3D Inference from Unposed Sparse View Images

Bharath Raj Nagoor Kani

CMU-RI-TR-24-15

April 23

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Professor Shubham Tulsiani, *chair*
Professor Deva K. Ramanan
Professor Jun-Yan Zhu
Neehar Peri

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Robotics.

To my parents.

Abstract

We propose UpFusion, a system that can perform novel view synthesis and infer 3D representations for generic objects given a sparse set of reference images *without* corresponding pose information. Current sparse-view 3D inference methods typically rely on camera poses to geometrically aggregate information from input views, but are not robust in-the-wild when such information is unavailable/inaccurate. In contrast, UpFusion sidesteps this requirement by learning to implicitly leverage the available images as context in a conditional generative model for synthesizing novel views. We incorporate two complementary forms of conditioning into diffusion models for leveraging the input views: a) via inferring query-view aligned features using a scene-level transformer, b) via intermediate attentional layers that can directly observe the input image tokens. We show that this mechanism allows generating high-fidelity novel views while improving the synthesis quality given additional (unposed) images. We evaluate our approach on the Co3Dv2 and Google Scanned Objects datasets and demonstrate the benefits of our method over pose-reliant sparse-view methods as well as single-view methods that cannot leverage additional views. Finally, we also show that our learned model can generalize beyond the training categories and even allow reconstruction from self-captured images of generic objects in-the-wild.

Acknowledgments

Over the last 2 years at CMU, I have gained an incredible amount of growth both professionally and personally. I can't even begin to imagine if I could have achieved even a fraction of this if I did not have access to an amazing community, including my advisor, labmates, friends and peer groups. I am incredibly grateful to everyone I have interacted with at CMU who have positively impacted my life for the better.

First, I would like to extend my deepest gratitude to my advisor, Shubham Tulsiani, without whom my expertise in 3D would not have flourished. I was infact a bit scared during the advisor matching phase that I may not get to explore my interests due to my insufficient prior research experience. I am incredible grateful to Shubham for taking a chance on me, and patiently letting me gain a considerable amount of experience under his guidance.

Discussing and interacting with members of my lab and Smith Hall through group meetings, reading groups and spontaneous discussions was invaluable for my growth as a researcher. I especially learnt a lot by observing the presentation skills of senior lab members (in particular, Jason and Yufei, who are now graduating as well!), and their depth of knowledge inspired me to be well versed in the fields of my interest. I would also like to extend special thanks to Z, Hanzhe and Yanbo, with whom I had spent countless hours working together, ranting about the insane pace of research and partaking in general tomfoolery which eased the pressure of meeting conference deadlines.

To say that my friends helped me get to the finish line would be an understatement. I am incredibly grateful for the amazing friends I made at CMU who made the intense grind feel significantly less stressful. Firstly, I would like to extend my deepest gratitude to some close friends without whom I might have felt incredibly lost over the last two years. I am thankful to Jay for being a constant partner in mischief ever since the first few weeks at CMU, for our shared appreciation of classical music, and for showing me that it is possible capsize your own kayak while trying to rescue him. I am grateful to Aman for consistently reminding me that there is more to life than just work by dragging me to fun activities, for co-authoring silly pranks and for showing me that there are indeed wrong

ways to cook food. I am thankful to Prachi for being my constant support system to rant about anything, for inspiring me with her ambition and dedication to understand everything thoroughly, and for procrastinating with me for hours at a time when neither of us could afford to. I am grateful to Sandhya for our frequent hangouts, remarkably similar interests in almost everything and trade of terrible memes, which provided the relatable comfort I desperately needed when I was feeling disconnected.

Additionally, I would also like to highlight and thank some current and former Smith Hall members who made late-night work sessions all the more palatable: Anish for random water cooler banter, for sharing his makeshift snacks bar and for giving me food cravings when I least needed it; Poorvi for making even simple conversations hilarious, for being excessively kind and for her talent in putting me in awkward situations; Shagun for offering to help even when she was very busy, for the occasional long yet refreshing chats and for trading beautiful pictures of the sky. I am also incredibly grateful to all my other friends at CMU (whose names are far too many to list!) with whom I had a lot of fun playing tennis, squash and table tennis at odd hours and spending countless hours hanging out everywhere, be it on campus or at someone's place. The last semester definitely felt a little different since a lot of my friends from MSCV had left CMU for jobs, highlighting just how much they impacted my life here. I would also like to extend special thanks to my friends who drove me to various state parks and distant destinations (especially Saujas, Adi and Nikhil), without whom I would not have gotten to experience some much needed breaks from the intense grind. Finally, I want to thank some close friends from my undergrad days, who stand the test of time and distance and pamper me with kindness: Kritajnya for being a constant pillar of support and for our monthly rant sessions; Kashyap for driving me around California just like how he did back in India; Divya for showering me with gifts from halfway around the world.

Last but certainly not the least, I would like to thank my parents for encouraging me to focus on my studies despite their financial difficulties. Their unwavering support over the years gradually nudged me to work towards a PhD program after my Masters, and I am incredibly grateful for that. Thank you, Mom and Dad!

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

The long-standing problem of recovering 3D objects from 2D images has witnessed remarkable recent progress. In particular, building on the success of generative modeling, several single-view to 3D methods [24, 25] have shown promising results for reconstructing generic objects. However, such single-view methods are inherently ill-suited for capturing details of a specific object as they need to hallucinate unobserved aspects. On the other hand, multi-view 3D methods methods [20, 29] excel at recovering highly detailed 3D models of objects or scenes given densely sampled observations.

However, in real-world scenarios such as casual capture settings and online marketplaces, obtaining dense multi-view images is often impractical. Instead, only a limited set of observed views may be available, often leaving some aspects of the object unobserved. With the goal of reconstructing similarly high-fidelity 3D objects in these settings, several learning-based methods [53, 57, 62] have pursued the task of sparse-view 3D inference. While these methods can yield impressive results, they crucially rely on known accurate camera poses for the input images, which are often only available in synthetic settings or using privileged information in additional views, and are thus not currently applicable for in-the-wild sparse-view reconstruction where camera poses are not available.

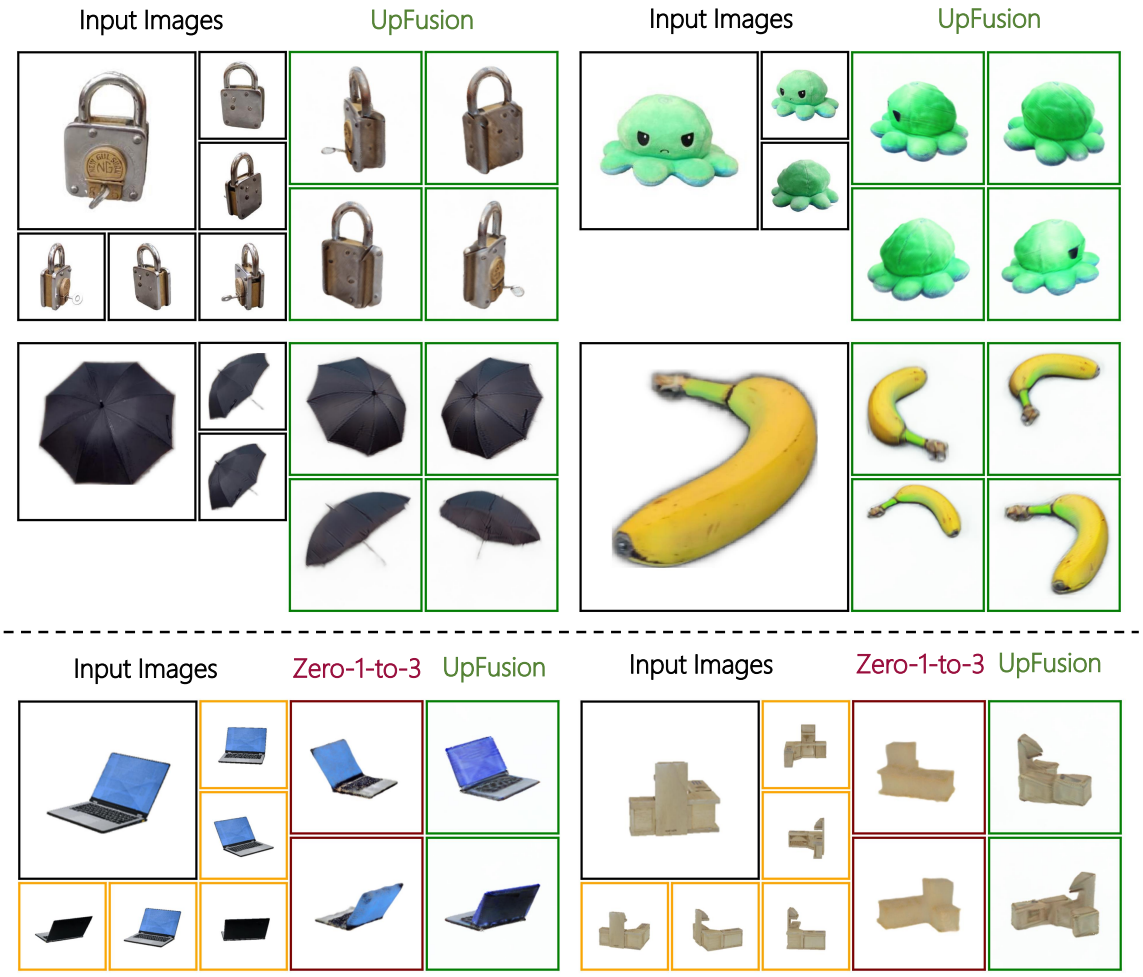

Figure 1.1: **3D Inference from Unposed Sparse views**. Given a sparse set of input images without associated camera poses, our proposed system *UpFusion* allows recovering a 3D representation of the underlying object. *Top:* Unposed input images (**black**) and corresponding novel views synthesized by UpFusion (green). *Bottom:* Compared to single-view 3D methods that cannot benefit from additional images (orange), UpFusion can better capture the instance-specific details. See supplementary for video results.

In this work, we seek to overcome the limitation of requiring known camera poses and address the task of 3D inference given *unposed* sparse views. Unlike pose-aware sparse-view 3D inference methods which use geometry-based techniques to leverage the available input, we introduce an approach that can implicitly use the available views for novel-view generation. Specifically, we designate one of the input images

as an anchor to define a coordinate frame, and adopt a scene-level transformer [38] that implicitly incorporates all available input images as context to compute per-ray features for a desired query viewpoint. Utilizing these query-aligned features, we can train a conditional denoising diffusion model to generate novel-view images.

However, we observe that relying solely on query-aligned features learned from unposed input views does not fully utilize the available context. To further enhance the instance-specificity in the generations, we propose to also add 'shortcuts' via attention mechanism in the diffusion process to allow direct attending to the input view features during the generation. Furthermore, to enable generalization to unseen categories, we adopt a pre-trained 2D foundation diffusion model [36, 59] as initialization and adapt it to leverage the two forms of context-based conditioning. Finally, the novel view images synthesized from the learned diffusion model, despite high fidelity, may not guarantee 3D consistency. Therefore, we additionally extract 3D-consistent models via score-based distillation [32, 61].

We present results using the challenging real-world dataset, Co3Dv2 [34], which comprises multi-view sequences from 51 categories with 6-DoF pose variations. To compare against recent single-view methods, we also train our model using renderings from Objaverse [7] and test on Google Scanned Objects [9]. In both these settings, we find that our approach allows extracting signal from the available unposed views, and that the performance improves with additional images. In particular, we show that our system significantly improves over recent pose-dependent methods when they fed predicted camera poses and can outperform SoTA single-view 3D methods which cannot leverage additional unposed images. We also demonstrate the ability of our method to generalize beyond the training categories by showcasing its performance on unseen object classes as well as on self-captured in-the-wild data.

Chapter 2

Related Work

2.1 3D from Dense Multi-view Captures

Multi-view observations of a scene naturally provide geometric cues for understanding its 3D structure, and this principle has been leveraged across decades to infer 3D from dense multi-view. Classical Multi-View Stereo (MVS) methods [13] leverage techniques such as structure from motion (SfM) [40] to estimate camera poses for dense matching to 3D points. Recent neural incarnations [1, 11, 29, 30, 48, 54] of these methods have further enabled breakthroughs in terms of the quality of the obtained dense 3D reconstruction. While these methods rely on classical techniques for camera estimation, subsequent approaches [2, 12, 23] have relaxed this requirement and can jointly estimate geometry and recover cameras. However, these methods are unable to predict unseen regions and crucially rely on densely-sampled images as input – a requirement our work seeks to overcome.

2.2 Single-view to 3D

On the other extreme from dense multi-view methods are approaches that aim to reconstruct a 3D representation from just a single view. While easily usable, developing such systems is highly challenging as it requires strong priors to recover unknown information. A common paradigm used to address this program is training

models conditioned on encoded image features to directly predict 3D geometry (*e.g.,* voxels [14], meshes [15, 47, 55], point clouds [10], or implicit functions [6, 28, 52]). However, given the uncertain nature of the task, the regression-based objectives in these methods limits their generation quality. More recently, there has been growing interest in distilling large text-to-image diffusion models [36, 37, 43] to generate 3D representations [5, 32, 46, 50]. Building upon these advances, several distillation-based [8, 25, 27, 33, 45, 51] and distillation-free [24, 26] single image to 3D methods were proposed. While these can infer detailed 3D, they cannot benefit from the information provided by additional (posed or unposed) views. Moreover, as they hallucinate details in unobserved regions, the reconstructed object may significantly differ from the one being imaged. If a user aims to faithfully capture a specific object of interest, single-view methods are fundamentally ill-suited for this task.

2.3 Sparse-view to 3D

With the goal of reducing the burden in the multi-view capture process while still enabling detailed capture of specific objects of interest, there has been a growing interest in spare-view 3D inference methods. By leveraging the benefits of both multi-view geometry and learning, regression-based methods achieve 3D consistency by using re-projected features obtained from input views [34, 49, 56]. However, the results tend to be blurry due to the mean-seeking nature of regression methods under uncertainty. To improve the quality of generations, another stream of work [4, 21, 35, 61] formulate the problem as a probabilistic generation task. These methods achieve better perceptual quality, yet usually require precise pose information, which is often not practically available. To overcome this issue, one may either consider leveraging recent sparse-view pose estimation methods [41, 58] in conjunction with state-of-the-art novel-view synthesis methods, or consider methods that optimize poses jointly with the objective of novel-view synthesis [19, 42]. However, the computation of explicit poses may not always be robust, and we empirically show that this leads to poor performance. Closer to our approach, UpSRT [38] and RUST [39] allow novel view synthesis without explicit pose estimation (i.e., directly from unposed sparse views). However, their regression-based pipelines limit the quality of the synthesized outputs.

Chapter 3

Approach

Our goal is to infer a 3D representation of an object given a sparse set of images. While prior works [4, 56, 61] typically aggregate information from the input views by using geometric projection and unprojection, these crucially rely on the availability of accurate camera poses which are not readily available in-the-wild. We instead aim to tackle the task of 3D inference given *unposed* sparse views.

Towards building a system capable of 3D inference in this unposed setting, we propose a mechanism for implicitly leveraging the available images as context when generating novel views. Specifically, we adapt Unposed Scene Representation Transformer (UpSRT) [38], a prior work that leverages transformers as a mechanism for implicitly aggregating information from input views, and computes query-view-aligned features for view synthesis. However, instead of their mean-seeking regression objective which results in blurry renderings, we enable probabilistic sparse view synthesis by using the internal representations of UpSRT to condition a diffusion model to perform novel view synthesis. While our diffusion model can yield high-fidelity generations, the outputs are not 3D consistent. To obtain a consistent 3D representation, we then train instance-specific neural representations [30, 44] which maximizes the likelihood of the renderings under the learned generative model. We detail our approach below, but first briefly review UpSRT and denoising diffusion models [18] that our work builds on.

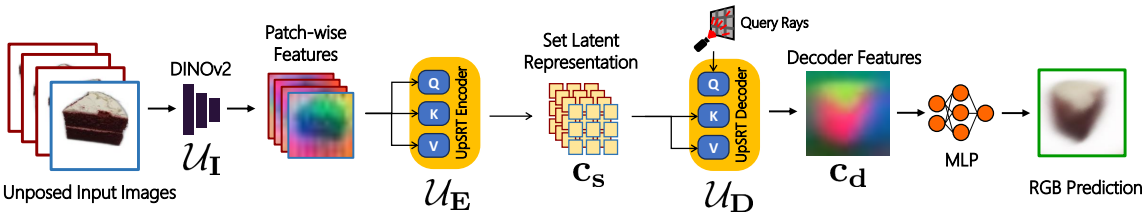

Figure 3.1: **UpSRT** [38] performs novel view synthesis from a set of unposed images. UpSRT consists of an encoder, a decoder, and an MLP. The encoder takes encoded image features as inputs and outputs a set-latent representation $\boldsymbol{c}_s$. The decoder takes query rays as inputs and attends to the set-latent representation to get features $\boldsymbol{c}_d$, which are then fed into MLP to obtain final novel view RGB images. The first input image (blue) is used as an anchor to define the coordinate system. We make use of both $\boldsymbol{c}_s$ and $\boldsymbol{c}_d$ to provide conditional context to our model.

3.1 Preliminaries

3.1.1 Unposed Scene Representation Transformer.

Given a set of N images $\boldsymbol{\mathcal{I}} = \{I_1, I_2, ..., I_N\}$, UpSRT [38] seeks to generate novel view images by predicting RGB color r for any query ray q, where the first input image is used as an anchor to define the coordinate system (see supplementary for details). As illustrated in figure 3.1, it first extracts patch-wise features for each image I_i with an image encoder $\mathcal{U}_I$. Then, it uses an encoder transformer $\mathcal{U}_E$ to obtain a set latent representation $\boldsymbol{c}_s$. Finally, it uses a decoder transformer $\mathcal{U}_D$ which attends to v_c, followed by an MLP, to predict the RGB color. In summary, the UpSRT workflow can be represented by the following equations:

$$\boldsymbol{c}_s = \mathcal{U}_E(\{\mathcal{U}_I(\boldsymbol{\mathcal{I}})\}), C(\boldsymbol{r}) = \text{MLP}(\mathcal{U}_D(\boldsymbol{r}|\boldsymbol{c}_s)) \tag{3.1}$$

We pre-train an UpSRT model using a pixel-level regression loss and leverage it for subsequent generative modeling. While we follow a similar design, we make several low-level modifications from the originally proposed UpSRT architecture (*e.g.*, improved backbone, differences in positional encoding, *etc.*), and we expand on these in the appendix.

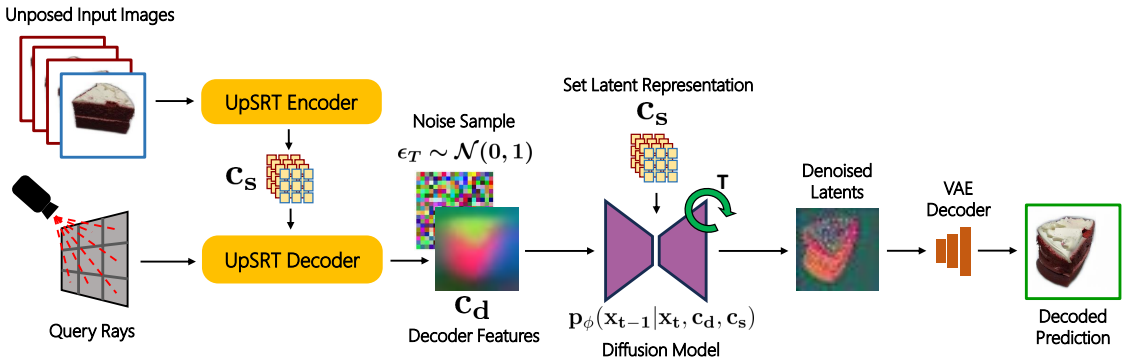

Figure 3.2: **UpFusion 2D** is the proposed conditional diffusion model performing novel view synthesis conditional on information extracted from a set of unposed images. To reason about the query view, Upfusion takes as additional inputs the view-aligned decoder features $\boldsymbol{c}_d$ obtained from UpSRT decoder. To further allow the model to attend to details from input views, UpFusion conditions on the set-latent representation $\boldsymbol{c}_s$ via attentional layers.

3.1.2 Denosing Diffusion.

Denoising diffusion models [18] seek to learn a generative model over data samples $\boldsymbol{x}$ by learning to reverse a forward process where noise is gradually added to original samples. The learning objective can be reduced to a denoising error, where a diffusion model ϵ_ϕ is trained to estimate the noise added to a current sample $\boldsymbol{x}_t$:

$$\mathcal{L}_{DM} = \mathbb{E}_{\boldsymbol{x}_0, t, \epsilon \sim \mathcal{N}(0,1)}[\|\epsilon_t - \epsilon_\phi(\boldsymbol{x}_t, t)\|_2^2] \tag{3.2}$$

While the above objective summarizes an unconditional diffusion model, it can be easily adapted to learn conditional generative models $p(\boldsymbol{x}|\boldsymbol{y})$ by adding a condition $\boldsymbol{y}$ (such as a set of unposed images) to the input of the denoising model $\epsilon_\phi(\boldsymbol{x}_t, t, \boldsymbol{y})$.

3.2 Probabilistic View Synthesis using Sparse Unposed Views

We aim to learn a generative model over novel views of an object given a sparse set of unposed images. Given this, our goal is to learn the distribution $p(\mathbf{I}|\boldsymbol{\mathcal{I}}, \pi)$, where π denotes a query pose, $\boldsymbol{\mathcal{I}}$ denotes the set of unposed images and $\mathbf{I}$ denotes the query-

view image. Instead of learning the distribution directly in pixel space, we follow a common practice of instead learning this distribution in latent space $p(\boldsymbol{x}|\boldsymbol{\mathcal{I}}, \pi)$, using pre-trained encoders and decoders corresponding to this latent space [36]: $\boldsymbol{x} = \mathcal{E}(\mathbf{I});\ \mathbf{I} = \mathcal{D}(\boldsymbol{x})$.

We model this probability distribution by training a conditional diffusion model which leverages the available unposed images as context, and seek to propose an architecture that embraces several desirable design principles. First, we note that such a diffusion model must be able to (implicitly) reason about the query view it is tasked with generating in the context of the available input, and leverage the UpSRT encoder-decoder framework to enable this. While the decoder features from UpSRT can ground the query-view generation, we note that these may abstract away the salient details in the input, and we propose to complement these by allowing the generative model to directly leverage the patch-wise latent features and more easily 'copy' content from input views. Lastly, to enable efficient training and generalization beyond training data, we propose to adapt off-the-shelf diffusion models for view-conditioned generation.

View-aligned Features for Image Generation. Given a target view π, we construct a set of rays $\mathcal{R}$ corresponding to a grid of 2D pixel locations in this view. We query the UpSRT decoder with this set of rays to obtain view-aligned decoder features $\boldsymbol{c}_d$ of the same resolution as the image latent $\boldsymbol{x}$. As illustrated in figure 3.2, these query-aligned features are concatenated with the (noisy) image latents to serve as inputs to the denoising diffusion model.

Incorporating Direct Attention to Input Patches. To allow the generation model to directly incorporate details visible in the input views, we also leverage the set-latent feature $\boldsymbol{c}_s$ representation extracted by the UpSRT encoder. Importantly, this representation comprises of per-patch features aligned with the input images and allows efficiently 'borrowing' details visible in these images. Unlike the view-aligned decoder feature which can be spatially concatenated with the noisy diffusion input, we condition on these set-latent features via attentional layers in the generation model.

Adapting Large-scale Diffusion Models for Novel-view Synthesis. Instead of training our generative model from scratch, we aim to take advantage of the strong priors learned by large diffusion models such as Stable Diffusion [36]. To

this end, we use a modified version of the ControlNet architecture [59] to adapt a pre-trained Stable Diffusion model to incorporate additional conditionings $\boldsymbol{c}_d, \boldsymbol{c}_s$ for view generation.

Putting it Together. In summary, we reduce the task of modeling $p(\boldsymbol{x}|\boldsymbol{\mathcal{I}}, \pi)$ to learning a denoising diffusion model $p_\phi(\boldsymbol{x}|\boldsymbol{c}_d, \boldsymbol{c}_s)$, and leverage the ControlNet architecture to incorporate the two conditioning features and learn a denoising model $\epsilon_\phi(\boldsymbol{x}_t, t, \boldsymbol{c}_d, \boldsymbol{c}_s)$. More specifically, ControlNet naturally allows adding the spatial feature $\boldsymbol{c}_d$ as via residual connections to the spatial layers of the UNet in a pre-trained Stable Diffusion model. To incorporate the set-level features $\boldsymbol{c}_s$, we modify the ControlNet encoder blocks to use $\boldsymbol{c}_s$ in place of a text encoding (see appendix for details). We can train such a model using any multi-view dataset, where we train the denoising diffusion model to generate the underlying image from a query view given a variable number of observed input views.

3.3 Inferring 3D Consistent Representations

While the proposed conditional diffusion model can provide high-fidelity renderings from query views, the generated views are not 3D consistent. To obtain a 3D representation given the inferred distribution over novel views, we subsequently optimize an instance-specific neural representation. Towards this, we follow SparseFusion [61] which seeks neural 3D modes by optimizing the likelihood of their renderings by adapting a Score Distillation Sampling (SDS) [32] loss to view-conditioned generative models.

Specifically, we optimize a neural 3D representation g_θ by ensuring its renderings have high likelihood under our learned distribution $p(\mathbf{I}|\boldsymbol{\mathcal{I}}, \pi)$. We do so by minimizing the difference between the renderings of the instance-specific neural model and the denoised predictions from the learned diffusion model. Denoting by $g_\theta(\pi)$ the rendering of the neural 3D representation from viewpoint π, and by $\hat{\boldsymbol{x}}_0$ the denoised prediction inferred from the learned diffusion model $\epsilon_\phi(\boldsymbol{x}_t; t, \boldsymbol{c}_d, \boldsymbol{c}_s)$, the training objective can be specified as:

$$\mathcal{L}_{3D} = \mathbb{E}_{t,\epsilon,\pi}[\|g_\theta(\pi) - \mathcal{D}(\hat{\boldsymbol{x}}_0)\|^2] \tag{3.3}$$

Unlike SparseFusion [61] which additionally uses a rendering loss for the available

input views using known cameras, we rely only on the above denoising objective for optimizing the underlying 3D representation given unposed input views.

3.4 Training Details

We follow a multi-stage training procedure to optimize our models. We first train the UpSRT model separately using a reconstruction loss on the color predicted for query rays given the set of reference images $\mathbf{I}$. Then, we train the denoising diffusion model while using the conditioning information from the pre-trained UpSRT, which is frozen in this stage.

To enable the usage of classifier-free guidance [17] during inference, we train our diffusion model in the unconditional mode for a small fraction of the time. We do this by following the condition dropout procedure used in [3, 25] that randomly replaces the conditioning information with null tokens (for more details, please refer to the supplementary).

Once the diffusion model is trained, we can extract a 3D representation for an object by optimizing an Instant-NGP [30, 44] using the neural mode seeking objective discussed in Section 3.3. We use DDIM [43] for fast multi-step denoising. Inspired by [50], we follow an annealed time schedule for score distillation. We also use some regularization losses while training the NeRF as used in [61]. For more details, please refer to the supplementary.

Chapter 4

Experiments

4.1 Experimental Setup

4.1.1 Dataset

We test our approach in two different experimental setups. First, we use the Co3Dv2 [34], a large-scale dataset with real multi-view images of objects from 51 categories. Following [22, 58], we train our model on 41 categories and hold out 10 categories to test the ability of our method to generalize to unseen categories. We use the *fewview-train* split for training and *fewview-dev* split for evaluation. We limit our focus to modelling only objects and not their backgrounds. To this end, we create a white background for our objects by using the masks available in the dataset. As our full method (as well as some baselines) optimize instance-specific neural representations, which can take 1hr per instance, we limit our evaluations to 5 object instances per category. To account for the inherent (scale) ambiguity in the coordinate systems between the predictions and ground-truth, we define aligned versions of the typical image reconstruction metrics (PSNR-A, SSIM-A, LPIPS-A). For more details, please refer to the supplementary.

To compare with popular state-of-the-art single-view baselines that are trained on Objaverse [7], we also fine-tune a version of our model (which are already pre-trained on Co3Dv2) on Objaverse renderings. We denote versions of our model fine-tuned on Objaverse with † as a superscript (for example, UpFusion† (3D)) and

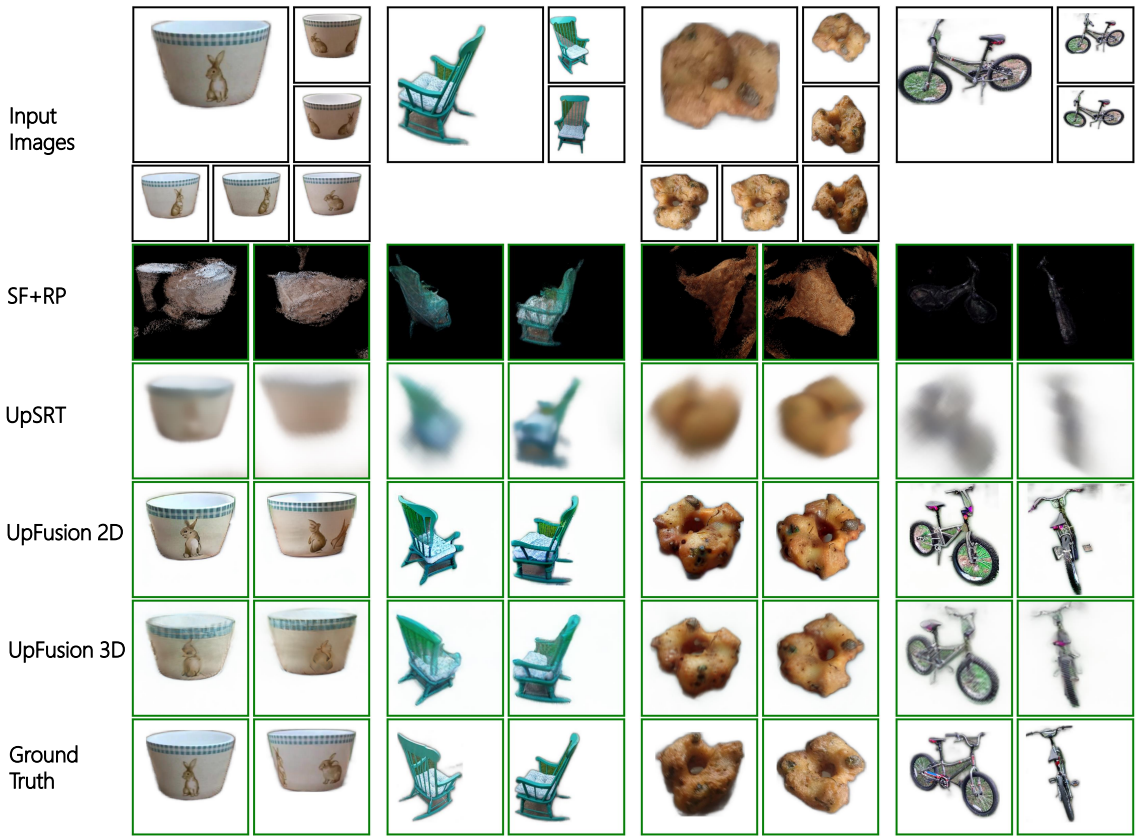

Figure 4.1: **Qualitative comparison with sparse-view baselines.** We compare UpFusion with baseline methods using 3 and 6 unposed images as inputs. SparseFusion fails to capture the correct geometry, due to the imperfect camera poses estimated by RelPose++. UpSRT generates blurry results due to the nature of regression-based methods. On the contrary, UpFusion 2D synthesizes sharp outputs with correct object poses. UpFusion 3D further improves the 3D consistency.

evaluate this model using the Google Scanned Objects [9] dataset. As both Objaverse and GSO comprise of normalized origin-centered objects (unlike CO3D where the reconstructions are in arbitrary SfM coordinate systems), we use the 'normal' image reconstruction metrics for evaluation.

4.1.2 Baselines

We highlight the benefits of our approach by comparing it to prior pose-dependent and unposed novel-view generation techniques. Specifically, we compare our 2D

diffusion model ('UpFusion 2D') and obtained 3D representations ('UpFusion 3D') against the following baselines:

SparseFusion [61] is a representative method for pose-dependent sparse-view inference on Co3Dv2. We compare against its performance when using a recent sparse-view pose estimation system RelPose++ [22], and also report its performance using GT camera poses as an upper bound.

UpSRT. We compare against the prediction from the UpSRT [38] backbone used in our approach, where the improvements over this indicate the gains from the diffusion and subsequent 3D distillation.

FORGE [19] is a method that jointly optimizes for poses while being trained on a novel-view synthesis objective. As FORGE uses the GSO dataset [9] to demonstrate its generalization capability, we compare it against our Objaverse fine-tuned UpFusion† (3D).

Single-view methods. To highlight the benefit of using more input views, we compare UpFusion† (3D) to two representative state-of-the-art single-view baselines: Zero-1-to-3 [25] and One-2-3-45 [24]. For Zero-1-to-3, we include comparisons with two versions – the original version which uses SJC [46] and the highly optimized threestudio implementation [16] (which uses additional tricks to aid 3D distillation). We compare against these baselines on the GSO dataset.

Type	Method	PSNR-A (↑)			SSIM-A (↑)			LPIPS-A (↓)		
		1V	3V	6V	1V	3V	6V	1V	3V	6V
Posed	SparseFusion (GT)	—	22.41	24.02	—	0.79	0.81	—	0.20	0.18
Unposed	SparseFusion (RelPose++)	—	17.76	17.12	—	0.67	0.64	—	0.30	0.33
	UpSRT	16.84	17.75	18.36	0.73	0.74	0.75	0.34	0.32	0.31
	UpFusion (2D)	16.54	17.12	17.41	0.71	0.72	0.73	0.23	0.22	0.22
	UpFusion (3D)	**18.17**	**18.68**	**18.96**	**0.75**	**0.76**	**0.76**	**0.22**	**0.21**	**0.21**

Table 4.1: **Sparse-view synthesis evaluation on seen categories (41 categories).** We conduct comparisons using 5 samples per category and then and report the average across these. UpFusion performs favorably against baseline methods, and demonstrates the capability to leverage additional unposed images. Moreover, UpFusion 3D consistently improves the results from UpFusion 2D.

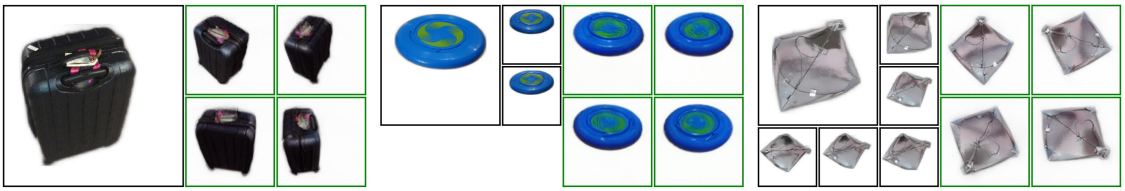

Figure 4.2: **Generalization beyond training categories.** We show results for UpFusion (3D) across object categories *not* seen in training. For each instance, we present the 1, 3, or 6 unposed input views (left), as well as 4 novel view renderings (right). We observe that despite not being trained on these categories, UpFusion is able to accurately infer the underlying 3D structure and generate detailed novel views.

4.2 Results

4.2.1 Novel-view synthesis on CO3Dv2

We compare UpFusion with baseline methods on the categories seen during training, as shown in Table 4.1. UpFusion performs favorably against both UpSRT and unposed SparseFusion. Furthermore, UpFusion consistently improves the prediction when more views are provided. However, there is still room for improvement compared to the methods using ground-truth poses. In Figure 4.1, we qualitatively present the novel view synthesis results. SparseFusion can capture some details visible in the input views but largely suffers due to the error in input poses. UpSRT, on the other hand, can robustly generate coarse renderings, but is unable to synthesize high-fidelity outputs from any viewpoints. Our 2D diffusion model, UpFusion 2D, generates higher

Method	PSNR-A (↑)			SSIM-A (↑)			LPIPS-A (↓)		
	1V	3V	6V	1V	3V	6V	1V	3V	6V
UpSRT	16.75	17.57	18.06	0.73	0.74	0.74	0.35	0.33	0.32
UpFusion (2D)	16.33	17.04	17.38	0.70	0.71	0.72	0.25	0.23	0.23
UpFusion (3D)	**18.27**	**18.83**	**19.11**	**0.75**	**0.76**	**0.76**	**0.23**	**0.22**	**0.22**

Table 4.2: **Sparse-view synthesis evaluation on unseen categories (10 categories).** We conduct comparisons using 5 samples per category and report the average across these. We observe a comparable performance to the results on seen categories.

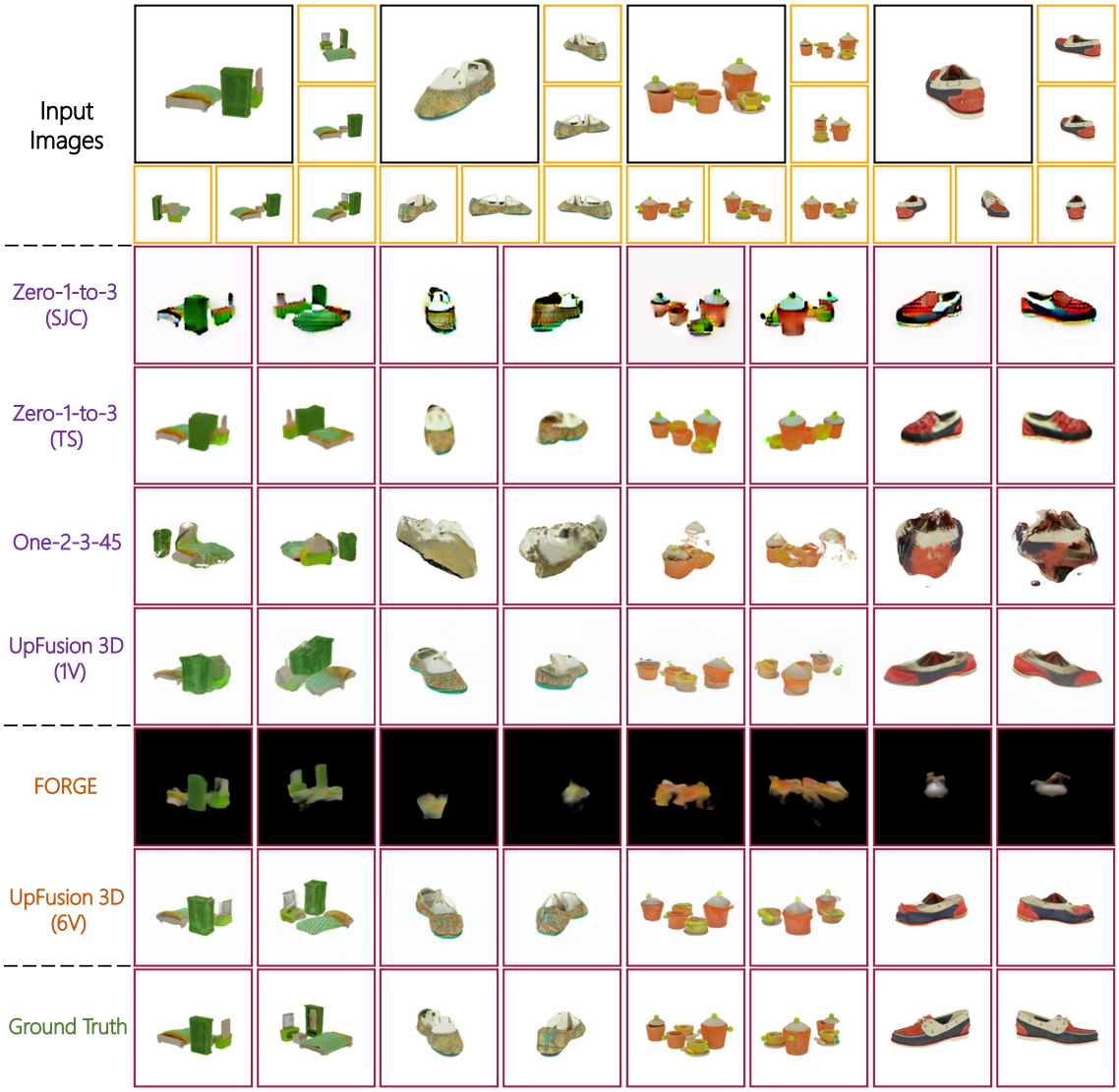

Figure 4.3: **Qualitative comparison on GSO.** We compare UpFusion† (3D) to two single-view baselines and one sparse-view baseline (FORGE) on the GSO dataset. For each instance, single-view methods use only the image with the black border as input, whereas sparse-view methods use all input images. We can observe that UpFusion† (3D) while using 6 inputs views is able to better understand the 3D structure of the object than the single-view baselines (*e.g.*, size of cabinet in bedroom). Moreover, it is able to incorporate information from the 6 inputs views much better than the sparse-view baseline.

# Input Views	Method	PSNR (↑)	SSIM (↑)	LPIPS (↓)
1V	Zero-1-to-3 (SJC)	18.72	0.90	0.12
	Zero-1-to-3 (TS)	21.71	**0.91**	0.09
	One-2-3-45	17.77	0.87	0.15
	UpFusion† (3D)	20.52	0.89	0.12
6V	FORGE	17.40	0.88	0.15
	UpFusion† (3D)	**22.51**	**0.91**	**0.08**

Table 4.3: **Novel-view synthesis evaluation on GSO.** We compare UpFusion 3D to single-view baselines as well as a sparse-view pose-optimization baseline on GSO dataset which is out of distribution for all methods.

fidelity images that improve over the baselines in the perceptual metrics. Finally, the 3D-consistent inferred representation Upfusion3D yields the best results.

Characterizing Generalization. As UpFusion is trained upon a pre-trained large-scale diffusion model providing strong general priors, the learned novel view synthesis capability is expected to be generalized to categories beyond training. We evaluate UpFusion on 10 unseen categories, as shown in Table 4.2. Encouragingly, we find that the performance does not degrade compared to the results on seen categories and believe this highlights the potential of our approach to perform in-the-wild sparse-view

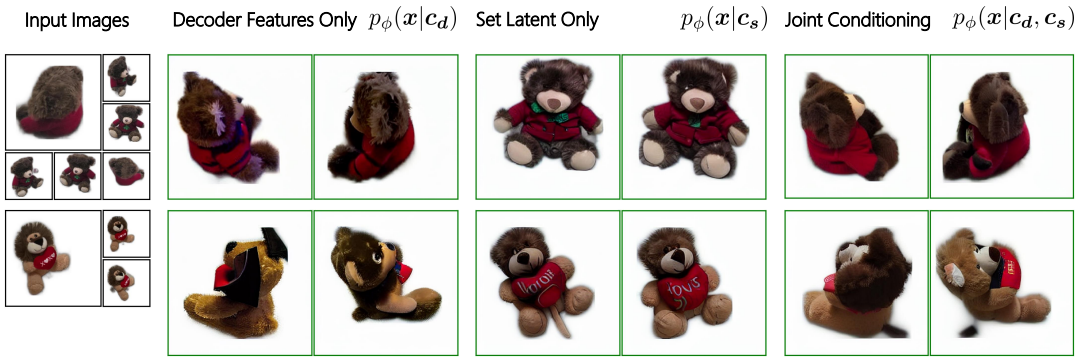

Figure 4.4: **Ablation of generative model conditioning.** Visualizations from category-specific models trained teddybears using varying conditioning for novel-view generation. We find that the model using only set-latent conditioning is unable to understand the query pose, while the one relying on only decoder features does not preserve instance identity. Our full model, using both features, allows respecting viewpoint and preserving instance-specific details.

Conditioning	PSNR-A (↑)			SSIM-A (↑)			LPIPS-A (↓)		
	1V	3V	6V	1V	3V	6V	1V	3V	6V
DF Only	14.65	15.28	15.53	0.63	0.64	0.64	0.32	0.30	0.30
SLT Only	13.15	13.38	13.49	0.60	0.60	0.60	0.36	0.35	0.35
DF+SLT	**15.58**	**16.11**	**16.26**	**0.65**	**0.66**	**0.66**	**0.30**	**0.28**	**0.28**

Table 4.4: **Ablation of generative model conditioning.** We ablate our conditional diffusion model with different conditional contexts. **DF** stands for decoder features $\boldsymbol{c}_d$, and **SLT** stands for set-latent representations $\boldsymbol{c}_s$. We train category-specific UpFusion (2D) models for this ablation on the teddybear category and report performance.

3D inference. We also depict some qualitative results on unseen objects in Figure 4.5.

4.2.2 Novel-view synthesis on GSO

We compare UpFusion† (3D) to two state-of-the-art single-view baselines (Zero-1-to-3 and One-2-3-45) and a sparse-view baseline (FORGE) on 20 randomly sampled instances from the GSO dataset. For Zero-1-to-3, we compare with both the original SJC implementation and threestudio (TS) implementation. From Table 4.3, we can

Figure 4.5: **3D from self-captured images.** Given 3-6 self-captured input images for a generic object (left), we show 4 novel viewpoints of the 3D asset recovered via UpFusion† (3D) (right).

observe that UpFusion† (3D) while using 6 inputs views is able to outperform all baselines. This demonstrates the ability of our method to effectively incorporate more information when additional views are available, which single-view baselines cannot. Moreover, we can see that our model significantly outperforms FORGE, which also uses 6 input views, and we believe this is because our approach allows bypassing explicit pose prediction which can lead to inaccurate predictions. Qualitative comparisons in Figure 4.3 further demonstrates the effectiveness of our approach in utilizing information from multiple unposed images.

4.2.3 Ablating Diffusion Conditioning

We empirically study the complementary benefits of the two forms of conditioning used. We illustrate the qualitative results in Figure 4.4 and quantitative results in Table 4.4. We find that both, the decoder features and the set-latent representations are complementary and instrumental to UpFusion.

Chapter 5

Discussion

We presented UpFusion, an approach for novel-view synthesis and 3D inference given unposed sparse views. While our approach provided a mechanism for effectively leveraging unposed images as context, we believe that several challenges still remain towards the goal of sparse-view 3D inference in-the-wild. In particular, although our approach allowed high-fidelity 2D generations, these are not always precisely consistent with the details in the (implicitly used) input views. Moreover, while our approach's performance does improve given additional context views, it does not exhibit a strong scaling similar to pose-aware methods that can geometrically identify relevant aspects of input images. Finally, while our work provided a possible path for 3D inference from unposed views by sidestepping the task of pose estimation, it remains an open question whether explicit pose inference for 3D estimation might be helpful in the long term.

Appendix A

Additional Results

We visualize additional samples from UpFusion (3D) for seen and unseen categories in Figures (A.1, A.2, A.3) and Figure A.4 respectively.

Figure A.1: **Additional results with 1 input view.** We show results for UpFusion (3D) across different object categories given 1 input view (left), and show 4 novel view renderings (right).

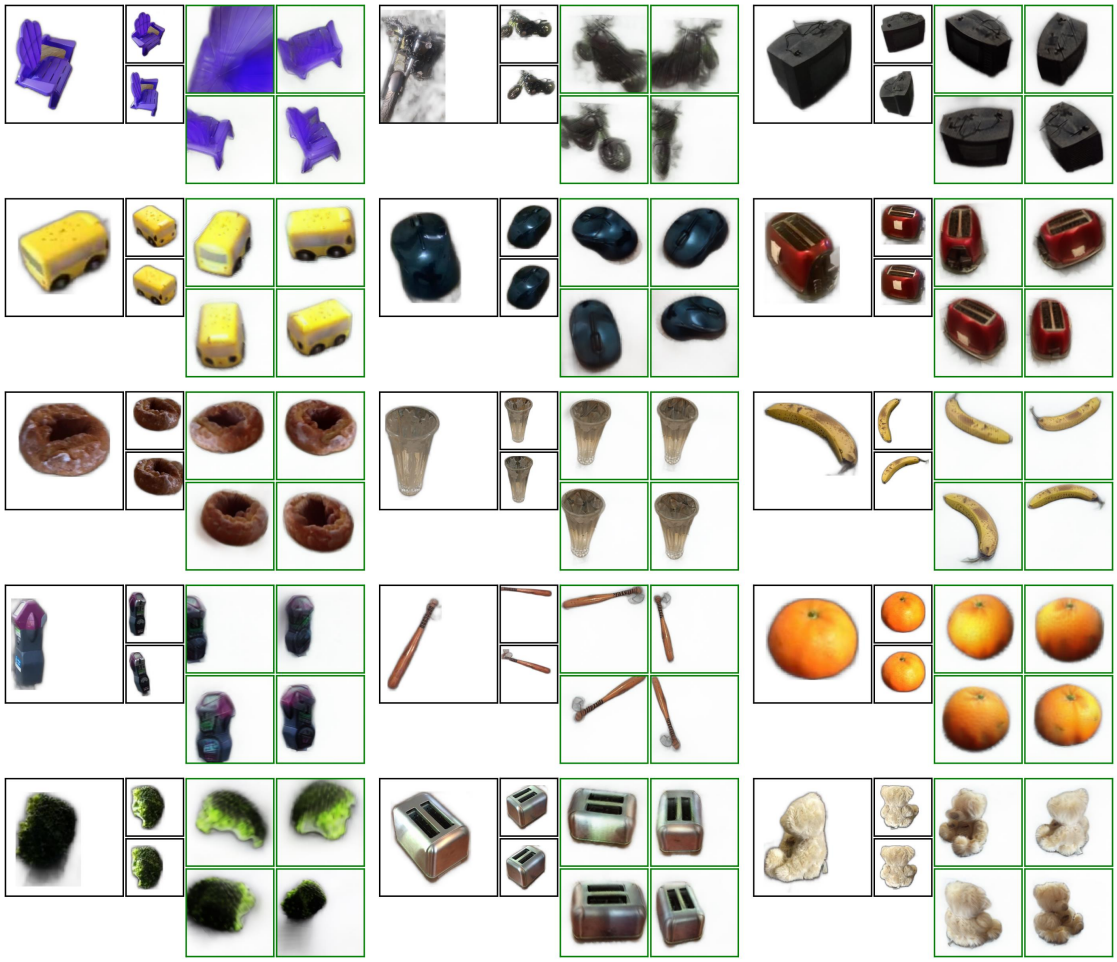

Figure A.2: **Additional results with 3 input views.** We show results for UpFusion (3D) across different object categories given 3 input views (left), and show 4 novel view renderings (right).

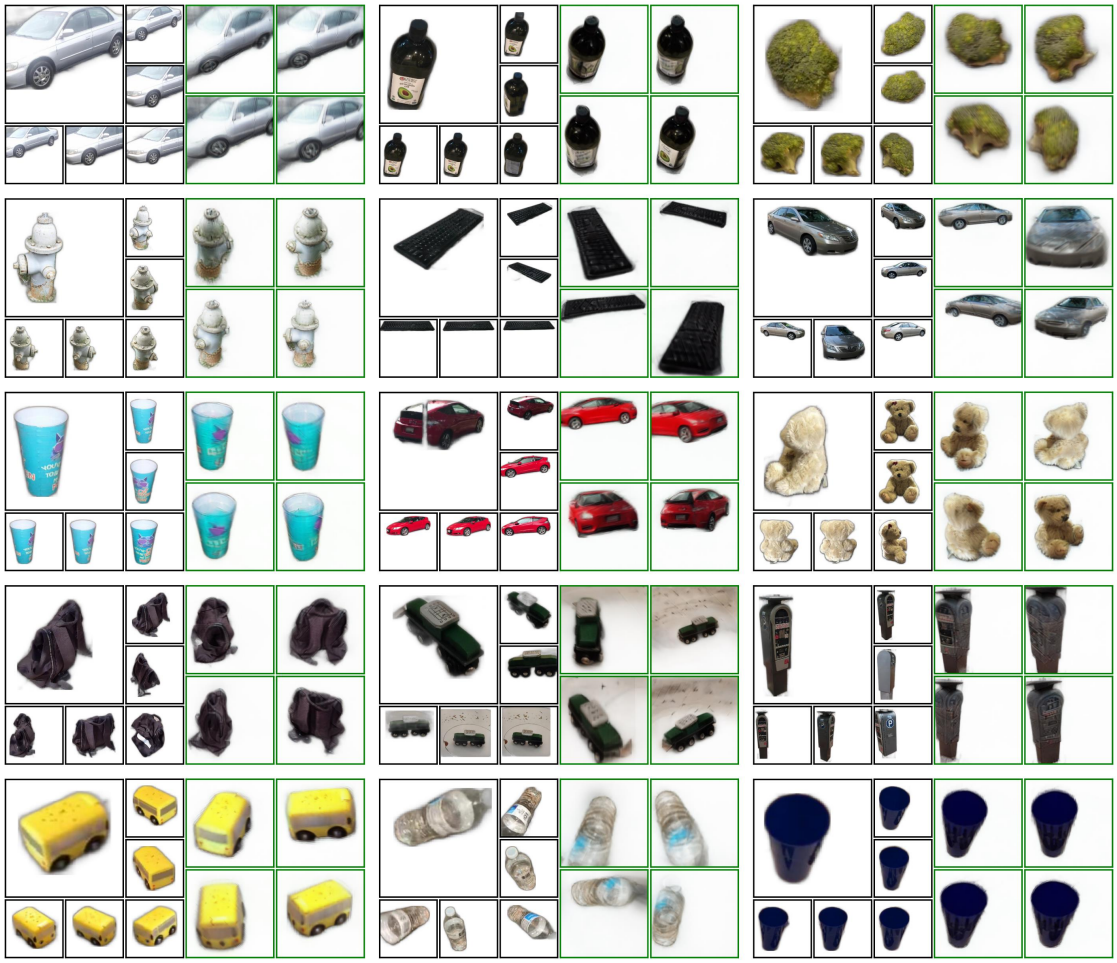

Figure A.3: **Additional results with 6 input view.** We show results for UpFusion (3D) across different object categories given 6 input views (left), and show 4 novel view renderings (right).

Figure A.4: **Additional results for generalization beyond training categories.** We show results for UpFusion (3D) across object categories *not* seen in training. For each instance, we present the 1,3, or 6 unposed input views (left), as well as 4 novel view renderings (right). We observe that despite not being trained on these categories, UpFusion is able to accurately infer the underlying 3D structure and generate detailed novel views.

Appendix B

Implementation and Training Details

B.1 Coordinate Frame

Given N input images of an object, we define a coordinate system $\mathcal{C}$ such that the first input image is 1 unit distance away along the Z-Axis from the origin. Our models consume query pose information that are defined in this coordinate system. Do note that each object in the Co3Dv2 dataset resides in an arbitrary coordinate system with an unknown scale. Hence, to use pose information from such datasets for training, we need to calculate a coordinate system transformation.

To do so, for each object instance during training, we need to identify a point in its existing coordinate system to serve as the origin of the new coordinate system $\mathcal{C}$. We do this in two steps. First, we pick 6 random views from the dataset for that object instance. Then, we solve for a point $\boldsymbol{p}$ that is closest to the rays that travel along the Z-Axis of cameras that are associated with those 6 views. We then define our new coordinate system $\mathcal{C}$ such that the first input image (which is part of those 6 views) is 1 unit distance away along the Z-Axis from this point $\boldsymbol{p}$.

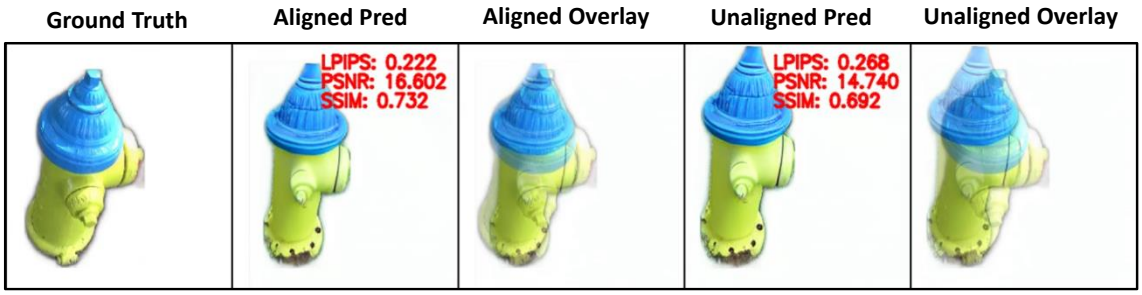

Figure B.1: **Comparison of aligned and unaligned metric.** Conventional image reconstruction metrics are not well-suited to evaluate unposed view synthesis methods due to the inherent ambiguities between coordinate systems. We adopt aligned versions of these metrics by first performing optimized image warping. We illustrate the images and metrics with and without the alignment.

B.2 Evaluating View Synthesis in Unposed Settings

We are interested in evaluating our performance using standard view-synthesis metrics such as PSNR, SSIM, and LPIPS [60]. However, these pixel-aligned metrics are not well suited for evaluating unposed view synthesis due to the fundamental ambiguities between the coordinate systems of the ground-truth and prediction. In particular, given unposed images, there can be an ambiguity up to a similarity transform between the coordinate frames of the reconstruction and prediction. While anchoring the coordinate orientation to the first camera reduces this uncertainty, we still need to consider scaling and shift between predictions and ground truth.

We highlight this issue in Figure B.1, where we observe that despite generally matching the ground truth, the prediction is misaligned in pixel space.

To circumvent this issue, we compute *aligned* versions of the standard image reconstructions metrics (PSNR-A, SSIM-A, and LPIPS-A) by first optimizing for an affine image warping transform $\mathcal{W}_A$ that best matches a predicted image to its corresponding ground truth and then computing the metric. In other words, we evaluate aligned metrics as $\min_{W_A} \mathcal{M}(\mathcal{W}_A(x), y)$, where $\mathcal{M}$ is a metric, x is a predicted image and y is the ground truth image. In practice, for expediency, we compute the optimal transform for minimizing a pixel-wise L2 error instead of computing a

per-metric warp.

B.3 UpSRT

Our UpSRT [38] model architecture mostly follows the original architecture with a few modifications. Unlike the original UpSRT model, which trains a light-weight CNN to extract patch-wise features from images, we use features from a frozen pre-trained DINOv2 [31] model (specifically, the *dinov2_vitb14* model). We leverage the *key* facet from the attention block number 8 to serve as patch-wise features. Our UpSRT transformer architecture has 8 encoder blocks and 4 decoder blocks. Also, we use sinusoidal positional encoding instead of learnable positional encoding for the camera and patch encoding. We also provide information about image intrinsics in the form of additional positional encoding. We trained our model on 41 classes of the the Co3Dv2 dataset (as described in section 4.1) for about 1M optimizer steps on 2 GPUs with a global batch size of 12.

Additionally, UpFusion† models were fine-tuned on Objaverse to allow for fair comparison with single-view baselines. To that end, we fine-tune the UpSRT model which was pre-trained on Co3Dv2 on renderings obtained from the Objaverse dataset for another 1M optimizer steps on 2 GPUs with a global batch size of 12. While performing this fine-tuning, we sample instances from the Objaverse dataset with 90% probability and instances from the Co3Dv2 dataset with 10% probability.

B.4 UpFusion 2D

Model Architecture. Our diffusion model is a modified version of the ControlNet [59] model. In the original architecture, the encoder has two branches, a frozen branch and a trainable branch, both of which incorporate text conditioning information using cross attention. In our architecture, instead of providing text conditioning for the trainable branch, we provide the set latent representation $\boldsymbol{c}_s$. For the frozen branch, we still need to provide a text prompt and we provide the following generic prompt: "*a high quality image with a white background*". Also, in the original architecture, the trainable branch receives an input image $\boldsymbol{c}$ as a conditioning input. In

our architecture, we provide the decoder features $\boldsymbol{c}_d$ as a conditioning input instead.

Adding Query Ray Context. Recall that, in section 3.2, we mentioned that the view-aligned decoder features $\boldsymbol{c}_d$ are generated using a set of rays $\mathcal{R}$. To provide additional context about query rays to the diffusion model, we concatenated Plücker coordinate representation of $\mathcal{R}$ to the view-aligned decoder features to create $\boldsymbol{c}_d$ that is used the condition input by the diffusion model. Similarly, the transformer blocks of the trainable branch of ControNet could benefit from additional query ray context, as it incorporates information from the set-latent features $\boldsymbol{c}_s$ into the U-Net features. To this end, we append Plücker coordinate representation of $\mathcal{R}$ to the U-Net features that are consumed by the transformer blocks.

Classifier-free Guidance. To enable classifier-free guidance [17] during inference, we train our diffusion model in the unconditional mode for a small fraction of the time. During training, similar to [17, 25], we devise a strategy to randomly replace only $\boldsymbol{c}_d$ with the null token Φ_d with 5% probability, only $\boldsymbol{c}_s$ with the null token Φ_s with 5% probability, and both $\boldsymbol{c}_d$ and $\boldsymbol{c}_s$ with Φ_d and Φ_c with 5% probability. The null tokens Φ_c and Φ_s are just zero tensors with the appropriate shape. During inference, we used a guidance weight of 9.0 for all experiments.

Training Details. Following suggestions from the ControlNet codebase, we start training our model with the decoder blocks locked for a few iterations. Then, we resume training with the decoder unlocked. We train our model for about 1M optimizer steps on 2 GPUs with a global batch size of 8. Additionally, UpFusion† models were fine-tuned on Objaverse to allow for fair comparison with single-view baselines. To that end, we fine-tune the diffusion model which was pre-trained on Co3Dv2 on renderings obtained from the Objaverse dataset for another 870k optimizer steps on 2 GPUs with a global batch size of 8. While performing this fine-tuning, we sample instances from the Objaverse dataset with 90% probability and instances from the Co3Dv2 dataset with 10% probability.

B.5 UpFusion 3D

For the multi-step diffusion model sampling, we use the DDIM [43] sampler with 30 steps. Inspired by [50], we use an annealed time schedule for optimizing our NeRF. For the first 300 iterations, we sample time steps corresponding to very high noise to

enable the NeRF to quickly learn coarse level details. Overall, the NeRF is trained for 3000 iterations, which takes a little more than an hour on an A5000 GPU. We use the same regularization losses as SparseFusion [61].

Bibliography

[1] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021. 2.1

[2] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *CVPR*, 2023. 2.1

[3] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 3.4

[4] Eric R. Chan, Koki Nagano, Matthew A. Chan, Alexander W. Bergman, Jeong Joon Park, Axel Levy, Miika Aittala, Shalini De Mello, Tero Karras, and Gordon Wetzstein. GeNVS: Generative novel view synthesis with 3D-aware diffusion models. In *arXiv*, 2023. 2.3, 3

[5] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *ICCV*, 2023. 2.2

[6] Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G Schwing, and Liang-Yan Gui. Sdfusion: Multimodal 3d shape completion, reconstruction, and generation. In *CVPR*, 2023. 2.2

[7] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, 2023. 1, 4.1.1

[8] Congyue Deng, Chiyu Jiang, Charles R Qi, Xinchen Yan, Yin Zhou, Leonidas Guibas, Dragomir Anguelov, et al. Nerdi: Single-view nerf synthesis with language-guided diffusion as general image priors. In *ICCV*, 2023. 2.2

[9] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B McHugh, and Vincent Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items. In *2022 International Conference on Robotics and Automation (ICRA)*, 2022. 1, 4.1.1,

4.1.2

[10] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, 2017. 2.2

[11] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022. 2.1

[12] Yang Fu, Ishan Misra, and Xiaolong Wang. Mononerf: Learning generalizable nerfs from monocular videos without camera poses. In *ICML*, 2023. 2.1

[13] Yasutaka Furukawa, Carlos Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2015. 2.1

[14] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016. 2.2

[15] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *ICCV*, 2019. 2.2

[16] Yuan-Chen Guo, Ying-Tian Liu, Ruizhi Shao, Christian Laforte, Vikram Voleti, Guan Luo, Chia-Hao Chen, Zi-Xin Zou, Chen Wang, Yan-Pei Cao, and Song-Hai Zhang. threestudio: A unified framework for 3d content generation. `https://github.com/threestudio-project/threestudio`, 2023. 4.1.2

[17] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. 3.4, B.4

[18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 3, 3.1.2

[19] Hanwen Jiang, Zhenyu Jiang, Kristen Grauman, and Yuke Zhu. Few-view object reconstruction with unknown categories and camera poses. *3DV*, 2024. 2.3, 4.1.2

[20] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 2023. 1

[21] Jonáš Kulhánek, Erik Derner, Torsten Sattler, and Robert Babuška. Viewformer: Nerf-free neural rendering from few images using transformers. In *ECCV*, 2022. 2.3

[22] Amy Lin, Jason Y Zhang, Deva Ramanan, and Shubham Tulsiani. Relpose++: Recovering 6d poses from sparse-view observations. In *3DV*, 2024. 4.1.1, 4.1.2

[23] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *ICCV*, 2021. 2.1

[24] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Zexiang Xu, Hao Su, et al. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. In *NeurIPS*, 2023. 1, 2.2, 4.1.2

[25] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *ICCV*, 2023. 1, 2.2, 3.4, 4.1.2, B.4

[26] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Learning to generate multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023. 2.2

[27] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *ICCV*, 2023. 2.2

[28] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *CVPR*, 2019. 2.2

[29] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2.1

[30] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 2022. 2.1, 3, 3.4

[31] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. B.3

[32] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*, 2022. 1, 2.2, 3.3

[33] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, et al. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. *arXiv preprint arXiv:2306.17843*, 2023. 2.2

[34] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *ICCV*, 2021. 1, 2.3, 4.1.1

[35] Robin Rombach, Patrick Esser, and Björn Ommer. Geometry-free view synthesis: Transformers and no 3d priors. In *ICCV*, 2021. 2.3

[36] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn

Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 1, 2.2, 3.2

[37] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 2022. 2.2

[38] Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lucic, Daniel Duckworth, Alexey Dosovitskiy, Jakob Uszkoreit, Thomas Funkhouser, and Andrea Tagliasacchi. Scene Representation Transformer: Geometry-Free Novel View Synthesis Through Set-Latent Scene Representations. *CVPR*, 2022. (document), 1, 2.3, 3, 3.1, 3.1.1, 4.1.2, B.3

[39] Mehdi SM Sajjadi, Aravindh Mahendran, Thomas Kipf, Etienne Pot, Daniel Duckworth, Mario Lučić, and Klaus Greff. Rust: Latent neural scene representations from unposed imagery. In *CVPR*, 2023. 2.3

[40] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 2.1

[41] Samarth Sinha, Jason Y Zhang, Andrea Tagliasacchi, Igor Gilitschenski, and David B Lindell. Sparsepose: Sparse-view camera pose regression and refinement. In *CVPR*, 2023. 2.3

[42] Cameron Smith, Yilun Du, Ayush Tewari, and Vincent Sitzmann. Flowcam: Training generalizable 3d radiance fields without camera poses via pixel-aligned scene flow. In *NeurIPS*, 2023. 2.3

[43] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2020. 2.2, 3.4, B.5

[44] Jiaxiang Tang. Torch-ngp: a pytorch implementation of instant-ngp, 2022. https://github.com/ashawkey/torch-ngp. 3, 3.4

[45] Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior. In *ICCV*, 2023. 2.2

[46] Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *CVPR*, 2023. 2.2, 4.1.2

[47] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018. 2.2

[48] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for

multi-view reconstruction. *NeurIPS*, 2021. 2.1

[49] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, 2021. 2.3

[50] 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. 2.2, 3.4, B.5

[51] Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Yi Wang, and Zhangyang Wang. Cvpr. 2023. 2.2

[52] Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *NeurIPS*, 2019. 2.2

[53] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, 2018. 1

[54] Lior Yariv, Yoni Kasten, Dror Moran, Meirav Galun, Matan Atzmon, Basri Ronen, and Yaron Lipman. Multiview neural surface reconstruction by disentangling geometry and appearance. *NeurIPS*, 2020. 2.1

[55] Yufei Ye, Shubham Tulsiani, and Abhinav Gupta. Shelf-supervised mesh prediction in the wild. In *CVPR*, 2021. 2.2

[56] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *CVPR*, 2021. 2.3, 3

[57] Zehao Yu and Shenghua Gao. Fast-mvsnet: Sparse-to-dense multi-view stereo with learned propagation and gauss-newton refinement. In *CVPR*, 2020. 1

[58] Jason Y Zhang, Deva Ramanan, and Shubham Tulsiani. Relpose: Predicting probabilistic relative rotation for single objects in the wild. In *ECCV*, 2022. 2.3, 4.1.1

[59] Lvmin Zhang and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. 2023. 1, 3.2, B.4

[60] 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. B.2

[61] Zhizhuo Zhou and Shubham Tulsiani. Sparsefusion: Distilling view-conditioned diffusion for 3d reconstruction. In *CVPR*, 2023. 1, 2.3, 3, 3.3, 3.3, 3.4, 4.1.2, B.5

[62] Zi-Xin Zou, Weihao Cheng, Yan-Pei Cao, Shi-Sheng Huang, Ying Shan, and Song-Hai Zhang. Sparse3d: Distilling multiview-consistent diffusion for object reconstruction from sparse views. *arXiv preprint arXiv:2308.14078*, 2023. 1