct three synthetic scenes with a total of 10 spatial edits and render the ground-truth 3D transformations in Blender. Note that while many baselines exist for per-image spatial editing [327, 349, 21], we use Magic Fixup as the per-image baseline since it is the base model we use for coupling. We use standard reconstruction metrics, and MEt3r



Figure 7.5: **Qualitative comparison on multi-view spatial editing.** The baselines struggle in preserving the identity of the input, and produce flickering artifacts across edited frames, while our results achieve both editing targets and multi-view consistency.

[350], which measures the 3D consistency of multi-view outputs. Table 7.1 demonstrates that our method achieves higher PSNR and SSIM scores, along with superior multi-view consistency.

7.4.2 Multi-view Stylization

Stylization is a common application of diffusion models: an input sequence provides the desired spatial structure, and a text prompt specifies the style. Control-Net [313] enables this type of stylization

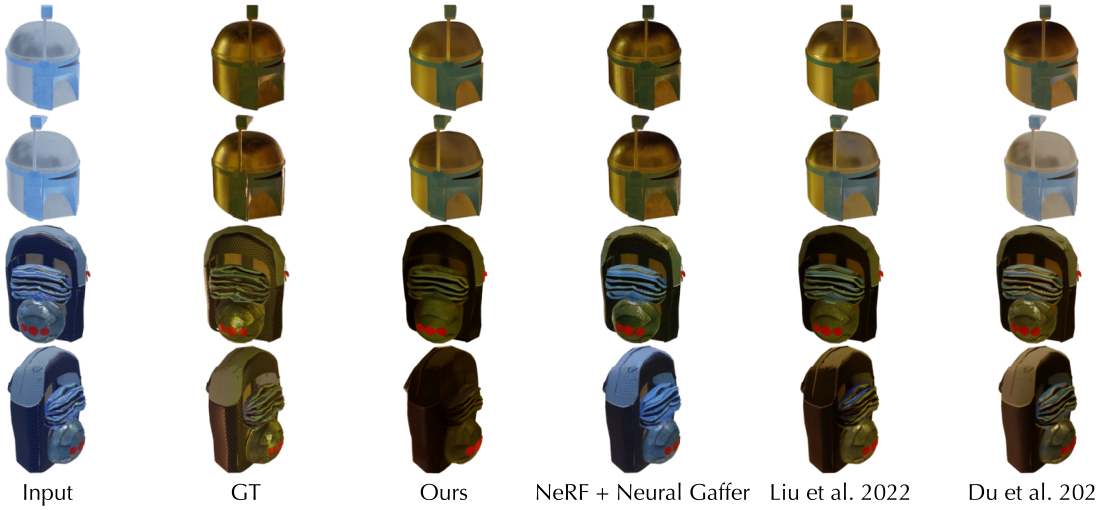


Figure 7.6: **Qualitative comparison on environment map based relighting.** Other methods tend to produce flickering artifacts (notice the change in color in the first two rows for [25, 26]), and using NeRF makes the lighting changes to be baked into the view dependent effects. Our method achieves the best overall result.

by incorporating geometry-related conditioning, such as the Canny edges of an image. A closely related task is 3D re-texturing, in which a 3D mesh is given, and a new texture is generated using a generative model. To assess our method, we rendered ten different objects and applied stylization to each using user-defined prompts. While many stylization baselines exist [351, 337, 352, 334], we select two strong baselines that operate directly on ground truth 3D mesh, such as TEXTure [27], which synthesizes new textures using SDS [314], and Hunyuan3D [348], which employs a SoTA feed-forward multi-view model to generate textures. We omit InstructNeRF2NeRF as it fails on our inputs. Note that both baselines access the GT mesh, a highly privileged input that our method does not have access to. In fig. 7.4, we present results from three representative examples. In the first example, score-averaging methods have difficulty preserving the identity of the edited subject, leading to color changes or changes in identity across frames. In contrast, TEXTure exhibits severe artifacts due to its SDS-based approach. Hunyuan3D produces very simple edits that often do not align with the text prompt. Prior work on composing diffusion models [25, 26] relies on the average scores of the two models, which pushes the samples off the multi-view manifold and causes flickering. On the other hand, our approach relies on softly steering the multi-view model to remain within the distribution and

Table 7.3: Quantitative comparison on relighting. We evaluate against GT relighting results in terms of per-image metrics, and evaluate multi-view consistency with MEt3r.

	Per-image metrics			MV metric	
Method	PSNR ↑	SSIM ↑	LPIPS ↓	MEt3r ↓	Users ↑
Per-image [259]	22.7	0.862	0.159	0.243	-
Image-to-MV [317]	19.3	0.815	0.193	0.229	-
Liu *et al*. [25]	**23.2**	**0.871**	**0.152**	0.220	10%
Du *et al*. [26]	22.1	0.863	0.158	**0.217**	19%
GT NeRF + Neural Gaffer [259]	22.4	0.865	0.162	**0.217**	25%
Coupled Sampling (Ours)	**23.2**	0.868	0.157	**0.217**	**46%**

produce consistent results.

Although the quantitative evaluation of stylization remains challenging, we assess temporal and subject consistency in our generated videos using VBench [353] and measure geometric consistency with MEt3r [350], as shown in Table 7.2. Our results show that our method achieves superior temporal and subject consistency compared to previous approaches for combining diffusion models. For reference, we also report results from mesh-based methods on rendered videos, which are inherently temporally consistent due to the underlying mesh representation.

7.4.3 Multi-view Relighting

When the variance of the 2D diffusion results is low, meaning the sampling distribution is narrow, radiance fields can effectively regularize inconsistencies. However, this requires obtaining a consistent geometry beforehand. As an alternative, we demonstrate that a multi-view diffusion model can regularize inconsistencies in 2D relighting through coupled sampling. Figure 7.6 presents two relighting examples to illustrate this. We observe that prior methods for combining diffusion models [25, 26] can introduce flickering artifacts, as evidenced by abrupt color changes in the top two rows. In contrast, NeRF-based approaches (which we provide with privileged GT NeRF) may incorrectly attribute



Figure 7.7: **Image space coupling.** Using Flux, we perform coupled sampling on different prompts. We show that the coupled samples are spatially aligned while being faithful to the prompt.



Figure 7.8: **Video space coupling.** Using Wan [28], we perform coupled sampling on videos conditioned on different prompts, and show that the coupled samples are highly aligned in each frame while being faithful to the prompt. *Refer to the project webpage for full videos.*

lighting variance to view-dependent effects, as illustrated in the bottom two rows of the backpack example. To quantitatively compare these methods, we use the 3D objects from Neural-Gaffer [259] and add both a diffuse and a glossy object, resulting in a total of 7 objects with 5 relightings each, for 35 comparisons in total. We compute per-image reconstruction metrics and geometric consistency using MEt3r, as shown in Table 7.3. Although these metrics do not capture subtle lighting flicker, our method achieves competitive results in both reconstruction and consistency. Importantly, we also report metrics for relighting each image individually, which serves as a coarse upper bound, and observe no degradation in performance.

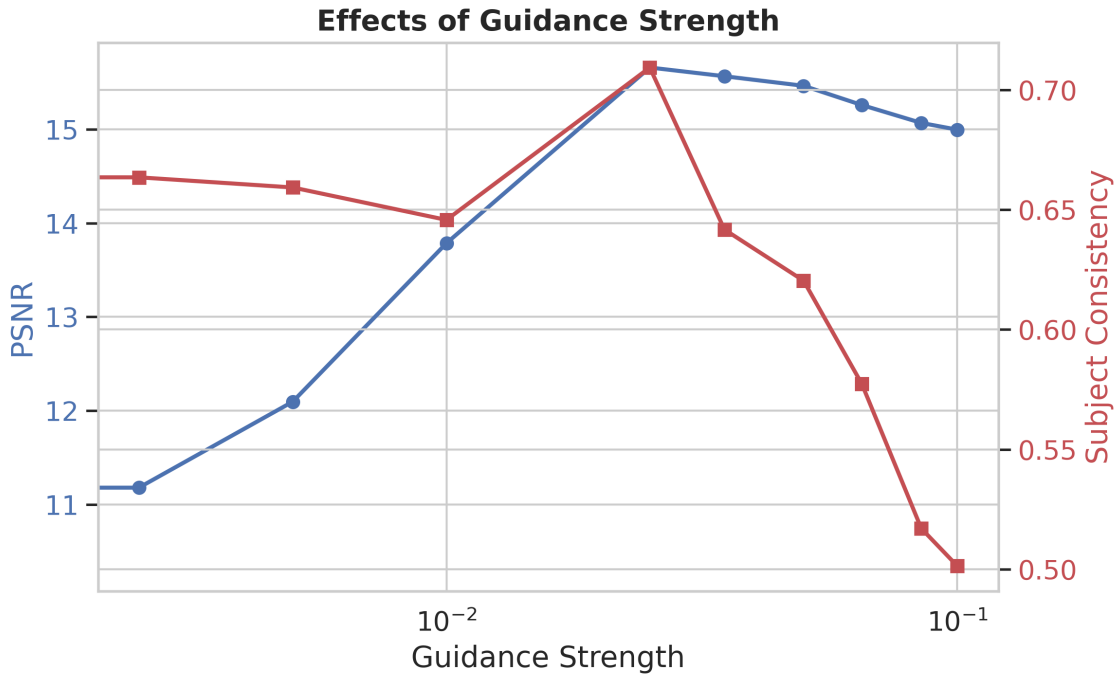


Figure 7.9: **Guidance strength analysis.** As the guidance strength increases, the reconstruction improves but the consistency drops.

7.5 Analysis Experiments

In this section, we demonstrate the effects of coupled sampling beyond multi-view models, and analyze the effects of varying the coupling strength. In the appendix, we include experiments on additional backbone variations, and coupling across different latent spaces.

Coupling Text-to-image and video flow models. To illustrate the effects of coupled sampling beyond multi-view models, we implement our method using the text-to-image model Flux [354]. Although Flux is a flow-based model [355, 356], we show that our coupling approach remains effective. We test coupled sampling by generating two samples from the same model, each conditioned on a different prompt. As shown in fig. 7.7, without coupling, the outputs are typically very distinct. With the coupled sampling, the outputs become spatially aligned while still reflecting their respective prompts. We further illustrate the effects of coupling with the video model Wan2.1 [28] in fig. 7.8 and show the

close alignment of coupled samples.

Guidance strength analysis. In fig. 7.9 we quantitatively evaluate the effects of guidance strength λ on spatial editing performance. When λ is very small, the model output resembles image-to-MV sampling, resulting in low reconstruction performance, and as we increase λ, the reconstruction improves. However, when making the coupling strength too large, consistency across frames degrades and the performance eventually collapses. We highlight that there is a regime for the guidance strength, where both the reconstruction and consistency produce optimal results, emphasizing the value of coupled sampling.

7.6 Discussion and Conclusion

We introduce a simple and effective approach for coupling diffusion models, enabling 2D diffusion models to generate consistent multi-view edits when used with multi-view diffusion models. Our method is efficient, versatile, and achieves high-quality results. By guiding the diffusion sampling process, our approach produces outputs that retain the strengths of the underlying models, while also naturally inheriting their limitations.

We believe this coupling strategy has potential applications beyond multi-view editing. In the future, our paradigm could extend the capabilities of image-editing models to video editing by integrating with video diffusion models, without incurring additional computational overhead.

Chapter 8: Masked Visual Actions for Unified World Modeling

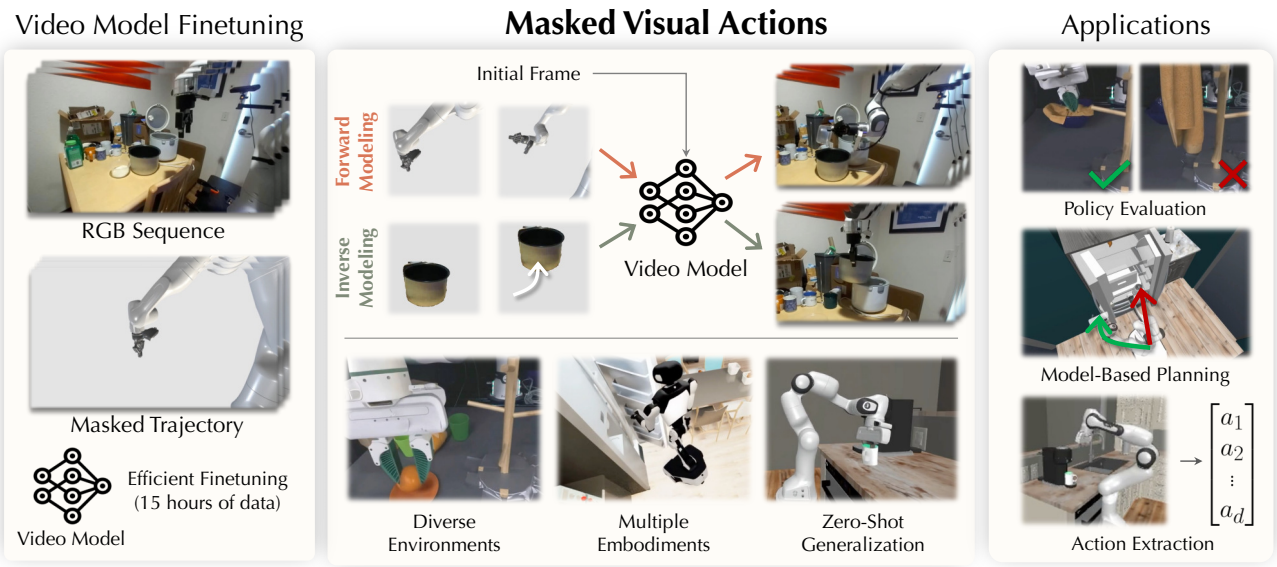


Figure 8.1: **Masked Visual Actions.** We finetune a video model to condition on masked trajectories of robots, representing robot actions as pixel-space masked motions. Efficiently finetuned on only 15 hours of data, a single checkpoint of the model can act as an action-conditioned forward model to simulate robotic interactions with diverse and unseen embodiments. By conditioning it on object motion, it can also act as an inverse model that synthesizes the robot motion needed to achieve the desired outcome. We showcase the efficacy of our model for policy evaluation, model-based-planning, and action extraction where the video model acts as a policy.

8.1 Introduction

Purposeful interaction requires connecting an agent's actions to their effects in the world, and vice versa. In sensorimotor control, skilled behavior is often described as coupling a forward model that anticipates the sensory consequences of movement with an inverse model that recovers the move-

ment needed to realize a desired state [357–359]. Robotic world models should similarly support both directions of reasoning in a single predictive framework.

Recent advances in video models offer a promising route to this ambition. Trained on large-scale observation, they accumulate remarkably broad priors over motion, contact, persistence, deformation, and change, far beyond what can usually be distilled from robot data alone. However, most still remain passive observers rather than tools for intervention. Existing models condition generation on text [29, 360], tracks [361, 30, 362–364], forces [365–367], keypoints [368, 369], or motor commands [370, 1], signals that are often sparse, embodiment-specific, or misaligned with the model's pre-trained visual experience. What remains missing is an action representation expressed directly in the visual space where pretrained video models learned their interaction priors. Once action is expressed *visually*, the same model can complete different parts of an interaction depending on the trajectory revealed: revealing robot motion prompts a scene response, while revealing object motion prompts robot behavior.

To realize this vision, we introduce **Masked Visual Actions**, a method that recasts action as a visual primitive directly within the pre-trained model's native representation. We finetune a pre-trained video model [29] to ingest actions as partially revealed spatiotemporal patterns in pixel space—a masked trajectory of an entity in the scene. When the revealed entity is the robot, the model predicts the scene's response and acts as a forward dynamics model; when the revealed entity is instead an object or desired object motion, the same model acts as an inverse model to recover robot behavior consistent with that outcome. In this view, active and passive roles are not properties of separate architectures, but different queries to the same interaction prior. While conceptually simple, this interface is pixel-aligned, embodiment-agnostic, native to video, and efficient to inject into a pretrained model through lightweight adaptation.

In addition to superior visual fidelity compared to prior work, we validate our framework across three applications in robot manipulation, both in simulation and in the real world: policy evaluation, model-based planning, and inverse modeling, where the video model is used as part of a robot policy. All experiments use a *single* checkpoint of the model finetuned on as few as 15 hours of robot interaction data. In policy evaluation, the model's imagined rollouts exhibit consistent correlation with real-world outcomes, so that simulated performance serves as a useful proxy for actual execution. In model-based planning, these same predictive capabilities are used to simulate the effects of different action trajectories and select the best one for execution, leading to consistent gains across diverse tasks and policy architectures. The same checkpoint can also be used in reverse: given desired object motion, it synthesizes robot motion that achieves the goal, and a learned inverse dynamics model extracts the resulting actions.

We summarize our contributions as follows: **(1)** we introduce Masked Visual Actions, a pixel-space control interface for pretrained video models, together with an efficient adaptation recipe based on masked examples from real and simulated data; **(2)** we show that forward and inverse robot world-modeling problems can be cast as complementary conditional prediction problems of the same video model, obtained by revealing different entities in the scene; and **(3)** we validate this framework in both simulation and the real world across three applications in robot manipulation: policy evaluation, model-based planning, and inverse modeling.

8.2 Related Work

Controllable video generation as a robotic interface. Video models become simulators once a control signal is added: physical forces [365, 366], warped flow [371], hand poses [368, 372, 373],



Figure 8.2: **Comparing action representations for learning.** Low-dimensional robot actions are compact, but embodiment-specific and not aligned with the image observations used by video models. End-effector poses or robot skeletons are more visual, but remain sparse and require the model to infer geometry, contact, and interaction effects. Our masked visual actions provide dense, image-aligned conditioning, making robot motion and action directly visible, yielding a more learnable representation across embodiments and object interaction.

point or trajectory tracks [362, 361, 30, 363], and goal images [374]. None of these signals is dense, pixel-aligned, or shareable across embodiments. Closer to us, visual prompting via inpainting [375] and world modeling as conditional inference [376] show that varying which region of an image is provided enables a generic task parametrization in visual domains; we extend this idea to robotic world modeling by using masked frames as the control interface.

Pixel-grounded action conditioning for robot world models. Most robotic video world models communicate actions through embodiment-specific channels: end-effector poses [377, 364, 1, 378], joint vectors [370, 1], or skeletons [369, 379]. A growing line replaces them with pixel-grounded signals. BridgeV2W [380] and Kinema4D [381] render the robot through its URDF and inject the resulting masks or pointmaps via ControlNet; Action Images [382] encodes 7-DoF actions as multi-view Gaussian heatmaps; ORV [383] conditions on 4D occupancy; Mask2IV [384] conditions on predicted mask trajectories; Mask World Model [385] predicts semantic masks as the output. A complementary line treats masks as a data-editing tool: Shadow [386, 387], Phantom [388], Masquerade [389], and EmbodiSwap [390] composite or render the robot onto human videos for cross-embodiment policy transfer. All of these works treat the robot as the active entity during training and run forward only. We expose the same masking interface to *any* subset of entities, so one model serves as a forward,

inverse, or unconditional generator without retraining.

Unified video-action models and downstream uses. UVA [391], UWM [392], AIM [393], X-WAM [394], and MotuBrain [395] unify forward dynamics, inverse dynamics, policy, and video generation in one model by masking modality channels (action vector vs. video) or by manipulating diffusion timesteps; large platforms such as Cosmos [360], Genie Envisioner [396], and DreamGen [397] package similar capabilities at foundation-model scale. Because the masking is over modality channels, actions remain low-dimensional vectors and the unification does not transfer across embodiments. Our masking is *spatial*: active and passive entities live on the same pixel canvas, so the same forward/inverse switch also bridges the embodiment gap. The forward direction is exercised for policy evaluation [398, 379, 399], policy improvement [1], planning [400–405], and direct video-as-policy [406, 407]; the inverse direction extracts robot motion through point tracking [408, 409], object flow [410–412], predicted object pose [413], or learned IDMs [400, 401, 414–417]. Critically, our inverse pipeline reuses the same backbone as the forward simulator, while prior work trains a separate IDM head or flow predictor.

8.3 Masked Visual Actions

Video generation models can be used to show how an initial scene evolves over time [29, 418], as they can model rich scene dynamics and object interactions. The video model captures the distribution $p(V)$ over videos $V \in \mathbb{R}^{T\times H\times W\times 3}$ depicting a scene. We view a scene $\mathcal{S}$ as a set of entities $e_1, e_2, ..., e_n$, and the video model generates a sequence of frames depicting how the entities interact over time. In the output video, each e_i has a spatiotemporal trajectory, and we abuse notation slightly and write e_i for both the entity and the spatiotemporal region of pixels it occupies. Aligned with recent

works built on structured masking and condition inference [376, 419–421], the video model implicitly captures the joint distribution over all entity trajectories

$$p(V) = p(e_1, e_2, \ldots, e_n), \tag{8.1}$$

including the interactions among them. Conditioning on a subset $\mathcal{S} \subseteq \{1, \ldots, n\}$ of entities yields the conditional distribution

$$p\big(\{e_i\}_{i \notin \mathcal{S}} \,\big|\, \{e_j\}_{j \in \mathcal{S}},\, I_0\big), \tag{8.2}$$

where I_0 is a reference image of the initial scene. By varying $\mathcal{S}$, the same model answers different questions about the same scene.

We realize this conditioning by masking. Let $M \in \{0,1\}^{T \times H \times W}$ be a binary mask indicating which spatiotemporal pixels are revealed to the model ($M_{t,h,w} = 1$) versus predicted ($M_{t,h,w} = 0$). For a chosen conditioning set $\mathcal{S}$, the mask is the union of the pixel regions occupied by the conditioned entities,

$$M(\mathcal{S}) = \bigcup_{i \in \mathcal{S}} e_i, \tag{8.3}$$

and the model receives as input the masked video $M \odot V$ together with the reference image I_0. Training proceeds by sampling M from a distribution over masks and learning the conditional $p_\theta(V \mid M \odot V, I_0)$. We draw inspiration from masked modeling in language [422] and masked-image prompting [375], where varying the masked input enables diverse applications with the same model.

In robotics, it is convenient to partition entities into two roles, as illustrated in Figure 8.3. We call entity e_i *active* if it acts on the scene through its own agency, such as a robot arm or a human, and *passive* if its motion arises from interaction with an active entity, such as a manipulated object.

Let $\mathcal{A} \subseteq \{1, \ldots, n\}$ index the active entities and $\mathcal{P} = \{1, \ldots, n\} \setminus \mathcal{A}$ the passive ones. This partition surfaces two natural ways to use the model.

Forward model. Setting $\mathcal{S} = \mathcal{A}$, we condition on the active entities and predict the passive ones,

$$p\big(\{e_i\}_{i \in \mathcal{P}} \,\big|\, \{e_j\}_{j \in \mathcal{A}},\, I_0\big). \tag{8.4}$$

This corresponds to the standard action-conditioned dynamics modeling, in which a robot's motion is provided, and the model simulates its effect on the scene. Unlike prior work that conditions on low-dimensional action commands [370, 1], the active conditioning here is supplied as masked videos that are agnostic to embodiments.

Inverse model. Setting $\mathcal{S} = \mathcal{P}$, we condition on the passive entities and predict the active ones,

$$p\big(\{e_i\}_{i \in \mathcal{A}} \,\big|\, \{e_j\}_{j \in \mathcal{P}},\, I_0\big). \tag{8.5}$$

This direction has no analog in conventional action-conditioned world models: the user specifies a desired outcome in the world, and the video model recovers the agent behavior consistent with it.

The active/passive distinction is a convenient way to describe how we use the model rather than a property of the model itself. The model is trained on masked video completion without any explicit notion of agency, and at inference, any subset $\mathcal{S}$ can be chosen. In fact, we trained our model only on masks depicting active robotic entities, yet it generalizes to queries conditioned on passive entities in a *zero-shot* manner. As observed in our empirical evaluations, this behavior is unique to our masked visual action conditioning, as conditioning the video model on sparser signals, such as low-level action

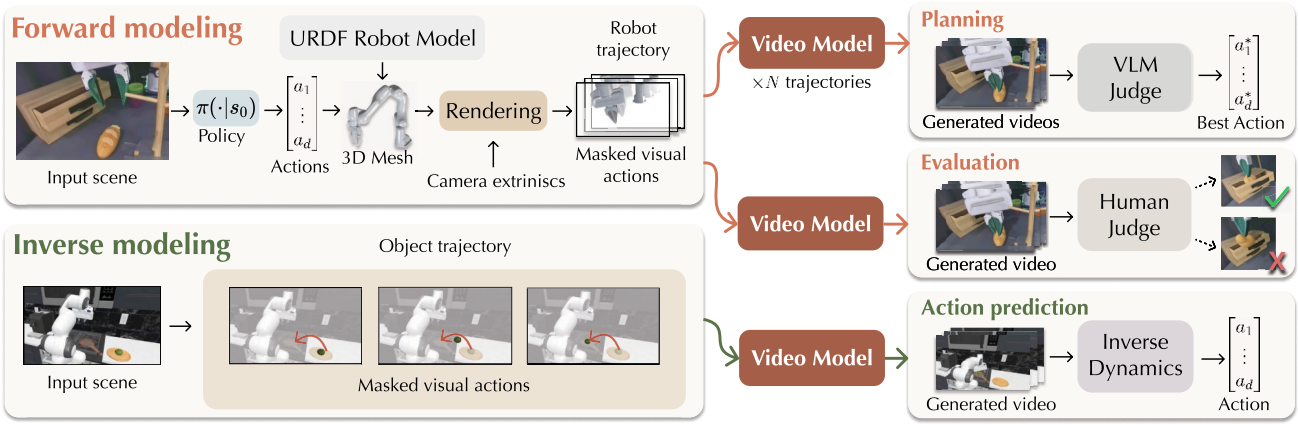


Figure 8.3: **Applications.** The masked visual actions allow using the video model as a forward model, conditioned on robot actions, or as an inverse model that predicts the robot motion that satisfies the object trajectory. The forward model can be used for planning and choosing the best trajectory sampled from a policy, or policy evaluation. On the other hand, the inverse modeling can be combined with an inverse dynamics model to estimate robot actions from the generated video.

commands or visualized skeletons, cannot achieve this level of generalization.

8.4 Method

8.4.1 Dataset construction

We construct the masked modeling dataset by combining real-world videos from DROID [423] and simulation data from Robocasa [424]. We use both success and failure trajectories from both datasets. We follow two approaches to construct masked conditioning for each video, based on video segmentation and rendering the robot state, as outlined below.

Segmentation based dataset Given any video, we can segment any entity in the scene using SegmentAnything [425], without the need for camera calibration or even explicitly knowing which robot is shown in the video. We use videos from DROID, and use the prompt "A robotic arm" for segmentation to isolate the robot. Using segmentation data enables the model to effectively learn to *inpaint* missing regions and to model the joint distribution over all entities in the scene. While a segmentation-based

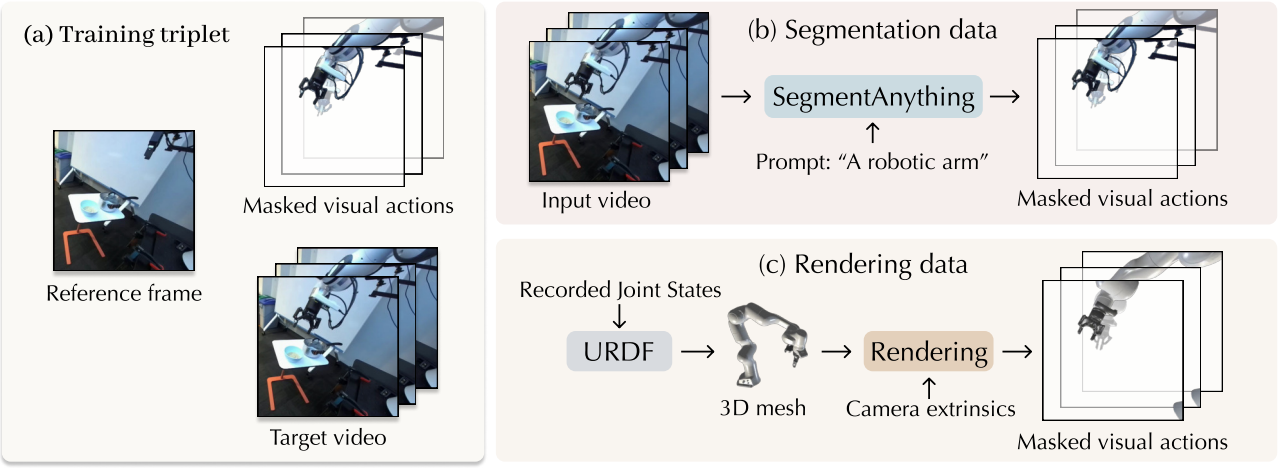


Figure 8.4: **Dataset construction.** (a) to train our model, we need a reference frame of the initial scene, and the masked visual actions, and train it to reproduce a realistic video of the robot executing the input actions. (b) We use segmentation-based approach by segmenting the robot arm from robotics datasets as the masked visual actions. (c) However, to allow the user to provide arbitrary action trajectory at inference, the model needs to also accept simulated mesh visualization of those actions. As a result, we also include rendering-based dataset. Given that DROID also contains the robot state at each timestep, we render the robot URDF that matches the original video to construct masked visual actions.

approach is highly general, it suffers from two major limitations: First, it is challenging for the user to provide an exact segmentation mask of entities at test time. Second, any occluded regions in the robot would implicitly leak information about the scene dynamics from the original video. To mitigate those limitations, we also explore the approach that explicitly renders robots from their recorded state.

Rendering based dataset Instead of relying on segmentation, we can also align a robot mesh with the input video and use that as the masked conditioning. By rendering the robot mesh, we can visualize arbitrary action trajectories during inference and then use them as masked conditioning for the video model. To construct a rendering-based dataset, we require the robot state corresponding to the input video and the camera calibration. We use the DROID dataset and follow the protocols from PointWorld [403] to refine the camera calibration to accurately align the robot URDF with the input trajectories. In the Robocasa simulation, we render only the robot, excluding the rest of the scene, to generate the masked conditioning. To allow the model to see the full robot without self-occlusion,

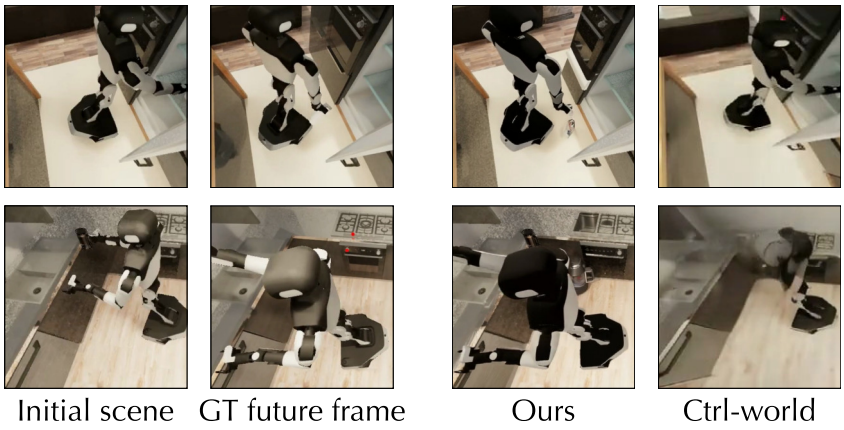


Figure 8.5: **Generalization to unseen embodiment.** While using the raw action state such as in Ctrl-world [1] can work well within the training domain, it collapses on unseen embodiments. However, our method can generalize well to unseen embodiments.

we render the robot only with translucent rendering and set the gripper fingers to bright red so the video model can easily observe the actions. Note that the rendering approach requires known camera calibration, and is limited to rendering the robot only as opposed to arbitrarily enabling masking any entity in the scene. As a result, we believe that the segmentation and rendering-based approaches are complementary.

8.4.2 Model implementation and training

We use Wan-Fun-Control 2.2 14B [29] as the base model. We encode the masked conditioning video using the same autoencoder as the video model and use concatenation as the conditioning mechanism. Concatenation is appropriate as the conditioning signal is spatially aligned with the desired output video. For the missing region from the masked conditioning, we set it to a uniform gray background. Instead of finetuning the entire model, we use LoRA finetuning with rank 256, and a batch size of 4 using 8 NVIDIA H200 GPUs. We train the model for approximately 10,000 steps over 4 days. *For reproducibility, we will release our code, data, and model weights.*

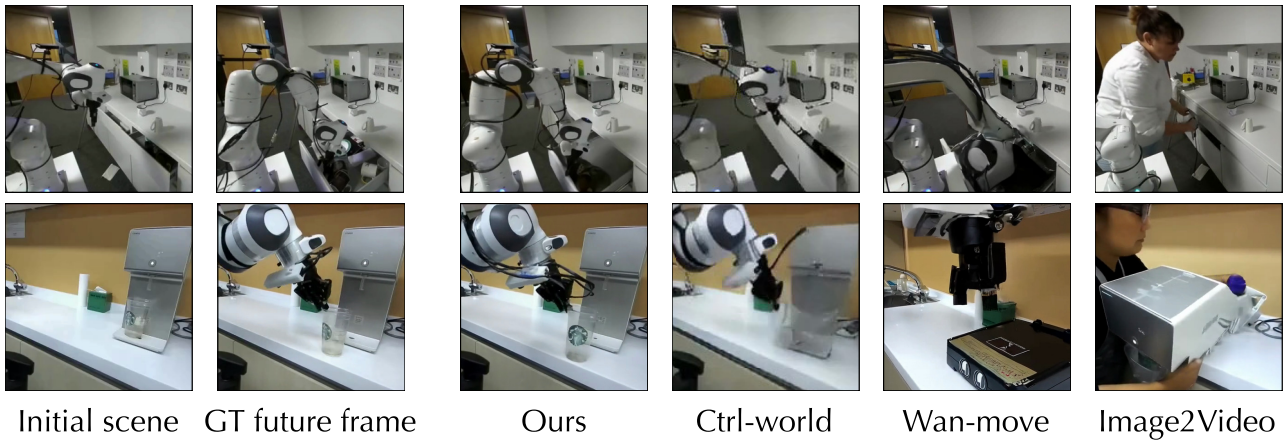


Figure 8.6: **Comparing baselines on DROID.** Using image-to-video [29] or even trajectory conditioned video generation [30] with GT tracks fails to execute the robot motion or preserve the input scene. On the other hand, our model can competitively match and outperform models that take the raw robot actions [1] while maintaining generalization.

8.5 Experiments

We start by evaluating Masked Visual Actions as a control signal for world modeling. We evaluate visual fidelity and controllability against prior work and highlight generalization to embodiments unseen during training. Afterward, we evaluate diverse robotic applications of our video model. In particular, we show how it can be used for planning by evaluating sampled trajectories, for policy evaluation, and for using the video model as the policy itself through inverse modeling.

8.5.1 Controllable video generation

Video generation traditionally conditioned on text is expressive, but underspecified. Conditioning on motion tracks preserves the generality while allowing us to condition on motion. On the other hand, conditioning on the action space for a specific robotic embodiment provides additional precision, but at the cost of generality. Through Masked Visual Actions, we aim to preserve the generality of the video model by visually conditioning the video model on the robotic actions, and setting the role

Table 8.1: **Baseline comparison on diverse embodiments.** We evaluate our method against Ctrl-World [1] on DROID as a seen robotic embodiment, as well as BEHAVIOR, which uses a bimanual robotic embodiment that's unseen for all methods. Our model outperform the baseline on both datasets, and we include image-to-video and trajectory conditioned video models as a reference.

	DROID			BEHAVIOR		
Method	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑
Image-to-video [29]	0.521	0.548	12.42	0.602	0.457	10.22
Wan-move [30]	0.534	0.562	12.99	0.312	0.756	13.17
Ctrl-World [1]	0.362	0.708	18.15	0.196	0.837	18.39
Masked Visual Actions (Ours)	**0.0945**	**0.887**	**23.74**	**0.123**	**0.843**	**22.90**

of the video model to answer: *given this visual masked action, how would the rest of the scene look like?* As a baseline for using the robot's raw end-effector state as input, we use Ctrl-world [1], a recent SoTA method. For track conditioning, we use Wan-move [30], conditioning it on ground-truth tracks computed from the robot mesh. Additionally, we include Wan2.2 14B image-to-video as a reference.

In Fig. 8.6, we highlight that both Masked Visual Actions and Ctrl-world accurately follow the robot actions on our held out scenes from DROID[1] On the other hand, both Wan-Move and Wan I2V completely collapse and transform the input scene. However, unlike our method, conditioning on the raw robot state cannot generalize to unseen embodiments. We use data from BEHAVIOR [426], which uses a bimanual robot, R1-Pro, to evaluate generalization on unseen embodiments. In Fig. 8.5, we demonstrate that Ctrl-world simply outputs static or corrupted videos for unseen embodiments, while our model can gracefully handle the unseen embodiment. We quantitatively evaluate performance on generated videos across both DROID and BEHAVIOR in Table 8.1 and show that our method outperforms the baselines.

Comparing the choice of visual actions Conditioning on visual actions allows diverse ways to represent the action. We compare against visualizing the end-effector pose, inspired by IRASim [364],

[1]Ctrl-World was trained on the entirety of DROID, so it has seen our held out scenes during training.

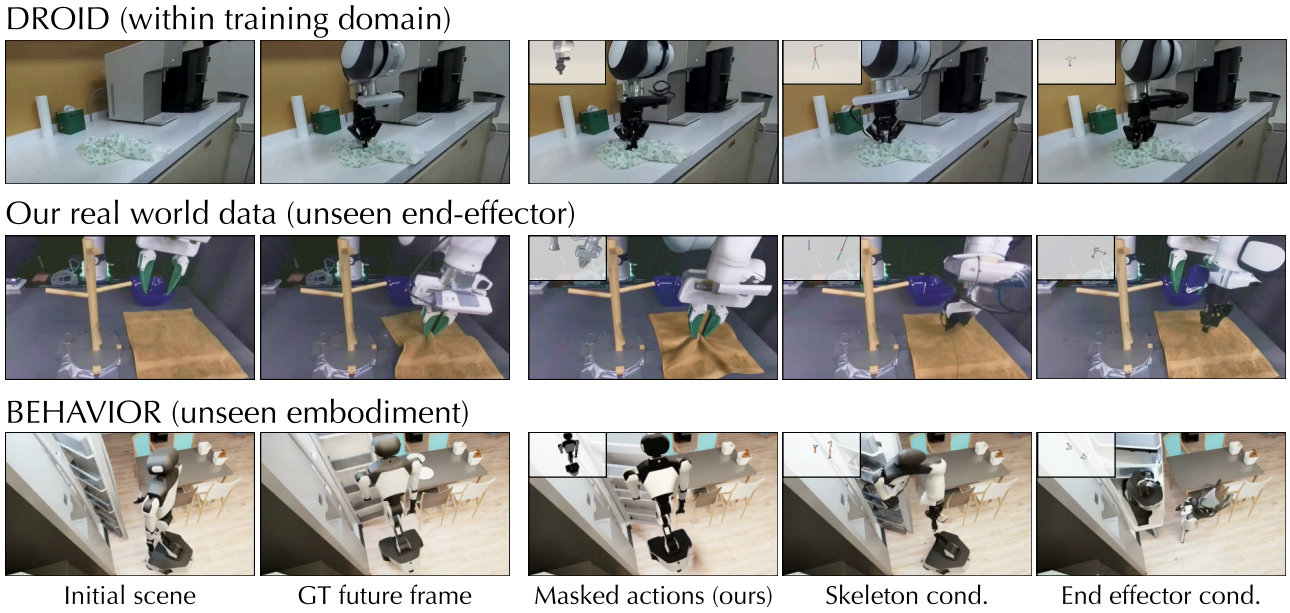


Figure 8.7: **Comparing action conditioning.** Training a video model on different conditioning signals on DROID such as masked visual actions, end effector visualization, or skeleton all work well within the training domain. However, when going beyond the training distribution, such as using a custom end-effector, the models trained on skeleton and end effector position would hallucinate the robot seen in training or transform the robot to match the training. Furthermore, on unseen embodiments such as bimanual robots in BEHAVIOR, using masked actions generalizes gracefully, while other conditioning signals transform and disfigure the robot.

and the robot skeleton, adopted in VAP [369]. We train the same base model used for our method, and use the same training dataset from DROID to train the baselines. While we expect the varying action conditioning to perform similarly on the same domain as the training set, sparse conditioning signals require the model to explicitly learn the correspondence between the sparse action and the target video. However, by conditioning on masked visual actions, the model simply needs to model the interaction between the masked input and the rest of the scene. In Fig. 8.7, we show that on DROID, all the variants of our model perform similarly. However, on real-world data we captured using a similar robot to the one used in DROID, the Franka Emika Panda, but with a custom 3D-printed end-effector, we find that using a sparse conditioning signal suffers significantly. In particular, when conditioning on the robot skeleton, the video model would transform the robot to match the embodiment seen

Table 8.2: **Ablation on visual conditioning signal.** When using sparse conditioning signal, the performance on held out data from the training distribution on DROID is similar to using masked visual actions. However, when using the same robot with unseen gripper (such as on our real world data), or on a robot from unseen embodiment in BEHAVIOR, the gap increases significantly between our masked actions and the other conditioning methods.

	DROID			Real world			BEHAVIOR		
Method	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑	LPIPS ↓	SSIM ↑	PSNR ↑
End-effector vis.	0.107	0.878	22.64	0.183	0.858	20.32	0.171	0.815	19.23
Skeleton vis.	0.106	0.878	22.74	0.169	**0.866**	21.02	0.162	0.824	19.58
Masked Visual Actions	**0.0945**	**0.887**	**23.74**	**0.148**	0.864	**22.79**	**0.123**	**0.843**	**22.90**

during training. When using the end-effector visualization as input, the model would simply introduce another robot into the scene that matches the training data. To further deviate from the training setting, we test the models on the R1 Pro in BEHAVIOR [426]. Given that R1 Pro has two end effectors, we adapt the baseline visualizations to show two end effectors and the skeleton poses of each. We find that conditioning on the end-effector or skeleton visualization completely collapses and distorts the robot. However, when using masked visual actions, the model can gracefully generalize and simulate the physical interaction of the robot opening the fridge. In Table 8.2, we quantitatively evaluate the video generation performance for masked visual actions, and the sparser conditioning mechanisms of end effector pose visualization and robot skeleton.

8.5.2 Robotics applications

We highlight multiple applications of our unified world model in robotics. We use our model as a forward model to simulate robot actions and demonstrate its use for planning and policy evaluation. We also use our model as an inverse model: given the desired object motion as a masked visual action, we generate a video of the robot performing the desired object manipulation and extract the actions using a learned inverse dynamics model. Across these applications, we use Robocasa [424] as the

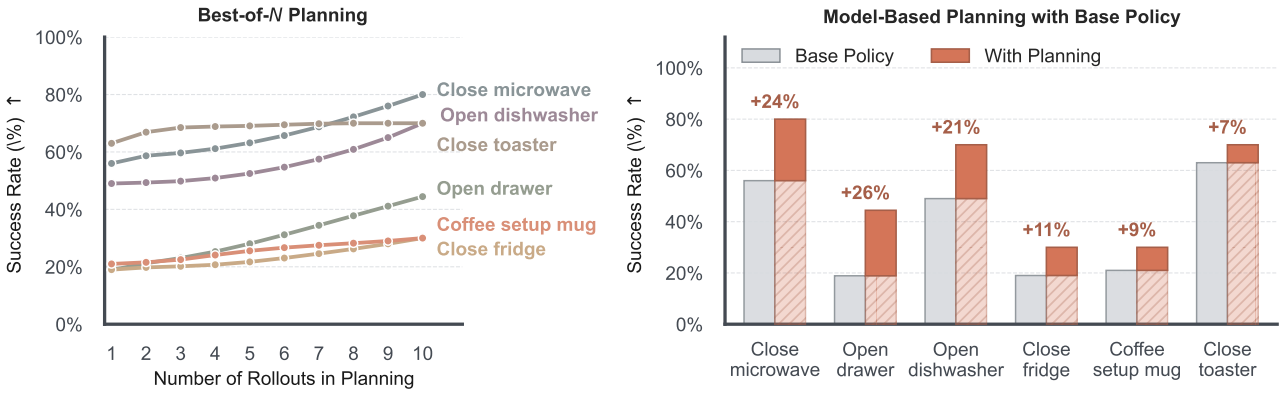


Figure 8.8: **Application on Planning.** By rolling out multiple trajectories from a pretrained diffusion policy, we can evaluate each trajectory by simulating the actions with the video model, and then using a VLM judge to pick the best action trajectory. We observe consistent improvement in task success when using the video model to roll out and choose best action sequences, as well as the positive correlation with the number of samples evaluated at test time. This demonstrates the ability of the model to simulate counterfactuals given the same initial condition.

simulation environment.

Planning Given the same environment observation, multiple rollouts sampled from a stochastic policy may achieve varying levels of task progress. By rolling them out in an action-conditioned video model, one may evaluate the trajectories purely in imagination before executing them in the actual environment. In our experiments, we use Diffusion Policy [427, 428] as the stochastic policy and Best-of-N as the simplest model-based planning algorithm. After simulating the action candidates using the video model, we evaluate each rollout with Gemini 3.1 Pro to assess their relative task success, interaction fidelity, and physical realism. We evaluate on 10 scenes per task, with $N = 10$. We include the detailed criteria in the appendix. After evaluating all the rollouts, we pick the best action sequence to execute. In Fig. 8.8, we highlight the improvement in performance on diverse tasks and show how success rate increases with the number of action samples. This approach can be viewed as a form of test-time scaling [429, 430], leveraging additional compute to achieve higher performance. In our case, the policy and video model act as the generator, and the VLM critic acts as the verifier.

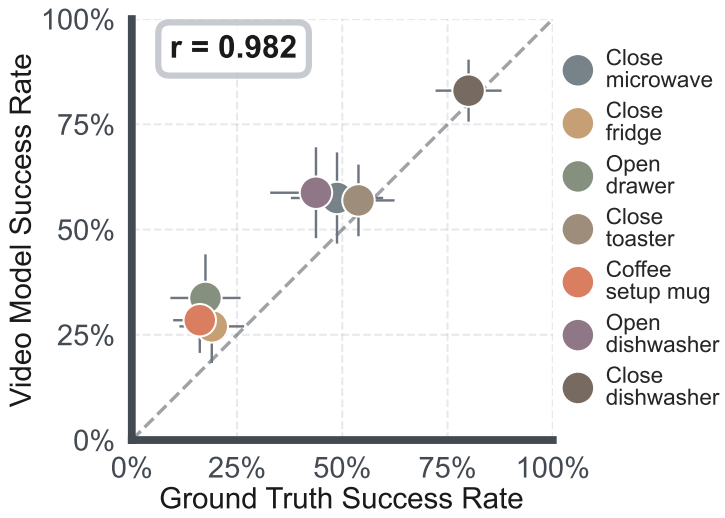


Figure 8.9: **Policy evaluation.** Video-model rollouts consistently track ground-truth success rates across RoboCasa tasks.

Policy evaluation We can also use our model to evaluate policy performance by comparing a policy's success rate within the video model to that within the ground-truth environment. Similarly to the section above, we use an open-loop diffusion policy across tasks and rollout 10 trajectories per scene. We simulate action trajectories using our model and manually evaluate each rollout as a success or failure based on predefined task rubrics. The simulated rollouts are additionally evaluated by physical interaction realism (e.g., hallucinated task progress without contact is considered failure). In Fig. 8.9, we plot the success rate of each policy in GT environment against that evaluated within the video model, which exhibits a strong correlation with $r = 0.982$. However, we observe that the video model shows a positive bias towards task progress, as evidenced by consistently higher task success rates in its imagination. In the supplementary material, we include additional real-world results demonstrating the correlation between the task success and the video model output.

Action extraction Instead of providing the model with robot actions to execute, we can alternatively leverage Masked Visual Actions to formulate an inverse modeling problem: given desired object mo-

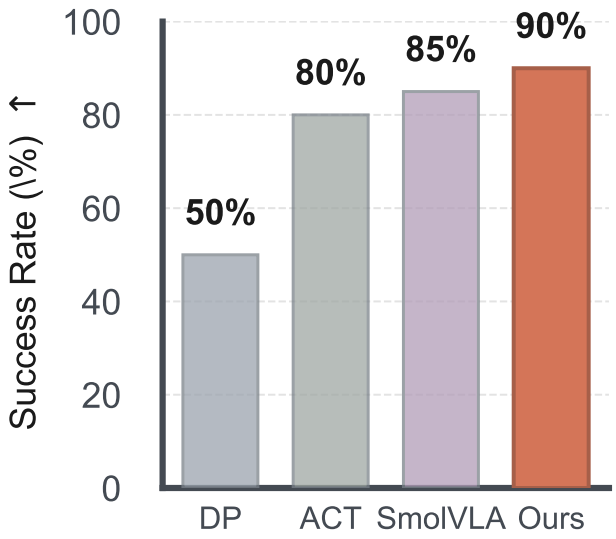


Figure 8.10: **Action extraction.** Even without task-specific video-model training, inverse modeling recovers competitive robot behavior.

tion, prompting the video model to synthesize a video of a robot achieving that outcome. Action extraction can then be cast as an inverse-dynamics problem: given the synthesized robot video, recover an executable low-level action sequence with a learned inverse-dynamics model. We initially expected this setting to require explicit inverse-modeling finetuning. Instead, the video model trained only on forward examples already generalizes zero-shot to the inverse setting, likely because the conditioning signal is well-aligned with the model's learned representation. We evaluate this pipeline on CoffeeServeMug in RoboCasa, where the robot must reach, grasp, transport, and place a mug from the coffee machine onto the table in a tightly constrained workspace. We compare to standard imitation learning baselines, including Diffusion Policy [427, 428], ACT [431], and SmolVLA [432]. The inverse-dynamics model and all baselines are trained on 100 demonstrations, whereas the video model itself has not seen examples from this task. Each method is evaluated with 20 trials, with success rates reported in Figure 8.10. Our method achieves the highest success rate at 90%. This indicates that, although the video model has not been trained on an inverse modeling problem, it can be ef-

fectively prompted to extract robot behaviors from its rich interaction priors, while maintaining the competitiveness of modern imitation learning methods.

8.6 Discussion and Conclusions

By finetuning a pretrained video model on a small amount of Masked Visual Actions data, we efficiently leverage the prior of the video model to synthesize counterfactuals by conditioning on a subset of scene entities. Our model can simulate robot actions when conditioned on robotic embodiment visualization as an action-conditioned forward model, and when acting as an inverse model, where it synthesizes suitable robot motion to realistically manipulate the object.

Limitations It is worth noting that our model, similarly to existing generative models, learns the *correlation* between object interaction rather than *causal relationships*, which remains an open research question. Furthermore, our method is naturally limited by the base video model's capabilities, in terms of both inference speed and what it can express, as it re-purposes the model's prior rather than modifying its capabilities.

Chapter 9: Conclusion

In this dissertation, we studied how structural and learned priors can extract the signals hidden in visual data, whether for reconstruction, editing, or refocusing. We started our discussion with applications that use per-instance optimization (like the 3D scene reconstruction from eye reflections), and then gradually moved to training generative models on large datasets (like learning how to perform spatial editing with Magic Fixup, or simulating robot actions with masked visual actions). The initial part of the thesis directly extracts information from the user input (portrait photos, videos with subtle motion, or dual-camera smartphone output) and uses it to uncover information about the world (3D structure, or hidden motion) or provide editing control (adjusting the defocus post-capture). Afterwards, using generative models enabled us to apply priors learned from large-scale datasets to help manipulate the user input realistically, in spatial editing, relighting, and stylization, or in simulating physical robotic interactions. This is in parallel with the direction of the computer vision field as a whole, with a shift that is moving from specialized systems to general large models that are more versatile.

While computer vision focuses on visual data, humans use more than vision to interact with the physical world. Analogously, while world modeling in visual space is powerful, it is only part of the picture. Questions like how much force the robot gripper is exerting, or how heavy the object is that we are about to lift, require incorporating force and tactile information as another modality. One question is how to parameterize force information in a way that world models can use. Furthermore, tactile information is data-scarce compared to videos, and thus we need a data-efficient approach to incorporating it into the model. Both finding the proper data parameterization and learning efficiently from scarce data recurred throughout the thesis, and will only grow more important as we continue incorporating data-scarce modalities into our large models. While extracting hidden signals from

visual data allows us to understand and edit the visual world, it is time to look beyond visual data.

Bibliography

[1] Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. In *The Fourteenth International Conference on Learning Representations*, 2026.

[2] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[3] Rakshit S Kothari, Aayush K Chaudhary, Reynold J Bailey, Jeff B Pelz, and Gabriel J Diaz. Ellseg: An ellipse segmentation framework for robust gaze tracking. *IEEE Transactions on Visualization and Computer Graphics*, 2020.

[4] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollar, and Ross Girshick. Segment Anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[5] Neal Wadhwa, Michael Rubinstein, Frédo Durand, and William T. Freeman. Riesz Pyramids for Fast Phase-based Video Magnification. *ICCP*, 2014.

[6] Neal Wadhwa, Michael Rubinstein, Frédo Durand, and William T. Freeman. Phase-Based Video Motion Processing. *ACM Transactions on Graphics (Proceedings SIGGRAPH 2013)*, 32(4), 2013.

[7] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. NeuMan: Neural Human Radiance Field from a Single Video. *ECCV*, 2022.

[8] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, 2018.

[9] Neal Wadhwa, Rahul Garg, David E. Jacobs, Bryan E. Feldman, Nori Kanazawa, Robert Carroll, Yair Movshovitz-Attias, Jonathan T. Barron, Yael Pritch Knaan, and Marc Levoy. Synthetic depth-of-field with a single-camera mobile phone. *SIGGRAPH*, 2018.

[10] Lingyan Ruan, Bin Chen, Jizhou Li, and Miuling Lam. Learning to deblur using light field generated and real defocus images. In *CVPR*, 2022.

[11] Juewen Peng, Zhiguo Cao, Xianrui Luo, Hao Lu, Ke Xian, and Jianming Zhang. Bokehme: When neural rendering meets classical rendering. In *CVPR*, 2022.

[12] 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.

[13] 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 (ICCV)*, pages 22560–22570, October 2023.

[14] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023.

[15] Nicholas Guttenberg. Diffusion with offset noise, jan 2023.

[16] 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.

[17] Xi Chen, Lianghua Huang, Yu Liu, Yujun Shen, Deli Zhao, and Hengshuang Zhao. Anydoor: Zero-shot object-level image customization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6593–6602, June 2024.

[18] Yujun Shi, Chuhui Xue, Jun Hao Liew, Jiachun Pan, Hanshu Yan, Wenqing Zhang, Vincent Y. F. Tan, and Song Bai. Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8839–8849, June 2024.

[19] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022.

[20] Sean J. Liu, Maneesh Agrawala, Stephen DiVerdi, and Aaron Hertzmann. Zoomshop: Depth-aware editing of photographic composition. *Computer Graphics Forum*, 41(2):57–70, 2022.

[21] Karran Pandey, Paul Guerrero, Matheus Gadelha, Yannick Hold-Geoffroy, Karan Singh, and Niloy J. Mitra. Diffusion handles enabling 3d edits for diffusion models by lifting activations to 3d. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7695–7704, June 2024.

[22] Dor Verbin, Pratul P. Srinivasan, Peter Hedman, Ben Mildenhall, Benjamin Attal, Richard Szeliski, and Jonathan T. Barron. NeRF-Casting: Improved View-Dependent Appearance with Consistent Reflections. *SIGGRAPH Asia*, 2024.

[23] Zhengfei Kuang, Kyle Olszewski, Menglei Chai, Zeng Huang, Panos Achlioptas, and Sergey Tulyakov. NeROIC: Neural Rendering of Objects from Online Image Collections. *SIGGRAPH*, 2022.

[24] Xiaoming Zhao, Pratul P. Srinivasan, Dor Verbin, Keunhong Park, Ricardo Martin Brualla, and Philipp Henzler. IllumiNeRF: 3D Relighting Without Inverse Rendering. *NeurIPS*, 2024.

[25] Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B Tenenbaum. Compositional visual generation with composable diffusion models. In *ECCV*, 2022.

[26] Yilun Du, Conor Durkan, Robin Strudel, Joshua B. Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Grathwohl. Reduce, reuse, recycle: compositional generation with energy-based diffusion models and mcmc. In *Int. Conf. Mach. Learn.*, 2023.

[27] Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. Texture: Text-guided texturing of 3d shapes. In *ACM SIGGRAPH Conference Proceedings*, 2023.

[28] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[29] Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[30] Ruihang Chu, Yefei He, Zhekai Chen, Shiwei Zhang, Xiaogang Xu, Bin Xia, Dingdong Wang, Hongwei Yi, Xihui Liu, Hengshuang Zhao, et al. Wan-move: Motion-controllable video generation via latent trajectory guidance. *arXiv preprint arXiv:2512.08765*, 2025.

[31] Hadi Alzayer, Kevin Zhang, Brandon Feng, Christopher A. Metzler, and Jia-Bin Huang. Seeing the world through your eyes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[32] Brandon Y. Feng, Hadi AlZayer, Michael Rubinstein, William T. Freeman, and Jia-Bin Huang. Visualizing subtle motions with time-varying neural fields. In *International Conference on Computer Vision (ICCV)*, 2023.

[33] Hadi Alzayer, Abdullah Abuolaim, Leung Chun Chan, Yang Yang, Ying Chen Lou, Jia-Bin Huang, and Abhishek Kar. Dc2: Dual-camera defocus control by learning to refocus. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages –, 2023.

[34] Hadi Alzayer, Zhihao Xia, Xuaner (Cecilia) Zhang, Eli Shechtman, Jia-Bin Huang, and Michael Gharbi. Magic fixup: Streamlining photo editing by watching dynamic videos. *ACM Trans. Graph.*, 2025.

[35] Hadi Alzayer, Philipp Henzler, Jonathan T. Barron, Jia-Bin Huang, Pratul P. Srinivasan, and Dor Verbin. Generative multiview relighting for 3d reconstruction under extreme illumination variation. In *CVPR*, June 2025.

[36] Hadi Alzayer, Yunzhi Zhang, Chen Geng, Jia-Bin Huang, and Jiajun Wu. Coupled diffusion sampling for training-free multi-view image editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[37] Hadi Alzayer, Wenlong Huang, Haonan Chen, Christopher Luey, Lvmin Zhang, Maneesh Agrawala, Gordon Wetzstein, Li Fei-Fei, Yilun Du, Jiajun Wu, and Jia-Bin Huang. Masked visual actions for unified world modeling, 2026. Preprint.

[38] Ko Nishino and Shree K. Nayar. The World in an Eye. *CVPR*, 2004.

[39] Ko Nishino and Shree K. Nayar. Corneal Imaging System: Environment from Eyes. *IJCV*, 2006.

[40] Ko Nishino, Peter N. Belhumeur, and Shree K. Nayar. Using eye reflections for face recognition under varying illumination. In *ICCV*, 2005.

[41] Rob Jenkins and Christie Kerr. Identifiable images of bystanders extracted from corneal reflections. *PloS one*, 8(12):e83325, 2013.

[42] Xiang Zhang, Kaori Ikematsu, Kunihiro Kato, and Yuta Sugiura. Reflectouch: Detecting grasp posture of smartphone using corneal reflection images. In *CHI Conference on Human Factors in Computing Systems*, 2022.

[43] Kentaro Takemura, Tomohisa Yamakawa, Jun Takamatsu, and Tsukasa Ogasawara. Estimation of a focused object using a corneal surface image for eye-based interaction. *Journal of eye movement research*, 7(3):1–9, 2014.

[44] Huiqiong Wang, S. Lin, Xiaopei Liu, and Sing Bing Kang. Separating reflections in human iris images for illumination estimation. In *ICCV*, October 2005.

[45] Ko Nishino and Shree K. Nayar. Eyes for relighting. *SIGGRAPH*, 2004.

[46] 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. *ECCV*, 2020.

[47] Katherine L. Bouman, Vickie Ye, Adam B. Yedidia, Frédo Durand, Gregory W. Wornell, Antonio Torralba, and William T. Freeman. Turning corners into cameras: Principles and methods. In *ICCV*, 2017.

[48] Antonio Torralba and William T. Freeman. Accidental pinhole and pinspeck cameras: Revealing the scene outside the picture. In *CVPR*, 2012.

[49] Prafull Sharma, Miika Aittala, Yoav Y. Schechner, Antonio Torralba, Gregory W. Wornell, William T. Freeman, and Frédo Durand. What you can learn by staring at a blank wall. In *ICCV*, 2021.

[50] Ruoshi Liu and Carl Vondrick. Humans as light bulbs: 3d human reconstruction from thermal reflection. In *CVPR*, 2023.

[51] Simon Baker and Shree K. Nayar. A theory of catadioptric image formation. *ICCV*, 1998.

[52] S. K. Nayar. Sphereo: Determining depth using two specular spheres and a single camera. In *Proceedings of SPIE Optics, Illumination, and Image Sensing for Machine Vision III*, volume 1005, pages 245 – 254, March 1989.

[53] S.A. Nene and S.K. Nayar. Stereo with mirrors. In *ICCV*, January 1998.

[54] Norimichi Tsumura, Minh Dang, and Yoichi Miyake. Estimating the directions to light sources using images of eye for reconstructing 3d human face. *International Conference on Communications in Computing*, 2003.

[55] Atsushi Nakazawa and Christian Nitschke. Point of Gaze Estimation through Corneal Surface Reflection in an Active Illumination Environment. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *ECCV*, Lecture Notes in Computer Science, 2012.

[56] Alexander Plopski, Yuta Itoh, Christian Nitschke, Kiyoshi Kiyokawa, Gudrun Klinker, and Haruo Takemura. Corneal-Imaging Calibration for Optical See-Through Head-Mounted Displays. *IEEE Transactions on Visualization and Computer Graphics*, 21(4):481–490, April 2015.

[57] Christian Nitschke, Atsushi Nakazawa, and Haruo Takemura. Display-camera calibration from eye reflections. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1226–1233, September 2009.

[58] 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. *arXiv*, 2021.

[59] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. *CVPR*, 2022.

[60] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B. Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Transactions on Graphics*, 2021.

[61] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. *CVPR*, 2021.

[62] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable Neural Radiance Fields. *ICCV*, 2021.

[63] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust Dynamic Radiance Fields. *CVPR*, 2023.

[64] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting Neural Radiance Fields. *ICCV*, 2021.

[65] 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.

[66] Wenjing Bian, Zirui Wang, Kejie Li, Jiawang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *CVPR*, 2023.

[67] Andreas Meuleman, Yu-Lun Liu, Chen Gao, Jia-Bin Huang, Changil Kim, Min H. Kim, and Johannes Kopf. Progressively optimized local radiance fields for robust view synthesis. In *CVPR*, 2023.

[68] Keunhong Park, Philipp Henzler, Ben Mildenhall, Jonathan T. Barron, and Ricardo Martin-Brualla. Camp: Camera preconditioning for neural radiance fields. *ACM Transactions on Graphics*, 2023.

[69] Ilya Chugunov, Yuxuan Zhang, and Felix Heide. Shakes on a plane: Unsupervised depth estimation from unstabilized photography. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13240–13251, 2023.

[70] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2021.

[71] Benjamin Attal, Jia-Bin Huang, Michael Zollhöfer, Johannes Kopf, and Changil Kim. Learning Neural Light Fields with Ray-space Embedding. *CVPR*, 2022.

[72] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Transactions on Graphics*, 2022.

[73] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. In *SIGGRAPH*, SIGGRAPH '23, 2023.

[74] Sudipta N. Sinha, Johannes Kopf, Michael Goesele, Daniel Scharstein, and Richard Szeliski. Image-based rendering for scenes with reflections. *ACM Transactions on Graphics*, 2012.

[75] Tianfan Xue, Michael Rubinstein, Ce Liu, and William T. Freeman. A computational approach for obstruction-free photography. *ACM Transactions on Graphics*, 2015.

[76] Kun Gai, Zhenwei Shi, and Changshui Zhang. Blind separation of superimposed moving images using image statistics. *IEEE TPAMI*, 2012.

[77] Yu-Lun Liu, Wei-Sheng Lai, Ming-Hsuan Yang, Yung-Yu Chuang, and Jia-Bin Huang. Learning to see through obstructions. In *CVPR*, 2020.

[78] Yu-Lun Liu, Wei-Sheng Lai, Ming-Hsuan Yang, Yung-Yu Chuang, and Jia-Bin Huang. Learning to see through obstructions with layered decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(11):8387–8402, 2021.

[79] Ilya Chugunov, David Shustin, Ruyu Yan, Chenyang Lei, and Felix Heide. Neural spline fields for burst image fusion and layer separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 25763–25773, 2024.

[80] Yu Li and Michael S. Brown. Single image layer separation using relative smoothness. In *CVPR*, 2014.

[81] Renjie Wan, Boxin Shi, Ah-Hwee Tan, and Alex Chichung Kot. Depth of field guided reflection removal. In *IEEE International Conference on Image Processing*, 2016.

[82] Abhijith Punnappurath and M. S. Brown. Reflection removal using a dual-pixel sensor. In *CVPR*, 2019.

[83] Xuaner Cecilia Zhang, Ren Ng, and Qifeng Chen. Single image reflection separation with perceptual losses. In *CVPR*, 2018.

[84] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd E. Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022.

[85] Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul E. Debevec, William T. Freeman, and Jonathan T. Barron. Nerfactor: Neural factorization of shape and reflectance under an unknown illumination. *ACM Transactions on Graphics*, 2021.

[86] Kushagra Tiwary, Akshat Dave, Nikhil Behari, Tzofi Klinghoffer, Ashok Veeraraghavan, and Ramesh Raskar. Orca: Glossy objects as radiance-field cameras. In *CVPR*, 2023.

[87] Akshat Dave, Yongyi Zhao, and Ashok Veeraraghavan. Pandora: Polarization-aided neural decomposition of radiance. In *ECCV*, 2022.

[88] Yuan-Chen Guo, Di Kang, Linchao Bao, Yu He, and Song-Hai Zhang. Nerfren: Neural radiance fields with reflections. In *CVPR*, 2022.

[89] Andreas Velten, Thomas Willwacher, Otkrist Gupta, Ashok Veeraraghavan, Moungi G Bawendi, and Ramesh Raskar. Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. *Nature Communications*, 2012.

[90] Anna Pandolfi and Federico Manganiello. A model for the human cornea: Constitutive formulation and numerical analysis. *Biomechanics and Modeling in Mechanobiology*, 5(4), 2006.

[91] Ce Liu, Antonio Torralba, William T. Freeman, Frédo Durand, and Edward H. Adelson. Motion Magnification. *ACM Transactions on Graphics*, 2005.

[92] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William T. Freeman. Eulerian Video Magnification for Revealing Subtle Changes in the World. *ACM Transactions on Graphics (Proceedings SIGGRAPH 2012)*, 31(4), 2012.

[93] Wenjin Wang, Sander Stuijk, and Gerard de Haan. Exploiting Spatial Redundancy of Image Sensor for Motion Robust rPPG. *IEEE Transactions on Biomedical Engineering*, 62, 2015.

[94] Justin G. Chen, Abe Davis, Neal Wadhwa, Frédo Durand, William T. Freeman, and Oral Büyüköztürk. Video Camera–based Vibration Measurement for Civil Infrastructure Applications. *Journal of Infrastructure Systems*, 23(3), 2017.

[95] Zhexiong Shang and Zhigang Shen. Multi-point Vibration Measurement and Mode Magnification of Civil Structures Using Video-based Motion Processing. *Automation in Construction*, 93, 2018.

[96] Ricard Lado-Roigé, Josep Font-Moré, and Marco A. Pérez. Learning-based Video Motion Magnification Approach for Vibration-based Damage Detection. *Measurement*, 206, 2023.

[97] Eric R. Chan, Connor Z. Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J. Guibas, et al. Efficient Geometry-aware 3D Generative Adversarial Networks. *CVPR*, 2022.

[98] Tae-Hyun Oh, Ronnachai Jaroensri, Changil Kim, Mohamed Elgharib, Frédo Durand, William T. Freeman, and Wojciech Matusik. Learning-based Video Motion Magnification. *ECCV*, 2018.

[99] Yichao Zhang, Silvia L. Pintea, and Jan C. van Gemert. Video Acceleration Magnification. *CVPR*, 2017.

[100] Eero P. Simoncelli and William T. Freeman. The Steerable Pyramid: A Flexible Architecture for Multi-scale Derivative Computation. *ICIP*, 1995.

[101] William T. Freeman and Edward H. Adelson. The Design and Use of Steerable Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1991.

[102] Mohamed Elgharib, Mohamed Hefeeda, Fredo Durand, and William T. Freeman. Video Magnification in Presence of Large Motions. *CVPR*, 2015.

[103] Abe Davis, Michael Rubinstein, Neal Wadhwa, Gautham J. Mysore, Frédo Durand, and William T. Freeman. The visual microphone: Passive recovery of sound from video. *ACM Transactions on Graphics*, 33(4), 2014.

[104] Mark Sheinin, Dorian Chan, Matthew O'Toole, and Srinivasa G. Narasimhan. Dual-Shutter Optical Vibration Sensing. *CVPR*, 2022.

[105] Ruilong Li, Matthew Tancik, and Angjoo Kanazawa. NerfAcc: A General NeRF Accleration Toolbox. *arXiv*, 2022.

[106] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-Planes: Explicit Radiance Fields in Space, Time, and Appearance. *arXiv*, 2023.

[107] Sara Fridovich-Keil and Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance Fields without Neural Networks. *CVPR*, 2022.

[108] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. TensoRF: Tensorial Radiance Fields. *ECCV*, 2022.

[109] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and Improving Neural Radiance Fields. *arXiv*, 2020.

[110] Andreas Meuleman, Yu-Lun Liu, Chen Gao, Jia-Bin Huang, Changil Kim, Min H. Kim, and Johannes Kopf. Progressively Optimized Local Radiance Fields for Robust View Synthesis. *CVPR*, 2023.

[111] Anpei Chen, Zexiang Xu, Xinyue Wei, Siyu Tang, Hao Su, and Andreas Geiger. Factor Fields: A Unified Framework for Neural Fields and Beyond. *arXiv*, 2023.

[112] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid Neural Radiance Fields: Reconstruction and Novel View Synthesis of A Dynamic Scene from Monocular Video. *ICCV*, 2021.

[113] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, and Zhaoyang Lv. Neural 3D Video Synthesis. *CVPR*, 2022.

[114] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. *CVPR*, 2021.

[115] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time Neural Irradiance Fields for Free-Viewpoint Video. *CVPR*, 2021.

[116] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic View Synthesis from Dynamic Monocular Video. *ICCV*, 2021.

[117] Benjamin Attal, Jia-Bin Huang, Christian Richardt, Michael Zollhoefer, Johannes Kopf, Matthew O'Toole, and Changil Kim. HyperReel: High-Fidelity 6-DoF Video with Ray-Conditioned Sampling. *CVPR*, 2023.

[118] Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. NeRFPlayer: A Streamable Dynamic Scene Representation with Decomposed Neural Radiance Fields. *arXiv*, 2022.

[119] Ang Cao and Justin Johnson. HexPlane: A Fast Representation for Dynamic Scenes. *CVPR*, 2023.

[120] Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4D: Efficient Neural 4D Decomposition for High-fidelity Dynamic Reconstruction and Rendering. *CVPR*, 2023.

[121] Fuqiang Zhao, Wei Yang, Jiakai Zhang, Pei Lin, Yingliang Zhang, Jingyi Yu, and Lan Xu. HumanNeRF: Efficiently Generated Human Radiance Field From Sparse Inputs. *CVPR*, 2022.

[122] Zhongang Cai, Daxuan Ren, Ailing Zeng, Zhengyu Lin, Tao Yu, Wenjia Wang, et al. HuMMan: Multi-modal 4D Human Dataset for Versatile Sensing and Modeling. *ECCV*, 2022.

[123] Lingjie Liu, Marc Habermann, Viktor Rudnev, Kripasindhu Sarkar, Jiatao Gu, and Christian Theobalt. Neural Actor: Neural Free-View Synthesis of Human Actors with Pose Control. *ACM Transactions on Graphics*, 2021.

[124] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Transactions on Graphics*, 2020.

[125] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust Consistent Video Depth Estimation. *CVPR*, 2021.

[126] Chung-Yi Weng, Brian Curless, Pratul P. Srinivasan, Jonathan T. Barron, and Ira Kemelmacher-Shlizerman. Humannerf: Free-viewpoint Rendering of Moving People from Monocular Video. *CVPR*, 2022.

[127] Vincent Sitzmann, Julien Martel, Alexander Bergman, David B. Lindell, and Gordon Wetzstein. Implicit Neural Representations with Periodic Activation Functions. *NeurIPS*, 2020.

[128] Rizal Fathony, Anit Kumar Sahu, Devin Willmott, and J Zico Kolter. Multiplicative Filter Networks. *ICLR*, 2020.

[129] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. *CVPR*, 2019.

[130] David B. Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. BACON: Band-limited Coordinate Networks for Multiscale Scene Representation. *CVPR*, 2022.

[131] Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G Baraniuk, and Ashok Veeraraghavan. MINER: Multiscale Implicit Neural Representations. *ECCV*, 2022.

[132] Kushagra Tiwary, Akshat Dave, Nikhil Behari, Tzofi Klinghoffer, Ashok Veeraraghavan, and Ramesh Raskar. ORCa: Glossy Objects as Radiance-Field Cameras. *CVPR*, 2023.

[133] Kushagra Tiwary, Tzofi Klinghoffer, and Ramesh Raskar. Towards Learning Neural Representations from Shadows. *ECCV*, 2022.

[134] Brandon Y. Feng and Amitabh Varshney. SIGNET: Efficient Neural Representation for Light Fields. *ICCV*, 2021.

[135] Brandon Y. Feng, Yinda Zhang, Danhang Tang, Ruofei Du, and Amitabh Varshney. PRIF: Primary Ray-Based Implicit Function. *ECCV*, 2022.

[136] Brandon Y. Feng, Susmija Jabbireddy, and Amitabh Varshney. VIINTER: View Interpolation with Implicit Neural Representations of Images. *SIGGRAPH Asia*, 2022.

[137] Shayan Shekarforoush, David B. Lindell, David J. Fleet, and Marcus A. Brubaker. Residual Multiplicative Filter Networks for Multiscale Reconstruction. *arXiv*, 2022.

[138] Long Mai and Feng Liu. Motion-Adjustable Neural Implicit Video Representation. *CVPR*, 2022.

[139] 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.

[140] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[141] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *NIPS*, 2012.

[142] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-Motion Revisited. *CVPR*, 2016.

[143] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. *ECCV*, 2016.

[144] David Cardinal. Smartphones vs cameras: Closing the gap on image quality. `https://www.dxomark.com/smartphones-vs-cameras-closing-the-gap\-on-image-quality/`, Apr 2021.

[145] Samuel W. Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T. Barron, Florian Kainz, Jiawen Chen, and Marc Levoy. Burst photography for high dynamic range and low-light imaging on mobile cameras. *ACM Trans. Graph.*, 35(6), dec 2016.

[146] Bartlomiej Wronski, Ignacio Garcia-Dorado, Manfred Ernst, Damien Kelly, Michael Krainin, Chia-Kai Liang, Marc Levoy, and Peyman Milanfar. Handheld multi-frame super-resolution. *ACM Trans. Graph.*, 38(4), jul 2019.

[147] Ali Karaali and Claudio Rosito Jung. Edge-based defocus blur estimation with adaptive scale selection. *IEEE Transactions on Image Processing*, 27(3):1126–1137, 2018.

[148] Jianping Shi, Li Xu, and Jiaya Jia. Just noticeable defocus blur detection and estimation. In *CVPR*, pages 657–665, 2015.

[149] Junyong Lee, Hyeongseok Son, Jaesung Rim, Sunghyun Cho, and Seungyong Lee. Iterative filter adaptive network for single image defocus deblurring. In *CVPR*, 2021.

[150] Hyeongseok Son, Junyong Lee, Sunghyun Cho, and Seungyong Lee. Single image defocus deblurring using kernel-sharing parallel atrous convolutions. In *ICCV*, 2021.

[151] Lingyan Ruan, Bin Chen, Jizhou Li, and Miu-Ling Lam. Aifnet: All-in-focus image restoration network using a light field-based dataset. *IEEE Transactions on Computational Imaging*, 7:675–688, 2021.

[152] Haoyu Ma, Shaojun Liu, Qingmin Liao, Juncheng Zhang, and Jing-Hao Xue. Defocus image deblurring network with defocus map estimation as auxiliary task. *IEEE Transactions on Image Processing*, 31:216–226, 2022.

[153] Abdullah Abuolaim, Mahmoud Afifi, and Michael S Brown. Improving single-image defocus deblurring: How dual-pixel images help through multi-task learning. In *WACV*, 2022.

[154] Abdullah Abuolaim, Mauricio Delbracio, Damien Kelly, Michael S Brown, and Peyman Milanfar. Learning to reduce defocus blur by realistically modeling dual-pixel data. In *ICCV*, 2021.

[155] Ruikang Xu, Zeyu Xiao, Jie Huang, Yueyi Zhang, and Zhiwei Xiong. Edpn: Enhanced deep pyramid network for blurry image restoration. In *CVPR Workshops*, June 2021.

[156] Liyuan Pan, Shah Chowdhury, Richard Hartley, Miaomiao Liu, Hongguang Zhang, and Hongdong Li. Dual pixel exploration: Simultaneous depth estimation and image restoration. In *CVPR*, pages 4340–4349, June 2021.

[157] Shumian Xin, Neal Wadhwa, Tianfan Xue, Jonathan T. Barron, Pratul P. Srinivasan, Jiawen Chen, Ioannis Gkioulekas, and Rahul Garg. Defocus map estimation and deblurring from a single dual-pixel image. *ICCV*, 2021.

[158] Abdullah Abuolaim and Michael S Brown. Defocus deblurring using dual-pixel data. In *ECCV*, 2020.

[159] Anat Levin, Yair Weiss, Fredo Durand, and William T Freeman. Understanding blind deconvolution algorithms. *IEEE transactions on pattern analysis and machine intelligence*, 33(12):2354–2367, 2011.

[160] Huixuan Tang and Kiriakos N Kutulakos. Utilizing optical aberrations for extended-depth-of-field panoramas. In *ACCV*, 2012.

[161] Thomas Hach, Johannes Steurer, Arvind Amruth, and Artur Pappenheim. Cinematic bokeh rendering for real scenes. In *Proceedings of the 12th European Conference on Visual Media Production*, CVMP '15, New York, NY, USA, 2015. Association for Computing Machinery.

[162] Jonathan T. Barron, Andrew Adams, YiChang Shih, and Carlos Hernández. Fast bilateral-space stereo for synthetic defocus. In *CVPR*, pages 4466–4474, 2015.

[163] Kuldeep Purohit, Maitreya Suin, Praveen Kandula, and Rajagopalan Ambasamudram. Depth-guided dense dynamic filtering network for bokeh effect rendering. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 3417–3426, 2019.

[164] Lei Xiao, Anton Kaplanyan, Alexander Fix, Matt Chapman, and Douglas Lanman. Deepfocus: Learned image synthesis for computational display. In *ACM SIGGRAPH 2018 Talks*, pages 1–2. Association for Computing Machinery, 2018.

[165] Ke Xian, Juewen Peng, Chao Zhang, Hao Lu, and Zhiguo Cao. Ranking-based salient object detection and depth prediction for shallow depth-of-field. *Sensors*, 21(5), 2021.

[166] Andrey Ignatov, Jagruti Patel, and Radu Timofte. Rendering natural camera bokeh effect with deep learning. In *CVPR Workshops*, 2020.

[167] Hariharan Nagasubramaniam and Rabih Younes. Bokeh effect rendering with vision transformers. *TechRxiv*, 2022.

[168] Parikshit Sakurikar, Ishit Mehta, Vineeth N. Balasubramanian, and P. J. Narayanan. Refocusgan: Scene refocusing using a single image. In *ECCV*, page 519–535, Berlin, Heidelberg, 2018. Springer-Verlag.

[169] Ren Ng, Marc Levoy, Mathieu Br, Gene Duval, Mark Horowitz, Pat Hanrahan, and Duval Design. *Light Field Photography with a Hand-held Plenoptic Camera*. PhD thesis, Stanford University, 2005.

[170] David E Jacobs, Jongmin Baek, and Marc Levoy. Focal stack compositing for depth of field control. *Stanford Computer Graphics Laboratory Technical Report*, 1(1):2012, 2012.

[171] Tengfei Wang, Jiaxin Xie, Wenxiu Sun, Qiong Yan, and Qifeng Chen. Dual-camera super-resolution with aligned attention modules. In *International Conference on Computer Vision (ICCV)*, 2021.

[172] Ting-Chun Wang, Jun-Yan Zhu, Nima Khademi Kalantari, Alexei A. Efros, and Ravi Ramamoorthi. Light field video capture using a learning-based hybrid imaging system. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2017)*, 36(4), 2017.

[173] Vivek Boominathan, Kaushik Mitra, and Ashok Veeraraghavan. Improving resolution and depth-of-field of light field cameras using a hybrid imaging system. In *ICCP*, 2014.

[174] Soonmin Bae and Frédo Durand. Defocus magnification. In *Computer Graphics Forum*, volume 26, pages 571–579. Wiley Online Library, 2007.

[175] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, 2018.

[176] Pratul P. Srinivasan, Rahul Garg, Neal Wadhwa, Ren Ng, and Jonathan T. Barron. Aperture supervision for monocular depth estimation. In *CVPR*, June 2018.

[177] Lijun Wang, Xiaohui Shen, Jianming Zhang, Oliver Wang, Zhe Lin, Chih-Yao Hsieh, Sarah Kong, and Huchuan Lu. Deeplens: Shallow depth of field from a single image. *ACM Transactions on Graphics*, 37:1–11, 12 2018.

[178] Abdullah Abuolaim, Abhijith Punnappurath, and Michael S Brown. Revisiting autofocus for smartphone cameras. In *ECCV*, 2018.

[179] Abdullah Abuolaim and Michael Brown. Online lens motion smoothing for video autofocus. In *WACV*, 2020.

[180] Charles Herrmann, Richard Strong Bowen, Neal Wadhwa, Rahul Garg, Qiurui He, Jonathan T Barron, and Ramin Zabih. Learning to autofocus. In *CVPR*, 2020.

[181] Yingqian Wang, Jungang Yang, Yulan Guo, Chao Xiao, and Wei An. Selective light field refocusing for camera arrays using bokeh rendering and superresolution. *IEEE Signal Processing Letters*, 26(1):204–208, jan 2019.

[182] Sujoy Paul, Ioana S Sevcenco, and Panajotis Agathoklis. Multi-exposure and multi-focus image fusion in gradient domain. *Journal of Circuits, Systems and Computers*, 25(10):1650123, 2016.

[183] Shutao Li, James T Kwok, and Yaonan Wang. Combination of images with diverse focuses using the spatial frequency. *Information fusion*, 2(3):169–176, 2001.

[184] Shiveta Bhat and Deepika Koundal. Multi-focus image fusion techniques: a survey. *Artificial Intelligence Review*, 54(8):5735–5787, 2021.

[185] Yosuke Bando and Tomoyuki Nishita. Towards digital refocusing from a single photograph. In *15th Pacific Conference on Computer Graphics and Applications (PG'07)*, pages 363–372. IEEE, 2007.

[186] Marc Comino Trinidad, Ricardo Martin Brualla, Florian Kainz, and Janne Kontkanen. Multi-view image fusion. In *ICCV*, 2019.

[187] Paul E Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. In *ACM SIGGRAPH 2008 classes*, pages 1–10, 2008.

[188] Abdelrahman Abdelhamed, Abhijith Punnappurath, and Michael S Brown. Leveraging the availability of two cameras for illuminant estimation. In *CVPR*, 2021.

[189] Radu Ciprian Bilcu, Adrian Burian, Aleksi Knuutila, and Markku Vehvilainen. High dynamic range imaging on mobile devices. In *2008 15th IEEE International Conference on Electronics, Circuits and Systems*, pages 1312–1315. IEEE, 2008.

[190] Jian Wang, Tianfan Xue, Jonathan T Barron, and Jiawen Chen. Stereoscopic dark flash for low-light photography. In *ICCP*, 2019.

[191] Clement Godard, Kevin Matzen, and Matt Uyttendaele. Deep burst denoising. In *ECCV*, September 2018.

[192] Ben Mildenhall, Jonathan T Barron, Jiawen Chen, Dillon Sharlet, Ren Ng, and Robert Carroll. Burst denoising with kernel prediction networks. In *CVPR*, 2018.

[193] Zhilu Zhang, Ruohao Wang, Hongzhi Zhang, Yunjin Chen, and Wangmeng Zuo. Self-supervised learning for real-world super-resolution from dual zoomed observations. In *ECCV*, 2022.

[194] Yubin Wu, Lianglun Cheng, Tao Wang, and Heng Wu. Infrared and visible light dual-camera super-resolution imaging with texture transfer network. *Signal Processing: Image Communication*, 108:116825, 2022.

[195] Junyong Lee, Myeonghee Lee, Sunghyun Cho, and Seungyong Lee. Reference-based video super-resolution using multi-camera video triplets. In *CVPR*, 2022.

[196] Wei-Sheng Lai, Yichang Shih, Lun-Cheng Chu, Xiaotong Wu, Sung-Fang Tsai, Michael Krainin, Deqing Sun, and Chia-Kai Liang. Face deblurring using dual camera fusion on mobile phones. *ACM Transactions on Graphics (TOG)*, 41(4), jul 2022.

[197] Andrey Ignatov, Nikolay Kobyshev, Radu Timofte, Kenneth Vanhoey, and Luc Van Gool. Dslr-quality photos on mobile devices with deep convolutional networks. *ICCV*, 2017.

[198] Yinda Zhang, Neal Wadhwa, Sergio Orts-Escolano, Christian Häne, Sean Fanello, and Rahul Garg. Du2net: Learning depth estimation from dual-cameras and dual-pixels. In *ECCV*, Berlin, Heidelberg, 2020. Springer-Verlag.

[199] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[200] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

[201] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[202] 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 (CVPR)*, pages 10684–10695, June 2022.

[203] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12873–12883, 2021.

[204] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851, 2020.

[205] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794, 2021.

[206] Alex Andonian, Sabrina Osmany, Audrey Cui, YeonHwan Park, Ali Jahanian, Antonio Torralba, and David Bau. Paint by word. *arXiv preprint arXiv:2103.10951*, 2021.

[207] Guillaume Couairon, Jakob Verbeek, Holger Schwenk, and Matthieu Cord. Diffedit: Diffusion-based semantic image editing with mask guidance. In *The Eleventh International Conference on Learning Representations*, 2023.

[208] Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2426–2435, 2022.

[209] Vishnu Sarukkai, Linden Li, Arden Ma, Christopher Ré, and Kayvon Fatahalian. Collage diffusion. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4208–4217, 2024.

[210] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. Drag your gan: Interactive point-based manipulation on the generative image manifold. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, New York, NY, USA, 2023. Association for Computing Machinery.

[211] Shaoan Xie, Zhifei Zhang, Zhe Lin, Tobias Hinz, and Kun Zhang. Smartbrush: Text and shape guided object inpainting with diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22428–22437, 2023.

[212] Su Wang, Chitwan Saharia, Ceslee Montgomery, Jordi Pont-Tuset, Shai Noy, Stefano Pellegrini, Yasumasa Onoe, Sarah Laszlo, David J Fleet, Radu Soricut, et al. Imagen editor and editbench: Advancing and evaluating text-guided image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18359–18369, 2023.

[213] Yizhi Song, Zhifei Zhang, Zhe Lin, Scott Cohen, Brian Price, Jianming Zhang, Soo Ye Kim, and Daniel Aliaga. Objectstitch: Object compositing with diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18310–18319, 2023.

[214] Chitwan Saharia, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. Palette: Image-to-image diffusion models. In *ACM SIGGRAPH 2022 Conference Proceedings*, pages 1–10, 2022.

[215] 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.

[216] Shai Avidan and Ariel Shamir. Seam carving for content-aware image resizing. In *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07, New York, NY, USA, 2007. Association for Computing Machinery.

[217] Michael Rubinstein, Ariel Shamir, and Shai Avidan. Improved seam carving for video retargeting. *ACM transactions on graphics (TOG)*, 27(3):1–9, 2008.

[218] Yu-Shuen Wang, Chiew-Lan Tai, Olga Sorkine, and Tong-Yee Lee. Optimized scale-and-stretch for image resizing. In *ACM SIGGRAPH Asia 2008 papers*, pages 1–8. Association for Computing Machinery, 2008.

[219] Denis Simakov, Yaron Caspi, Eli Shechtman, and Michal Irani. Summarizing visual data using bidirectional similarity. In *2008 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2008.

[220] Taeg Sang Cho, Moshe Butman, Shai Avidan, and William T Freeman. The patch transform and its applications to image editing. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[221] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009.

[222] Jiaya Jia, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Drag-and-drop pasting. *ACM Trans. Graph.*, 25(3):631–637, jul 2006.

[223] Kalyan Sunkavalli, Micah K. Johnson, Wojciech Matusik, and Hanspeter Pfister. Multi-scale image harmonization. *ACM Trans. Graph.*, 29(4), jul 2010.

[224] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4432–4441, 2019.

[225] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8296–8305, 2020.

[226] Lucy Chai, Jonas Wulff, and Phillip Isola. Using latent space regression to analyze and leverage compositionality in {gan}s. In *International Conference on Learning Representations*, 2021.

[227] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision*, 2016.

[228] 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.

[229] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022.

[230] Katherine Crowson, Stella Biderman, Daniel Kornis, Dashiell Stander, Eric Hallahan, Louis Castricato, and Edward Raff. Vqgan-clip: Open domain image generation and editing with natural language guidance. In *European Conference on Computer Vision*, pages 88–105. Springer, 2022.

[231] 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.

[232] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross-attention control. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2023.

[233] 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.

[234] Chong Mou, Xintao Wang, Jiechong Song, Ying Shan, and Jian Zhang. Dragondiffusion: Enabling drag-style manipulation on diffusion models. In *The Twelfth International Conference on Learning Representations*, 2024.

[235] Grace Luo, Trevor Darrell, Oliver Wang, Dan B Goldman, and Aleksander Holynski. Readout guidance: Learning control from diffusion features. In *CVPR*, 2024.

[236] Jiraphon Yenphraphai, Xichen Pan, Sainan Liu, Daniele Panozzo, and Saining Xie. Image sculpting: Precise object editing with 3d geometry control. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[237] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

[238] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Soumyadip Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pages 843–852, June 2023.

[239] Binxin Yang, Shuyang Gu, Bo Zhang, Ting Zhang, Xuejin Chen, Xiaoyan Sun, Dong Chen, and Fang Wen. Paint by example: Exemplar-based image editing with diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18381–18391, 2023.

[240] Dave Epstein, Allan Jabri, Ben Poole, Alexei Efros, and Aleksander Holynski. Diffusion self-guidance for controllable image generation. In *Advances in Neural Information Processing Systems*, volume 36, pages 16222–16239. Curran Associates, Inc., 2023.

[241] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernández, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Mike Rabbat, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 11(1):1–28, 2024.

[242] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023.

[243] Li Hu. Animate anyone: Consistent and controllable image-to-video synthesis for character animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8153–8163, 2024.

[244] Zhongcong Xu, Jianfeng Zhang, Jun Hao Liew, Hanshu Yan, Jia-Wei Liu, Chenxu Zhang, Jiashi Feng, and Mike Zheng Shou. Magicanimate: Temporally consistent human image animation using diffusion model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1481–1490, June 2024.

[245] Daniel Geng and Andrew Owens. Motion guidance: Diffusion-based image editing with differentiable motion estimators. In *International Conference on Representation Learning*, volume 2024, pages 17451–17472, 2024.

[246] Marcelo Bertalmío, A. Bertozzi, and Guillermo Sapiro. Navier-stokes, fluid dynamics, and image and video inpainting. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 1:I–I, 2001.

[247] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020.

[248] Simon Niklaus and Feng Liu. Softmax splatting for video frame interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[249] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1623–1637, 2020.

[250] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12179–12188, October 2021.

[251] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[252] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *ICML*, 2022.

[253] Amir Hertz, Andrey Voynov, Shlomi Fruchter, and Daniel Cohen-Or. Style aligned image generation via shared attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4775–4785, June 2024.

[254] Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown, Quanfu Fan, Dan Gutfruend, Carl Vondrick, et al. Moments in time dataset: One million videos for event understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):502–517, 2020.

[255] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *IEEE International Conference on Computer Vision*, 2021.

[256] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. *CVPR*, 2021.

[257] Piotr Bojanowski, Armand Joulin, David Lopez-Pas, and Arthur Szlam. Optimizing the latent space of generative networks. *ICML*, 2018.

[258] Mark Boss, Raphael Braun, Varun Jampani, Jonathan T. Barron, Ce Liu, and Hendrik P.A. Lensch. NeRD: Neural Reflectance Decomposition from Image Collections. *ICCV*, 2021.

[259] Haian Jin, Yuan Li, Fujun Luan, Yuanbo Xiangli, Sai Bi, Kai Zhang, Zexiang Xu, Jin Sun, and Noah Snavely. Neural gaffer: Relighting any object via diffusion. *NeurIPS*, 2024.

[260] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. *CVPR*, 2023.

[261] Varun Jampani, Kevis-Kokitsi Maninis, Andreas Engelhardt, Arjun Karpur, Karen Truong, Kyle Sargent, Stefan Popov, Andre Araujo, Ricardo Martin-Brualla, Kaushal Patel, Daniel Vlasic, Vittorio Ferrari, Ameesh Makadia, Ce Liu, Yuanzhen Li, and Howard Zhou. NAVI: Category-agnostic image collections with high-quality 3D shape and pose annotations. *NeurIPS*, 2023.

[262] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. *CVPR*, 2022.

[263] Mojtaba Bemana, Karol Myszkowski, Jeppe Revall Frisvad, Hans-Peter Seidel, and Tobias Ritschel. Eikonal fields for refractive novel-view synthesis. *SIGGRAPH*, 2022.

[264] Ruofan Liang, Huiting Chen, Chunlin Li, Fan Chen, Selvakumar Panneer, and Nandita Vijaykumar. ENVIDR: Implicit differentiable renderer with neural environment lighting. *ICCV*, 2023.

[265] Liwen Wu, Sai Bi, Zexiang Xu, Fujun Luan, Kai Zhang, Iliyan Georgiev, Kalyan Sunkavalli, and Ravi Ramamoorthi. Neural directional encoding for efficient and accurate view-dependent appearance modeling. *CVPR*, 2024.

[266] Li Ma, Vasu Agrawal, Haithem Turki, Changil Kim, Chen Gao, Pedro Sander, Michael Zollhöfer, and Christian Richardt. SpecNeRF: Gaussian directional encoding for specular reflections. *CVPR*, 2024.

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

[268] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-NeRF: Scalable Large Scene Neural View Synthesis. *CVPR*, 2022.

[269] Harry Barrow, J. Tenenbaum, A. Hanson, and E. Riseman. Recovering intrinsic scene characteristics. *Computer Vision Systems*, 1978.

[270] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. MIT Press, 2023.

[271] Haian Jin, Isabella Liu, Peijia Xu, Xiaoshuai Zhang, Songfang Han, Sai Bi, Xiaowei Zhou, Zexiang Xu, and Hao Su. TensoIR: Tensorial inverse rendering. *CVPR*, 2023.

[272] Mark Boss, Varun Jampani, Raphael Braun, Ce Liu, Jonathan T. Barron, and Hendrik P.A. Lensch. Neural-PIL: Neural Pre-Integrated Lighting for Reflectance Decomposition. *NeurIPS*, 2021.

[273] Andreas Engelhardt, Amit Raj, Mark Boss, Yunzhi Zhang, Abhishek Kar, Yuanzhen Li, Deqing Sun, Jonathan T. Barron, Hendrik P.A. Lensch, and Varun Jampani. SHINOBI: Shape and Illumination using Neural Object decomposition via BRDF optimization In-the-wild. *CVPR*, 2024.

[274] Yuan Liu, Peng Wang, Cheng Lin, Xiaoxiao Long, Jiepeng Wang, Lingjie Liu, Taku Komura, and Wenping Wang. NeRO: Neural Geometry and BRDF Reconstruction of Reflective Objects from Multiview Images. *SIGGRAPH*, 2023.

[275] Zhi-Hao Lin, Bohan Liu, Yi-Ting Chen, Kuan-Sheng Chen, David Forsyth, Jia-Bin Huang, Anand Bhattad, and Shenlong Wang. Urbanir: Large-scale urban scene inverse rendering from a single video. *3DV*, 2025.

[276] Alexander Mai, Dor Verbin, Falko Kuester, and Sara Fridovich-Keil. Neural microfacet fields for inverse rendering. *ICCV*, 2023.

[277] Cheng Sun, Guangyan Cai, Zhengqin Li, Kai Yan, Cheng Zhang, Carl Marshall, Jia-Bin Huang, Shuang Zhao, and Zhao Dong. Neural-pbir reconstruction of shape, material, and illumination. *ICCV*, 2023.

[278] Pratul P. Srinivasan, Boyang Deng, Xiuming Zhang, Matthew Tancik, Ben Mildenhall, and Jonathan T. Barron. NeRV: Neural reflectance and visibility fields for relighting and view synthesis. *CVPR*, 2021.

[279] Benjamin Attal, Dor Verbin, Ben Mildenhall, Peter Hedman, Jonathan T. Barron, Matthew O'Toole, and Pratul P. Srinivasan. Flash Cache: Reducing Bias in Radiance Cache Based Inverse Rendering. *ECCV*, 2024.

[280] Sai Bi, Zexiang Xu, Pratul P. Srinivasan, Ben Mildenhall, Kalyan Sunkavalli, Miloš Hašan, Yannick Hold-Geoffroy, David Kriegman, and Ravi Ramamoorthi. Neural reflectance fields for appearance acquisition. *arXiv:2008.03824*, 2020.

[281] Zhi-Hao Lin, Jia-Bin Huang, Zhengqin Li, Zhao Dong, Christian Richardt, Tuotuo Li, Michael Zollhöfer, Johannes Kopf, Shenlong Wang, and Changil Kim. Iris: Inverse rendering of indoor scenes from low dynamic range images. *CVPR*, 2025.

[282] Kai Zhang, Fujun Luan, Qianqian Wang, Kavita Bala, and Noah Snavely. PhySG: Inverse rendering with spherical gaussians for physics-based material editing and relighting. *CVPR*, 2021.

[283] Xiuming Zhang, Pratul P. Srinivasan, Boyang Deng, Paul Debevec, William T. Freeman, and Jonathan T. Barron. NeRFactor: Neural Factorization of Shape and Reflectance Under an Unknown Illumination. *SIGGRAPH Asia*, 2021.

[284] Jean-François Lalonde, Alexei A Efros, and Srinivasa G Narasimhan. Estimating the natural illumination conditions from a single outdoor image. *IJCV*, 2012.

[285] Pratul P. Srinivasan, Ben Mildenhall, Matthew Tancik, Jonathan T. Barron, Richard Tucker, and Noah Snavely. Lighthouse: Predicting lighting volumes for spatially-coherent illumination. *CVPR*, 2020.

[286] Marc-Andre Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagne, and Jean-Francois Lalonde. Learning to predict indoor illumination from a single image. *SIGGRAPH Asia*, 2017.

[287] Mathieu Garon, Kalyan Sunkavalli, Sunil Hadap, Nathan Carr, and Jean-Francois Lalonde. Fast spatially-varying indoor lighting estimation. *CVPR*, 2019.

[288] Chloe LeGendre, Wan-Chun Ma, Graham Fyffe, John Flynn, Laurent Charbonnel, Jay Busch, and Paul Debevec. Deeplight: Learning illumination for unconstrained mobile mixed reality. *CVPR*, 2019.

[289] Hong-Xing Yu, Samir Agarwala, Charles Herrmann, Richard Szeliski, Noah Snavely, Jiajun Wu, and Deqing Sun. Accidental Light Probes. *CVPR*, 2023.

[290] Pakkapon Phongthawee, Worameth Chinchuthakun, Nontaphat Sinsunthithet, Amit Raj, Varun Jampani, Pramook Khungurn, and Supasorn Suwajanakorn. DiffusionLight: Light probes for free by painting a chrome ball. *CVPR*, 2023.

[291] Chong Zeng, Yue Dong, Pieter Peers, Youkang Kong, Hongzhi Wu, and Xin Tong. DiLightNet: Fine-grained lighting control for diffusion-based image generation. *SIGGRAPH*, 2024.

[292] Yohan Poirier-Ginter, Alban Gauthier, Julien Philip, Jean-François Lalonde, and George Drettakis. A Diffusion Approach to Radiance Field Relighting using Multi-Illumination Synthesis. *Computer Graphics Forum*, 2024.

[293] Tianyuan Zhang, Zhengfei Kuang, Haian Jin, Zexiang Xu, Sai Bi, Hao Tan, He Zhang, Yiwei Hu, Milos Hasan, William T. Freeman, Kai Zhang, and Fujun Luan. RelitLRM: Generative relightable radiance for large reconstruction models. In *The Thirteenth International Conference on Learning Representations*, 2025.

[294] Carlos Hernández, George Vogiatzis, and Roberto Cipolla. Multiview photometric stereo. *IEEE TPAMI*, 2008.

[295] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *NeurIPS*, 2022.

[296] 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. *CVPR*, 2023.

[297] 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. *NeurIPS*, 2024.

[298] Daniel Watson, William Chan, Ricardo Martin-Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel view synthesis with diffusion models. *arXiv:2210.04628*, 2022.

[299] Mehdi SM Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lučić, Daniel Duckworth, Alexey Dosovitskiy, et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. *CVPR*, 2022.

[300] Poly Haven. `https://polyhaven.com`.

[301] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *CVPR*, 2022.

[302] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv:2207.12598*, 2022.

[303] HDR Maps. `https://hdrmaps.com`.

[304] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. *CVPR*, 2024.

[305] Jason Y Zhang, Amy Lin, Moneish Kumar, Tzu-Hsuan Yang, Deva Ramanan, and Shubham Tulsiani. Cameras as rays: Pose estimation via ray diffusion. *ICLR*, 2024.

[306] Eric Brachmann, Jamie Wynn, Shuai Chen, Tommaso Cavallari, Áron Monszpart, Daniyar Turmukhambetov, and Victor Adrian Prisacariu. Scene coordinate reconstruction: Posing of image collections via incremental learning of a relocalizer. In *ECCV*, 2024.

[307] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arXiv:2410.03825*, 2024.

[308] Nadav Magar, Amir Hertz, Eric Tabellion, Yael Pritch, Alex Rav-Acha, Ariel Shamir, and Yedid Hoshen. Lightlab: Controlling light sources in images with diffusion models. In *SIGGRAPH*, 2025.

[309] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Scaling in-the-wild training for diffusion-based illumination harmonization and editing by imposing consistent light transport. In *ICLR*, 2025.

[310] Ziyi Wu, Yulia Rubanova, Rishabh Kabra, Drew A. Hudson, Igor Gilitschenski, Yusuf Aytar, Sjoerd van Steenkiste, Kelsey R Allen, and Thomas Kipf. Neural assets: 3d-aware multi-object scene synthesis with image diffusion models. In *NeurIPS*, 2024.

[311] Jiteng Mu, Michaël Gharbi, Richard Zhang, Eli Shechtman, Nuno Vasconcelos, Xiaolong Wang, and Taesung Park. Editable image elements for controllable synthesis. In *ECCV*, 2024.

[312] Vaibhav Vavilala, Seemandhar Jain, Rahul Vasanth, D. A. Forsyth, and Anand Bhattad. Generative blocks world: Moving things around in pictures, 2025.

[313] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023.

[314] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*, 2023.

[315] Ayaan Haque, Matthew Tancik, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. Instruct-nerf2nerf: Editing 3d scenes with instructions. In *ICCV*, 2023.

[316] 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.

[317] Jensen (Jinghao) Zhou, Hang Gao, Vikram Voleti, Aaryaman Vasishta, Chun-Han Yao, Mark Boss, Philip Torr, Christian Rupprecht, and Varun Jampani. Stable virtual camera: Generative view synthesis with diffusion models. In *ICCV*, 2025.

[318] Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, et al. Inference-time scaling for diffusion models beyond scaling denoising steps. In *CVPR*, 2025.

[319] Xiner Li, Yulai Zhao, Chenyu Wang, Gabriele Scalia, Gokcen Eraslan, Surag Nair, Tommaso Biancalani, Shuiwang Ji, Aviv Regev, Sergey Levine, et al. Derivative-free guidance in continuous and discrete diffusion models with soft value-based decoding. *arXiv preprint arXiv:2408.08252*, 2024.

[320] Jaihoon Kim, Taehoon Yoon, Jisung Hwang, and Minhyuk Sung. Inference-time scaling for flow models via stochastic generation and rollover budget forcing. In *NeurIPS*, 2025.

[321] Arpit Bansal, Hong-Min Chu, Avi Schwarzschild, Roni Sengupta, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Universal guidance for diffusion models. In *ICLR*, 2024.

[322] Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. In *NeurIPS*, 2022.

[323] Yinhuai Wang, Jiwen Yu, and Jian Zhang. Zero-shot image restoration using denoising diffusion null-space model. In *ICLR*, 2023.

[324] Hyungjin Chung, Jeongsol Kim, Michael T Mccann, Marc L Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *ICLR*, 2023.

[325] Vladimir Kulikov, Matan Kleiner, Inbar Huberman-Spiegelglas, and Tomer Michaeli. Flowedit: Inversion-free text-based editing using pre-trained flow models. In *ICCV*, 2025.

[326] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *CVPR*, 2023.

[327] Rahul Sajnani, Jeroen Vanbaar, Jie Min, Kapil D Katyal, and Srinath Sridhar. Geodiffuser: Geometry-based image editing with diffusion models. In *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*, 2025.

[328] Zixin Yin, Ling-Hao Chen, Lionel Ni, and Xili Dai. Consistedit: Highly consistent and precise training-free visual editing. In *SIGGRAPH Asia 2025 Conference Papers*, 2025.

[329] Haoming Cai, Tsung-Wei Huang, Shiv Gehlot, Brandon Y Feng, Sachin Shah, Guan-Ming Su, and Christopher Metzler. Parametric shadow control for portrait generation in text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 18207–18217, 2025.

[330] 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. *CVPR*, 2024.

[331] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In *NeurIPS*, 2023.

[332] David McAllister, Songwei Ge, Jia-Bin Huang, David W. Jacobs, Alexei A. Efros, Aleksander Holynski, and Angjoo Kanazawa. Rethinking score distillation as a bridge between image distributions. In *NeurIPS*, 2024.

[333] Runjie Yan, Yinbo Chen, and Xiaolong Wang. Consistent flow distillation for text-to-3d generation. In *ICLR*, 2025.

[334] Etai Sella, Gal Fiebelman, Peter Hedman, and Hadar Averbuch-Elor. Vox-e: Text-guided voxel editing of 3d objects. In *ICCV*, 2023.

[335] Yehonathan Litman, Fernando De la Torre, and Shubham Tulsiani. Lightswitch: Multi-view relighting with material-guided diffusion. In *ICCV*, 2025.

[336] Alex Trevithick, Roni Paiss, Philipp Henzler, Dor Verbin, Rundi Wu, Hadi Alzayer, Ruiqi Gao, Ben Poole, Jonathan T. Barron, Aleksander Holynski, Ravi Ramamoorthi, and Pratul P. Srinivasan. Simvs: Simulating world inconsistencies for robust view synthesis. In *CVPR*, 2025.

[337] Peng Li, Suizhi Ma, Jialiang Chen, Yuan Liu, Congyi Zhang, Wei Xue, Wenhan Luo, Alla Sheffer, Wenping Wang, and Yike Guo. Cmd: Controllable multiview diffusion for 3d editing and progressive generation. In *ACM SIGGRAPH*, 2025.

[338] Amir Barda, Matheus Gadelha, Vladimir G. Kim, Noam Aigerman, Amit H. Bermano, and Thibault Groueix. Instant3dit: Multiview inpainting for fast editing of 3d objects. In *CVPR*, 2025.

[339] Jianfeng Xiang, Zelong Lv, Sicheng Xu, Yu Deng, Ruicheng Wang, Bowen Zhang, Dong Chen, Xin Tong, and Jiaolong Yang. Structured 3d latents for scalable and versatile 3d generation. In *CVPR*, 2025.

[340] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[341] Yunzhi Zhang, Carson Murtuza-Lanier, Zizhang Li, Yilun Du, and Jiajun Wu. Product of experts for visual generation. *arXiv preprint arXiv:2506.08894*, 2025.

[342] Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. Multidiffusion: fusing diffusion paths for controlled image generation. In *Int. Conf. Mach. Learn.*, 2023.

[343] Jaihoon Kim, Juil Koo, Kyeongmin Yeo, and Minhyuk Sung. Synctweedies: A general generative framework based on synchronized diffusions. In *NeurIPS*, 2024.

[344] Luhuan Wu, Brian Trippe, Christian Naesseth, David Blei, and John P Cunningham. Practical and asymptotically exact conditional sampling in diffusion models. In *NeurIPS*, 2023.

[345] Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye. Improving diffusion models for inverse problems using manifold constraints. In *NeurIPS*, 2022.

[346] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021.

[347] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[348] Tencent Hunyuan3D Team. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation, 2025.

[349] Yen-Chi Cheng, Krishna Kumar Singh, Jae Shin Yoon, Alexander Schwing, Liangyan Gui, Matheus Gadelha, Paul Guerrero, and Nanxuan Zhao. 3D-Fixup: Advancing Photo Editing with 3D Priors. In *Proceedings of the SIGGRAPH Conference Papers*, 2025.

[350] Mohammad Asim, Christopher Wewer, Thomas Wimmer, Bernt Schiele, and Jan Eric Lenssen. Met3r: Measuring multi-view consistency in generated images. In *CVPR*, 2025.

[351] Jingyu Zhuang, Di Kang, Yan-Pei Cao, Guanbin Li, Liang Lin, and Ying Shan. Tip-editor: An accurate 3d editor following both text-prompts and image-prompts. In *ACM SIGGRAPH*, 2024.

[352] Zhangyang Qi, Yunhan Yang, Mengchen Zhang, Long Xing, Xiaoyang Wu, Tong Wu, Dahua Lin, Xihui Liu, Jiaqi Wang, and Hengshuang Zhao. Tailor3d: Customized 3d assets editing and generation with dual-side images, 2024.

[353] Fan Zhang, Shulin Tian, Ziqi Huang, Yu Qiao, and Ziwei Liu. Evaluation agent: Efficient and promptable evaluation framework for visual generative models. *arXiv preprint arXiv:2412.09645*, 2024.

[354] Black Forest Labs. Flux. `https://github.com/black-forest-labs/flux`, 2024.

[355] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023.

[356] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *ICLR*, 2023.

[357] Daniel M. Wolpert, Zoubin Ghahramani, and Michael I. Jordan. An internal model for sensorimotor integration. *Science*, 269(5232):1880–1882, 1995.

[358] Daniel M. Wolpert, R. Chris Miall, and Mitsuo Kawato. Internal models in the cerebellum. *Trends in Cognitive Sciences*, 2(9):338–347, 1998.

[359] Daniel M. Wolpert and J. Randall Flanagan. Motor prediction. *Current Biology*, 11(18):R729–R732, 2001.

[360] NVIDIA, :, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo

Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical ai, 2025.

[361] Daniel Geng, Charles Herrmann, Junhwa Hur, Forrester Cole, Serena Zhang, Tobias Pfaff, Tatiana Lopez-Guevara, Carl Doersch, Yusuf Aytar, Michael Rubinstein, Chen Sun, Oliver Wang, Andrew Owens, and Deqing Sun. Motion prompting: Controlling video generation with motion trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[362] Joonghyuk Shin, Zhengqi Li, Richard Zhang, Jun-Yan Zhu, Jaesik Park, Eli Shechtman, and Xun Huang. MotionStream: Real-Time Video Generation with Interactive Motion Controls. In *Proceedings of the International Conference on Learning Representations (ICLR*, 2026.

[363] Assaf Singer, Noam Rotstein, Amir Mann, Ron Kimmel, and Or Litany. Time-to-move: Training-free motion-controlled video generation via dual-clock denoising. In *The Fourteenth International Conference on Learning Representations*, 2026.

[364] Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. Irasim: Learning interactive real-robot action simulators. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10 2025.

[365] Nate Gillman, Charles Herrmann, Michael Freeman, Daksh Aggarwal, Evan Luo, Deqing Sun, and Chen Sun. Force prompting: Video generation models can learn and generalize physics-based control signals. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

[366] Nate Gillman, Yinghua Zhou, Zitian Tang, Evan Luo, Arjan Chakravarthy, Daksh Aggarwal, Michael Freeman, Charles Herrmann, and Chen Sun. Goal force: Teaching video models to accomplish physics-conditioned goals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[367] Wei Liu, Ziyu Chen, Zizhang Li, Yue Wang, Hong-Xing Yu, and Jiajun Wu. Realwonder: Real-time physical action-conditioned video generation, 2026.

[368] Linxi Xie, Lisong C. Sun, Ashley Neall, Tong Wu, Shengqu Cai, and Gordon Wetzstein. Generated reality: Human-centric world simulation using interactive video generation with hand and camera control. *arXiv preprint arXiv:2602.18422*, 2026.

[369] Yuang Wang, Chao Wen, Haoyu Guo, Sida Peng, Minghan Qin, Hujun Bao, Xiaowei Zhou, and Ruizhen Hu. Precise action-to-video generation through visual action prompts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 10 2025.

[370] Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, Qianli Ma, Seungjun Nah, Loic Magne, Jiannan Xiang, Yuqi Xie, Ruijie Zheng, Dantong Niu, You Liang Tan, K.R. Zentner, George Kurian, Suneel Indupuru, Pooya Jannaty, Jinwei Gu, Jun Zhang, Jitendra Malik, Pieter Abbeel, Ming-Yu Liu, Yuke Zhu, Joel Jang, and Linxi "Jim" Fan. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.

[371] Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, Michael Ryoo, Paul Debevec, and Ning Yu. Go-with-the-flow: Motion-controllable video diffusion models using real-time warped noise. In *CVPR*, 2025.

[372] Byungjun Kim, Taeksoo Kim, Junyoung Lee, and Hanbyul Joo. Dexterous world models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

[373] Raktim Gautam Goswami, Amir Bar, David Fan, Tsung-Yen Yang, Gaoyue Zhou, Prashanth Krishnamurthy, Michael Rabbat, Farshad Khorrami, and Yann LeCun. World models for learning dexterous hand-object interactions from human videos. *arXiv preprint arXiv:2512.13644*, 2026.

[374] Siqiao Huang, Jialong Wu, Qixing Zhou, Shangchen Miao, and Mingsheng Long. Vid2world: Crafting video diffusion models to interactive world models, 2025.

[375] Amir Bar, Yossi Gandelsman, Trevor Darrell, Amir Globerson, and Alexei A. Efros. Visual prompting via image inpainting, 2022.

[376] Klemen Kotar, Wanhee Lee, Rahul Venkatesh, Honglin Chen, Daniel Bear, Jared Watrous, Simon Kim, Khai Loong Aw, Lilian Naing Chen, Stefan Stojanov, et al. World modeling with probabilistic structure integration. *arXiv preprint arXiv:2509.09737*, 2025.

[377] Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023.

[378] Han Qi, Haocheng Yin, Aris Zhu, Yilun Du, and Heng Yang. Inference-time enhancement of generative robot policies via predictive world modeling. *IEEE Robotics and Automation Letters*, 2026.

[379] Gemini Robotics Team, Krzysztof Choromanski, Coline Devin, Yilun Du, Debidatta Dwibedi, Ruiqi Gao, Abhishek Jindal, Thomas Kipf, Sean Kirmani, Isabel Leal, Fangchen Liu, Anirudha Majumdar, Andrew Marmon, Carolina Parada, Yulia Rubanova, Dhruv Shah, Vikas Sindhwani, Jie Tan, Fei Xia, Ted Xiao, Sherry Yang, Wenhao Yu, and Allan Zhou. Evaluating gemini robotics policies in a veo world simulator, 2025.

[380] Yixiang Chen, Peiyan Li, Jiabing Yang, Keji He, Xiangnan Wu, Yuan Xu, Kai Wang, Jing Liu, Nianfeng Liu, Yan Huang, et al. Bridgev2w: Bridging video generation models to embodied world models via embodiment masks. *arXiv preprint arXiv:2602.03793*, 2026.

[381] Mutian Xu, Tianbao Zhang, Tianqi Liu, Zhaoxi Chen, Xiaoguang Han, and Ziwei Liu. Kinema4d: Kinematic4d world modeling for spatiotemporal embodied simulation. *arXiv preprint arXiv:2603.16669*, 2026.

[382] Haoyu Zhen, Zixian Gao, Qiao Sun, Yilin Zhao, Yuncong Yang, Yilun Du, Pengsheng Guo, Tsun-Hsuan Wang, Yi-Ling Qiao, and Chuang Gan. Action images: End-to-end policy learning via multiview video generation, 2026.

[383] Xiuyu Yang, Bohan Li, Shaocong Xu, Nan Wang, Chongjie Ye, Zhaoxi Chen, Minghan Qin, Yikang Ding, Zheng Zhu, Xin Jin, Hang Zhao, and Hao Zhao. Orv: 4d occupancy-centric robot video generation, 2025.

[384] Gen Li, Bo Zhao, Jianfei Yang, and Laura Sevilla-Lara. Mask2iv: Interaction-centric video generation via mask trajectories, 2025.

[385] Yunfan Lou, Xiaowei Chi, Xiaojie Zhang, Zezhong Qian, Chengxuan Li, Rongyu Zhang, Yaoxu Lyu, Guoyu Song, Chuyao Fu, Haoxuan Xu, Pengwei Wang, and Shanghang Zhang. Mask world model: Predicting what matters for robust robot policy learning, 2026.

[386] Marion Lepert, Ria Doshi, and Jeannette Bohg. Shadow: Leveraging segmentation masks for cross-embodiment policy transfer, 2025.

[387] Haonan Chen, Cheng Zhu, Shuijing Liu, Yunzhu Li, and Katherine Rose Driggs-Campbell. Tool-as-interface: Learning robot policies from observing human tool use. In *Proceedings of Robotics: Conference on Robot Learning (CoRL)*, 2025.

[388] Marion Lepert, Jiaying Fang, and Jeannette Bohg. Phantom: Training robots without robots using only human videos, 2025.

[389] Marion Lepert, Jiaying Fang, and Jeannette Bohg. Masquerade: Learning from in-the-wild human videos using data-editing, 2025.

[390] Eadom Dessalene, Pavan Mantripragada, Michael Maynord, and Yiannis Aloimonos. Embodiswap for zero-shot robot imitation learning, 2025.

[391] Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model, 2025.

[392] Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets, 2025.

[393] Liaoyuan Fan, Zetian Xu, Chen Cao, Wenyao Zhang, Mingqi Yuan, and Jiayu Chen. Aim: Intent-aware unified world action modeling with spatial value maps, 2026.

[394] Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4d world action modeling from video priors with asynchronous denoising, 2026.

[395] MotuBrain Team, Chendong Xiang, Fan Bao, Haitian Liu, Hengkai Tan, Hongzhe Bi, James Li, Jiabao Liu, Jingrui Pang, Kiro Jing, Louis Liu, Mengchen Cai, Rongxu Cui, Ruowen Zhao, Runqing Wang, Shuhe Huang, Yao Feng, Yinze Rong, Zeyuan Wang, and Jun Zhu. Motubrain: An advanced world action model for robot control, 2026.

[396] Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, Liliang Chen, Shuicheng Yan, Maoqing Yao, and Guanghui Ren. Genie envisioner: A unified world foundation platform for robotic manipulation, 2025.

[397] Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loic Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, and Linxi Fan. Dreamgen: Unlocking generalization in robot learning through video world models, 2025.

[398] Kaifeng Zhang, Shuo Sha, Hanxiao Jiang, Matthew Loper, Hyunjong Song, Guangyan Cai, Zhuo Xu, Xiaochen Hu, Changxi Zheng, and Yunzhu Li. Real-to-sim robot policy evaluation with gaussian splatting simulation of soft-body interactions. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

[399] Yixuan Wang, Rhythm Syed, Fangyu Wu, Mengchao Zhang, Aykut Onol, Jose Barreiros, Hooshang Nayyeri, Tony Dear, Huan Zhang, and Yunzhu Li. Interactive world simulator for robot policy training and evaluation. *arXiv preprint arXiv:2603.08546*, 2026.

[400] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in neural information processing systems*, 36:9156–9172, 2023.

[401] Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, Leslie Pack Kaelbling, et al. Video language planning. In *The Twelfth International Conference on Learning Representations*, 2023.

[402] Haoyu Zhen, Qiao Sun, Hongxin Zhang, Junyan Li, Siyuan Zhou, Yilun Du, and Chuang Gan. Tesseract: learning 4d embodied world models. *arXiv preprint arXiv:2504.20995*, 2025.

[403] Wenlong Huang, Yu-Wei Chao, Arsalan Mousavian, Ming-Yu Liu, Dieter Fox, Kaichun Mo, and Fei-Fei Li. Pointworld: Scaling 3d world models for in-the-wild robotic manipulation. In *CVPR*, 2026.

[404] Boyuan Chen, Tianyuan Zhang, Haoran Geng, Kiwhan Song, William T. Freeman, Jitendra Malik, Russ Tedrake, Vincent Sitzmann, and Yilun Du. Large video planner, 2025.

[405] Rhoda AI Team. Causal video models are data-efficient robot policy learners. *Rhoda AI Blog*, 2026.

[406] Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning, 2026.

[407] Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations, 2024.

[408] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation, 2024.

[409] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos, 2024.

[410] Po-Chen Ko, Jiayuan Mao, Yilun Du, Shao-Hua Sun, and Joshua B Tenenbaum. Learning to act from actionless videos through dense correspondences. *arXiv preprint arXiv:2310.08576*, 2023.

[411] Hongyu Li, Lingfeng Sun, Yafei Hu, Duy Ta, Jennifer Barry, George Konidaris, and Jiahui Fu. Novaflow: Zero-shot manipulation via actionable flow from generated videos, 2025.

[412] Hongyan Zhi, Peihao Chen, Siyuan Zhou, Yubo Dong, Quanxi Wu, Lei Han, and Mingkui Tan. 3dflowaction: Learning cross-embodiment manipulation from 3d flow world model, 2025.

[413] Yue Su, Xinyu Zhan, Hongjie Fang, Yong-Lu Li, Cewu Lu, and Lixin Yang. Motion before action: Diffusing object motion as manipulation condition, 2024.

[414] Ruixiang Wang, Qingming Liu, Yueci Deng, Guiliang Liu, Zhen Liu, and Kui Jia. Eva: Aligning video world models with executable robot actions via inverse dynamics rewards, 2026.

[415] Zhongru Zhang, Chenghan Yang, Qingzhou Lu, Yanjiang Guo, Jianke Zhang, Yucheng Hu, and Jianyu Chen. Veo-act: How far can frontier video models advance generalizable robot manipulation?, 2026.

[416] Haonan Chen, Jiaming Xu, Lily Sheng, Tianchen Ji, Shuijing Liu, Yunzhu Li, and Katherine Driggs-Campbell. Learning coordinated bimanual manipulation policies using state diffusion and inverse dynamics models. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[417] Jonas Pai, Liam Achenbach, Victoriano Montesinos, Benedek Forrai, Oier Mees, and Elvis Nava. mimic-video: Video-action models for generalizable robot control beyond vlas, 2025.

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

[419] Daniel M Bear, Kevin Feigelis, Honglin Chen, Wanhee Lee, Rahul Venkatesh, Klemen Kotar, Alex Durango, and Daniel LK Yamins. Unifying (machine) vision via counterfactual world modeling. *arXiv preprint arXiv:2306.01828*, 2023.

[420] Rahul Venkatesh, Honglin Chen, Kevin Feigelis, Daniel M Bear, Khaled Jedoui, Klemen Kotar, Felix Binder, Wanhee Lee, Sherry Liu, Kevin A Smith, et al. Understanding physical dynamics with counterfactual world modeling. In *European Conference on Computer Vision*, pages 368–387. Springer, 2024.

[421] Khai Loong Aw, Klemen Kotar, Wanhee Lee, Seungwoo Kim, Khaled Jedoui, Rahul Venkatesh, Lilian Naing Chen, Michael C Frank, and Daniel LK Yamins. Zero-shot world models are developmentally efficient learners. *arXiv e-prints*, pages arXiv–2604, 2026.

[422] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[423] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, Vitor Guizilini, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Muhammad Zubair Irshad, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. Droid: A large-scale in-the-wild robot manipulation dataset. 2024.

[424] Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. Robocasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations (ICLR)*, 2026.

[425] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, RISHI HAZRA, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollar, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. SAM 3: Segment anything with concepts. In *The Fourteenth International Conference on Learning Representations*, 2026.

[426] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Wensi Ai, Benjamin Martinez, Hang Yin, Michael Lingelbach, Minjune Hwang, Ayano Hiranaka, Sujay Garlanka, Arman Aydin, Sharon Lee, Jiankai

Sun, Mona Anvari, Manasi Sharma, Dhruva Bansal, Samuel Hunter, Kyu-Young Kim, Alan Lou, Caleb R Matthews, Ivan Villa-Renteria, Jerry Huayang Tang, Claire Tang, Fei Xia, Yunzhu Li, Silvio Savarese, Hyowon Gweon, C. Karen Liu, Jiajun Wu, and Li Fei-Fei. Behavior-1k: A human-centered, embodied ai benchmark with 1,000 everyday activities and realistic simulation. *arXiv preprint arXiv:2403.09227*, 2024.

[427] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[428] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 2024.

[429] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng, editors, *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 20275–20321, Suzhou, China, November 2025. Association for Computational Linguistics.

[430] Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, and Saining Xie. Inference-time scaling for diffusion models beyond scaling denoising steps. 2025.

[431] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[432] Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.