Probabilistic Inverse Cameras: Image to 3D via Multiview Geometry

Rishabh Kabra[1,2], **Drew A. Hudson**[1], **Sjoerd van Steenkiste**[1], **Joao Carreira**[1], **Niloy J. Mitra**[2]

{rkabra, dorarad, svansteenkiste, joaoluis}@google.com, n.mitra@cs.ucl.ac.uk

[1]Google DeepMind, [2]University College London

https://unpic.github.io

Abstract

We introduce a hierarchical probabilistic approach to go from a 2D image to multiview 3D: a diffusion "prior" models the unseen 3D geometry, which then conditions a diffusion "decoder" to generate novel views of the subject. We use a pointmap-based geometric representation in a multiview image format to coordinate the generation of multiple target views simultaneously. We facilitate correspondence between views by assuming fixed target camera poses relative to the source camera, and constructing a predictable distribution of geometric features per target. Our modular, geometry-driven approach to novel-view synthesis (called "unPIC") beats SoTA baselines such as CAT3D and One-2-3-45 on held-out objects from ObjaverseXL, as well as real-world objects ranging from Google Scanned Objects, Amazon Berkeley Objects, to the Digital Twin Catalog.

1. Introduction

> "One does not simply reason about shapes at the level of pixels."
>
> — *Anon*

Recovering 3D geometry from a single image is a hard, underspecified problem [9]. SoTA methods for novel view synthesis (NVS) find it liberating to be geometry-free: they use a given image to predict novel views of the subject directly, before reconstructing any 3D [8, 23, 26]. Methods that attempt, on the other hand, to produce and use a geometric explanation (i.e., a geometric *prior*) are often bottlenecked by per-scene optimization [31, 42]. Else they lack the ability to imagine multiple plausible 3D scenarios [13, 48]. They may in fact rely on class labels and semantic cues [24, 48], or on known correspondences between representations and views [13], thus reducing their generalizability. One might be tempted to conclude that geometric priors distract from and impede the progress of data-driven image-to-image NVS. We show this need not be the case.

We introduce a modular framework to facilitate geometry-based NVS. The generation task is decomposed into two: first use a given image to predict multiview features, then use the multiview features (and original input) to generate the corresponding target images. Our approach is analogous to the unCLIP approach behind DALL-E 2 [33] for text-to-2D synthesis: they first map CLIP text embeddings to image embeddings using a probabilistic *prior*, then *decode* the image embeddings to pixels. Since 2D to 3D from a single image is also a one-to-many mapping, modeling it as the composition of two probabilistic maps can improve the range and accuracy of the realized outputs. Inspired by unCLIP, our hierarchical approach is called "unPIC" (undo-a-picture with Probabilistic Inverse Cameras).

To specify the target shape of an object before filling in appearance-level details, we use a pointmap-based intermediate representation. We adapt the idea of a Normalized Object Coordinate Space (NOCS [39]), a function that maps every point on an object's surface to a unique RGB color based on its spatial coordinates. When rendered as images (aka pointmaps), NOCS establishes point-to-point correspondence across viewpoints (Fig. 1). It is naturally suited to help coordinate the generation of consistent multiview target images. We introduce a version of NOCS that exploits an available degree of freedom—the orientation of the color space—to make the pointmaps more predictable for the prior module. Our version (called "CROCS") allows the prior to make camera-pose-invariant predictions.

Ultimately, our goal is to produce realistic and diverse shape and appearance completions from a single 2D image. To motivate our design, we present the following experiments: we first show that a diffusion decoder achieves better NVS when equipped with ground-truth NOCS pointmaps. Next, we train a diffusion prior to predict multiview pointmaps from a given source image. Our camera-relative version of NOCS significantly improves the diffusion prior's performance, achieving a 4x boost in target shape predictability. Finally, we stack the separately trained prior and decoder for hierarchical NVS from a single image. We beat the SoTA baselines CAT3D, One-2-3-45, and OpenLRM on appearance and shape reconstruction,
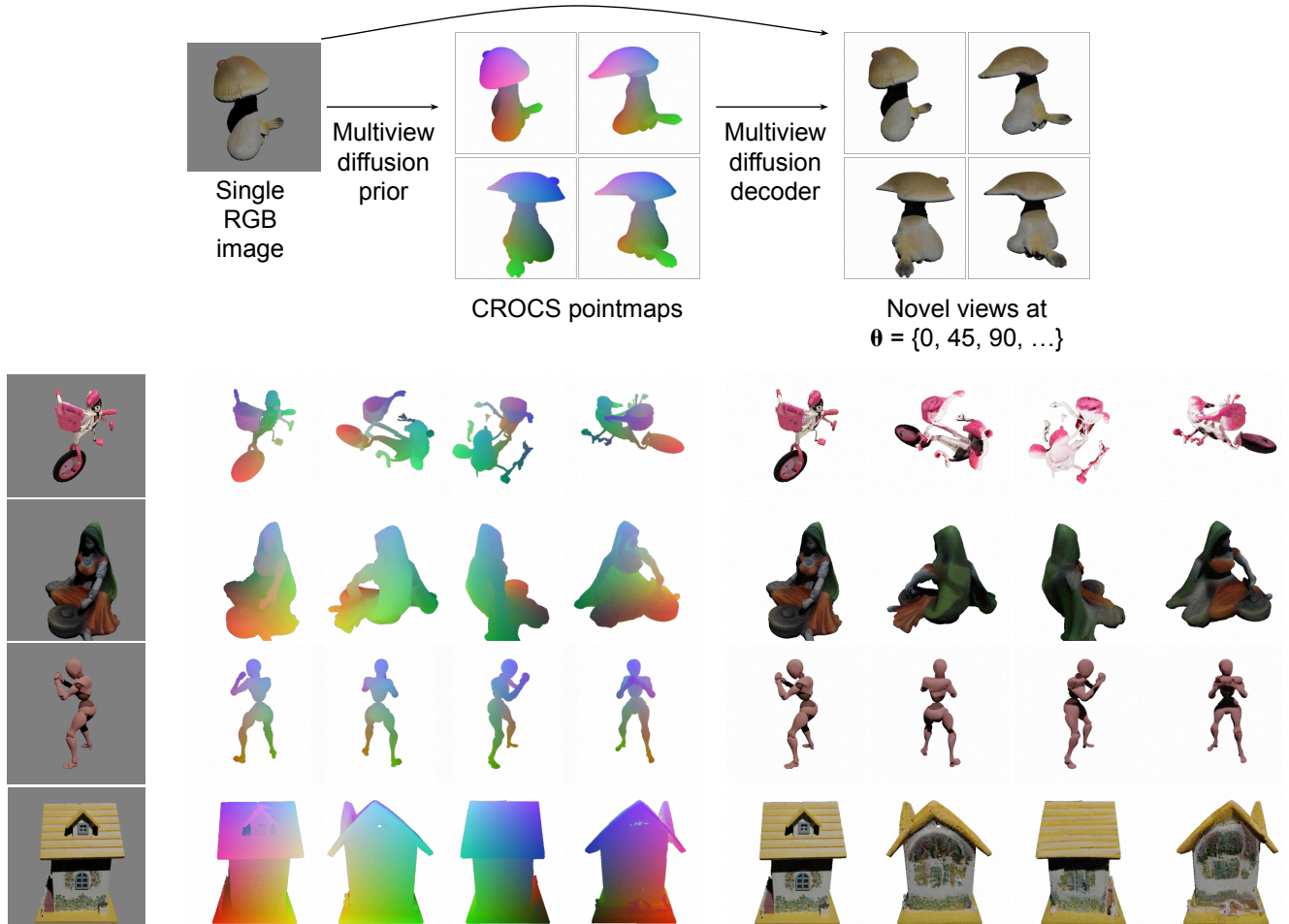
1

Figure 1. **Top:** A hierarchical approach to NVS. A prior models multi-view features from a single image. These are jointly decoded to the target novel-view images. Our choice of intermediate features establishes point-to-point correspondence across views. **Bottom**: More examples from the prior and decoder. Our model exhibits transferrable shape understanding having never seen a real-world pixel.

demonstrating our method's superior shape understanding and the viability of geometry-based multiview modeling.

2. Related Work

Novel view synthesis: Optimization-based NVS methods (such as Neural Radiance Fields [25] or Gaussian Splatting [20]) fit a geometric representation to a specific scene. Such methods need several consistent input images to be able to interpolate between camera poses. When only one image is available, we need a learned prior to predict what an object/scene looks like from alternative views. One line of work builds on spatially arranged implicit representations such as convolutional feature grids in pixelNeRF [48] or attention-based triplane tokens in LRM [13] to enable feed-forward inference. These original methods are deterministic, thus suffering from averaged/blurry reconstructions, and inconsistent outputs from different views of the same

object. Probabilistic extensions such as DMV3D [47] help instantiate multiple versions of an object from a single image. In a similar vein, methods such as DreamFusion [31], Score Jacobian Chaining [40], Zero-1-to-3 [24], and ZeroNVS [35] use diffusion-based priors to iterate between novel view synthesis and 3D reconstruction. Such methods tend to be slow due to scene-specific optimization (typically NeRF- or SDF-based). They also suffer from issues due to bootstrapping from pretrained text-to-image models. More recently, it has become common to generate multiple novel views *before* reconstructing 3D. EscherNet [21], ReconFusion [44], and CAT3D [8] all take this geometry-free approach. And while diffusion-based approaches like SODA's latent bottleneck [16] and 3DiM's stochastic conditioning [43] help reduce inconsistencies in generating novel views, there are no guarantees they are consistent with a feasible object.

2

Geometry-conditioned generative models: ControlNet [50] popularized the use of geometric hints to condition and guide a diffusion model. Other applications of geometry conditioning include motion brush animation [28] and pose-conditioned multi-object generation [17, 45]. This approach is especially tempting as diffusion models can be trained with conditioning dropout to enable classifier-free guidance [11], i.e., model a conditional and unconditional distribution simultaneously. At inference time, the model can cope with not having any geometric information, and still sample from the unconditional or source image-conditioned distribution. Few prior works (with the prominent exception of Motion-I2V [37]) have explored sampling geometric features using a diffusion prior, and feeding them to a decoder in place of true or provided features. We show this is not only possible, but desirable: it ensures the geometric conditioning is realistic for the given example, potentially diverse (e.g., supports sampling of multiple different 3D shapes), and obtainable without human effort.

Pointmaps: Point-based representations are ubiquitous in 3D modeling. For instance, point-set registration (determining the correspondence between two point clouds) is a classic task in 3D scanning. Recently, DUSt3R [41] introduced a novel approach to 3D reconstruction, integrating pairs of images based on a regression of pointmaps without using any camera information. DUSt3R and our work share two similarities: **1)** DUSt3R estimates pointmaps for two images at once. Likewise we model 8 pointmaps simultaneously to synchronize between them. **2)** To handle scale ambiguity, DUSt3R normalizes pointmaps using their norm. This is similar to the scale normalization performed by NOCS. Two key differences between DUSt3R and our work are as follows: **a)** We use a diffusion-based probabilistic model to handle the underconstrained nature of our problem. **b)** In DUSt3R, pointmaps are initially expressed in pairwise reference frames. These need to be aligned globally via optimization. In contrast, we express all pointmaps in a shared reference frame (i.e., CROCS color cube) from the outset. We go one step further—we set the orientation of the color cube based on the camera pose of the source image in each example. This makes the coloring predictable from different viewpoints, and learnable across examples.

Diffusion-based geometric priors: Pretrained diffusion models been shown to yield latents useful for various vision tasks including depth estimation and semantic correspondence [7]. They have also been successfully trained to output dense annotations such as depth and surface normals [10]. All this portends well for our attempt to predict pointmaps annotating the source and target images. In fact, one recent work (SpaRP [46]) did use a diffusion model to predict NOCS. But they only predict NOCS for a sparse set of input images to infer their correspondence (rather than use it to model unseen geometry at novel views).

3. Method

3.1. A Probabilistic Multiview Framework

unPIC is a hierarchical diffusion model that converts a 2D image into multiview 3D. Given a single view of an object or scene, unPIC comprises (1) a diffusion *prior* that infers representations of the subject's multiview geometry or appearance. This inference step combines the task of annotating the source image and predicting representations for novel views as well. (2) A diffusion decoder that transforms all predicted annotations into corresponding novel views of the subject. We make the modeling task easier by assuming the desired views are regularly spaced and the camera trajectory is closed; hence the prior can be seen as predicting a representation loop.

Annotating the source image alone may be a deterministic step (e.g., if using features from a deterministic encoder-decoder system). In contrast, predicting novel-view features is inherently non-deterministic, due to the partial information observed in a single image (see Fig. 2). There may be several valid loops that go through the same point that describes the source image. Hence, a probabilistic prior is essential, as opposed to the bulk of prior work [13, 18, 48].
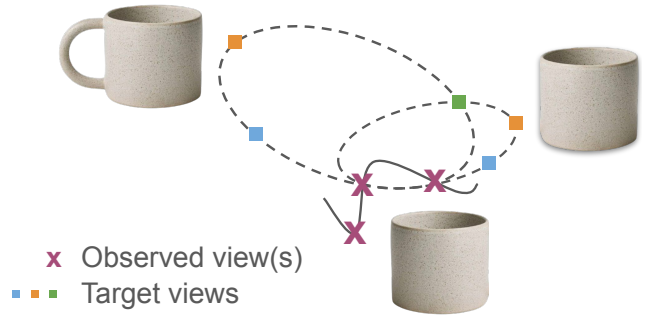

Figure 2. **Schrödinger's cup: two sets of valid novel views, following different trajectories in representation space.** To model diverse trajectories without losing view consistency, we use the following ingredients: (a) equidistant target-camera poses on a circular trajectory, (b) intermediate representations with cyclic structure (CROCS), and (c) an architectural inductive bias, namely, convolutions on a grid of images tiled in circular order.

Depending on the choice of representation, the target annotations may vary haphazardly or not vary at all (e.g., global semantic features such as the class) as a function of the camera pose. A suitable representation can ensure the loops are well structured and predictable. This increases the benefit of modeling multiple views simultaneously (see our empirical results in Tab. 2), and is perhaps key to encouraging view consistency.

As for the decoder module—which takes intermediate representations and maps them to images—that could be either (a) deterministic or stochastic; (b) separately conditioned on single views or jointly on multiviews. Our choice of representation (CROCS) captures the geometry but leaves the texture of the subject entirely unknown. Hence we use a probabilistic model again.

3.2. Deterministic Target-Camera Poses

Our target camera poses are defined relative to the source camera pose. We predict novel views at K fixed camera poses for a given image. These correspond to rotations $\theta_{target_k} = 2\pi(k-1)/K$ of the object about its vertical axis, which remains aligned with the vertical axis of the world. More generally, one can also vary the camera elevation angle ϕ_k (as in [46]), but we do not unless mentioned otherwise. Including the source camera pose as one of the targets, $\theta_{target_1} = 0$, helps anchor the model's predictions of novel views. The advantage of defining target poses relative to the source camera is that our model is pose-free thereafter: by assuming regularly-spaced camera poses, we can avoid providing them to our model in an explicit form.

Note that using deterministic camera poses does not equate to a fixed multi-camera rig (as in [1]). Rather, we allow the source camera to vary freely (on hemispheres of arbitrary size) around the object. Given a source camera, the target cameras are at the same height as the source, the same distance to the object, but at deterministic rotations around the object's vertical axis, and oriented to face the object.

3.3. Camera-Relative (Normalized-) Object Coordinates

We predict the geometry of an object before decoding its multiview appearance. To do so, we rely on NOCS, a scale-free representation of geometry. The object/scene is first uniformly scaled to fit a 3D unit cube, so all spatial coordinates lie in $[0, 1]$. These can therefore be interpreted as RGB colors and rendered as images, which maps neatly onto our multiview framework. While one NOCS image serves as a dense image annotation, a set of NOCS images forms a point cloud.

NOCS on its own does not provide a predictable coloring of the object's surface—any rotation of the RGB color space would still produce a valid coloring of a fixed object. We introduce a version of NOCS that is canonicalized to paint an object relative to the source camera pose (see Fig. 3). To track the source camera through a yaw rotation of θ about the vertical axis, we apply an SO2 rotation to the Red and Green channels—which span the NOCS ground plane—of all pointmaps for the given object. Note that this destroys any pose relations across objects (our training data does not, for instance, paint all cups' handles with a consistent color).

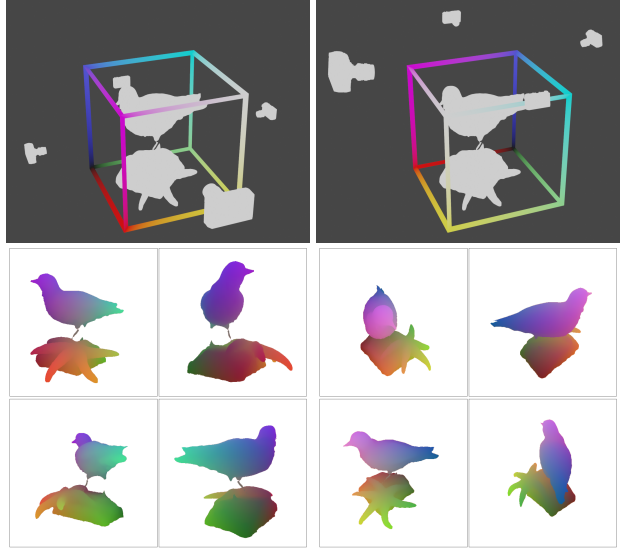We address two geometric subtleties in working with

Figure 3. **Camera-Relative Object Coordinate Spaces.** We show two data-points (Left and Right columns) obtained from one object. **Top-left**: The wireframe shows the RGB reference cube used to paint the object surface. The large camera denotes the source view, whereas the smaller cameras denote (3 of 7) novel views. **Top-right:** Say all camera locations are rotated by $\theta = 120$ degrees around the vertical axis (the object stays fixed). Then we also rotate the color reference cube by the same degree. This ensures each camera faces the same side of the cube that it was facing prior to the cameras' rotation. **Bottom:** Target CROCS images in clockwise order corresponding to the cameras above. In a given data-point, any part of the object is consistently colored across target images. Across data-points, although a given part of the object may change colors, its color is predictable based on its location(s) in the target image(s). Each target view has a consistent color bias that is learnable across examples.

CROCS in Appendix B—the first one is about renormalizing pointmaps after a change of reference frame to ensure they are still bounded in $[0, 1]$. The second one explains our choice not to canonicalize CROCS colors with respect to varying camera elevations ϕ.

Our CROCS representation diverges from the typical NOCS formulation [22, 39] a second way: when dealing with a multi-object scene, we treat it as a single object. This is better suited for our task, as we care less about capturing the pose of individual objects, but more about the spatial arrangement and relative position of different objects.

3.4. Dataset

We train on a combination of object and scene assets from Objaverse [3] and the alignment finetuning subset of ObjaverseXL [4] . We holdout 50k randomly sampled objects from ObjaverseXL for evaluation. For each asset, we pre-render images from a dense grid of camera poses (at various θ, ϕ, and r; see Sec. 3.2). We also pre-render one NOCS

pointmap per image using a fixed NOCS RGB reference frame. At training time, we adapt the pre-rendered NOCS pointmaps to CROCS on demand based on which camera is chosen as the source, using rotations of the RGB space as mentioned in Sec. 3.3.

When rendering images, we add an elevated point light behind the camera. We ensure it stays fixed relative to the camera, rather than fixed in the scene. This ensures the model can treat each image the same in terms of lighting, and cannot use lighting cues. (Note: a small number of assets have preexisting lights. We do not remove these.).

3.5. Diffusion Model

We use a standard pixel-based Diffusion Denoising setup [12]. The prior and decoder models use an identical architecture, broadly following the UNet from [8]. We train the two models independently to maximize the geometric diversity of the prior—else it would be forced to output the most optimal representations for the decoder's loss (e.g., hacks to copy bits of texture from the source image). Moreover, we train the models from scratch, without relying on pretrained weights, to avoid ambiguity about where the models gain their representational and reconstruction abilities.

To denoise $K = 8$ multiview images simultaneously, we find that tiling them to form a superimage works best. We tile clockwise along 4 columns and 2 row (see Fig 3) — this ensures that adjacent tiles always correspond to adjacent camera poses in 3D. The source image is also provided to the model as a superimage, but with 7 of 8 tiles remaining empty. For $K = 4$, the superimage grid size is 2x2.

We run initial experiments (Secs 4.1 and 4.2) at base image size 64x64, predicting either 1, 4, or 8 targets. We use a standard cosine diffusion schedule [27] to train these models. Our final model works on 256x256 base images, predicting 8 targets at a superimage resolution of 1024x512. To scale to this size, we use the interpolated, logSNR-offset schedule from Eq. 6 of [14], which is necessary for diffusion to overcome the correlations at this resolution.

We train both models with a conditioning dropout probability of 0.05. This opens the possibility of using classifier-free guidance (CFG). We show the effect of different CFG weights in the Appendix. For all other evaluations in the paper, we use a fixed CFG value of 2.0. The prior model is normally conditioned on the timestep and the source image. We apply CFG to the source image. The decoder model is additionally conditioned on the prior's output CROCS. We only apply CFG to the source image, as the decoder tracks CROCS pointmaps closely even without CFG.

4. Experiments

4.1. Diffusion Decoder

To motivate geometric representations and the use of CROCS specifically, we run an experiment to decode a single novel view of an object at a fixed rotation ($\theta = 90$ degrees). We train diffusion models that take a source image and an annotation of the target image to predict the target view. The annotations range from geometric (like depth maps and CROCS), image-space (alpha masks), through feature maps (from DINOv2 [29] and CLIP [32]). All annotations are treated as images—they are upscaled (if required) and concatenated to the input latent before it is fed to the UNet for denoising. We run these experiments at 64x64 image size to allow multiple seeds.

We find in Tab. 1 that geometric annotations consistently outperform other choices. Given a source image, there is more ambiguity about the target shape than its texture or appearance, explaining the empirical results. We further see one of the benefits of CROCS over NOCS: its consistently colored maps make it a more useful condition than arbitrary colored ones. CROCS's full superiority is revealed when we try to *estimate* both CROCS and NOCS in Sec 4.2.

Multiple targets. To examine the benefit of geometric features for multiview prediction, we run experiments with and without ground-truth annotations while predicting 4 or 8 target views. The target superimage grids' dimensions are 128x128 and 256x128, respectively. We carefully control the diffusion architecture, applying attention at two downsampled resolutions in each case (at heights 16 and 8).

In Tab. 2, we echo the findings in [21] that predicting more views simultaneously (e.g., $K = 4$) improves metrics rather than predicting just one. But in the case where only the unannotated source image is available, the effect tapers off and reverses from $K = 4$ to $K = 8$ views. On the other hand, we see a 5x improvement in the MSE by conditioning on multiview CROCS pointmaps, and we see increasing benefits up to $K = 8$. This is remarkable especially because the set of self-attention tokens doubles with the number of views, potentially making it harder to model long-range spatial relationships (and perhaps leading to the drop we see in the unconditional case). These results help establish the advantage of (ground-truth) CROCS annotations in coordinating multiview synthesis.

4.2. Diffusion Prior

Having used ground-truth representations in Sec. 4.1, we now move to the question of estimating multiview geometric representations from a single source image. We train a diffusion prior to predict pointmaps at 8 target views. We compare two choices of representations for how predictable they are from arbitrary source views: (a) NOCS images in a static reference frame, and (b) CROCS, where we change

Table 1. **Single-target diffusion decoder experiments.** We predict a fixed 90-degree object rotation from a given source image, with the help of a ground-truth annotation of the target image. We report the pixel-space Mean Square Error on random source views across 10k heldout objects. Stds. are across 3 training runs.

Annotation Type	MSE (1e-3)
Source image only	19.042 ± 0.722
+ CLIP ViT L/16 (24, 24, 1024)	6.059 ± 0.448
+ DINOv2 B/14 (16, 16, 768)	5.734 ± 0.815
+ Alpha mask	8.169 ± 0.392
+ Depth map (from NOCS)	5.580 ± 0.849
+ NOCS	5.318 ± 0.070
+ CROCS	$\mathbf{4.914 \pm 0.224}$

Table 2. **Multiple-target diffusion decoder experiments** using ground-truth annotations. We report the pixel MSE (1e-3) on $K-1$ novel views as in Table 1. Stds. are across 5 training runs.

Annotation Type	K=4 views	K=8 views
Source image only	4.739 ± 0.215	5.817 ± 0.230
+ CROCS (K views)	1.065 ± 0.064	$\mathbf{1.039 \pm 1.331}$

Table 3. **Diffusion prior experiments.** We predict pointmaps at 4 and 8 target views, and report the pointmap MSE (1e-3).

Annotation Type	K=4 views	K=8 views
NOCS	11.94	15.82
CROCS	**1.21**	**3.92**

the reference frame based on the source camera.

We find a 4x increase in predictability from using CROCS (see Tab. 3). Without CROCS, choosing a random source camera leaves the colors of the target images unpredictable because we do not condition the model on any camera extrinsics. With CROCS, each target pose has a biased distribution of colors, consistent across objects. This makes the geometric feature space unambiguous, and each representation loop more predictable, in turn freeing up the model's probabilistic capacity for the ambiguous task of predicting unseen object/scene geometry.

4.3. Comparison with Other Methods

We now turn to our primary goal of improving novel view synthesis, especially shape prediction, relative to SoTA baselines. We compare against the following methods:

- **OpenLRM** (288x288): a geometry-aware but deterministic method based on the original LRM [13]. It uses a Transformer to convert DINO features into implicit triplane representations of 3D shape. The triplane tokens are spatially indexed/queried to parameterize a NeRF model of the scene, mitigating view consistency issues and allowing image-generation at arbitrary camera poses.
- **One-2-3-45** (256x256): a SoTA version of Zero-1-to-3

Table 4. Comparisons with baselines on 4 datasets.

	PSNR ↑	IoU ↑	FID ↓	LPIPS ↓
ObjXL				
unPIC (ours)	**23.86**	**0.79**	109.77	0.48
CAT3D	22.10	0.63	84.49	**0.23**
OpenLRM	13.86	0.62	103.20	**0.23**
One-2-3-45	12.42	0.58	**67.89**	0.25
GSO				
unPIC (ours)	**23.93**	**0.89**	105.98	0.48
CAT3D	23.50	0.68	77.67	**0.23**
OpenLRM	13.82	0.67	101.44	0.25
One-2-3-45	13.16	0.65	**57.08**	0.25
ABO				
unPIC (ours)	**26.05**	**0.87**	83.63	0.44
CAT3D	22.02	0.64	70.20	**0.26**
OpenLRM	11.62	0.59	114.64	0.29
One-2-3-45	11.41	0.61	**47.81**	0.27
DTC				
unPIC (ours)	**26.26**	**0.91**	123.94	0.44
CAT3D	23.11	0.74	68.49	**0.20**
OpenLRM	14.34	0.71	88.12	0.22
One-2-3-45	14.30	0.74	**48.54**	**0.20**

[24], available as open-source. Its diffusion-based image prior takes camera rotation and translation (R and T) parameters to transform a given image. Each novel view is sampled individually, leaving the outputs prone to multiview inconsistencies. We focus on evaluating the geometry-free stage, feeding R and T parameters to match our target poses (4 of which are identical), but using the model's own estimation of the camera height.

- **CAT3D** (512x512): a SoTA-quality, geometry-free diffusion model trained using masked modeling of views on diverse datasets. While the original model is closed source, we received a Colab and checkpoint from the authors. The model uses a coarse view-sampling strategy to generate 7 anchor views (given 1 source image). Subsequently, it conditions on the anchor views to generate a finer set of views using the same model. We focus on the coarse stage for our evaluation.

Compared to our approach, these methods all have the advantage of starting from a pretrained model (e.g., Zero-1-to-3 starts from Stable Diffusion, CAT3D starts from a pretrained Latent Diffusion Model [34], and OpenLRM bootstraps from DINO). Some of them are further trained on real scenes (e.g., MVImgNet [49]) in addition to digital assets. However, in this work, we choose to demonstrate how far we can get with synthetic data alone. We find that identifying test datasets for the baselines was also a challenge – all available datasets tend to be used for training purposes. Though we held out Google Scanned Objects (GSO [6]) and Amazon Berkeley Object (ABO [2]) for our model, some

Figure 4. **Qualitative comparison.** One-2-3-45 produces multiview inconsistencies. CAT3D can squash the shapes in unseen views.

of the baselines seem likely to have been trained on them. One-2-3-45 was certainly trained on ObjaverseXL (we use the standard Zero123-XL checkpoint). Due it its recency, the Digital Twin Catalog (DTC [30]) is one candidate that may have not been seen by any of the models.

We run all models, including ours, on A100 GPUs, and downsize all images to 256x256 for comparison. We focus on evaluating (a) reconstruction and (b) perceptual quality. When computing metrics, we sample a random target from the set of 7 novel views for each example. This ensures the computed statistics are independent (particularly because the FID relies on second-order statistics). We do not repeat any object (as in training) with an alternative source view.

Results. Tab. 4 shows that unPIC beats all baselines on PSNR, and outperforms them substantially on the shape-focused IoU metric. We examine this advantage qualitatively in Fig. 5. Although all methods tend to produce plausible looking images, the generated poses and shapes can be

completely arbitrary. Being trained on Objaverse and ObjaverseXL alone, unPIC still manages to achieve reasonable perceptual metrics (FID and LPIPS) compared to the baselines, which all rely on pre-trained weights/features.

We find a common failure mode that affects shape and pose prediction for CAT3D and OpenLRM: despite keeping a constant camera height while generating surrounding views, OpenLRM appears to level and raise the camera (w.r.t. the ground plane) as it moves around the object (see Fig. 2 in the LRM paper [13]). The issue appears consistently in OpenLRM outputs, and intermittently in CAT3D's as well. It may stem from how the training data was generated for those models, e.g., if they oriented the object randomly before capturing images. This is especially undesirable for assets placed on a planar surface, as the surface appears to tilt while the camera orbits the object. In our case, we avoid this issue by ensuring the object's vertical axis is always aligned with the world's.

Such camera-control failures are perhaps compounded in OpenLRM and CAT3D due to a failure to estimate the camera height correctly, an issue that One-2-3-45 and unPIC handle robustly. Since CROCS does not adjust for the camera height, it leads to distinct color modes when objects are observed, for instance, from side views or top-down views. This forces unPIC's prior to estimate the camera height in order to predict the right color mode.

5. Discussion

We focus on generating novel views that span an azimuthal rotation of an object. These sparse views can serve as "anchors" for downstream generation of further views from a denser range of camera poses, an arbitrary target pose, or an explicit 3D reconstruction. Our pipeline can thus precede and support a large number of existing methods/models. We aimed to ensure multiview consistency among our outputs to ensure downstream reconstruction tasks are well-posed.

There are several advantages to our hierarchical approach: (a) it allows maximizing the size of the prior and decoder models, as they can be trained separately; (b) we
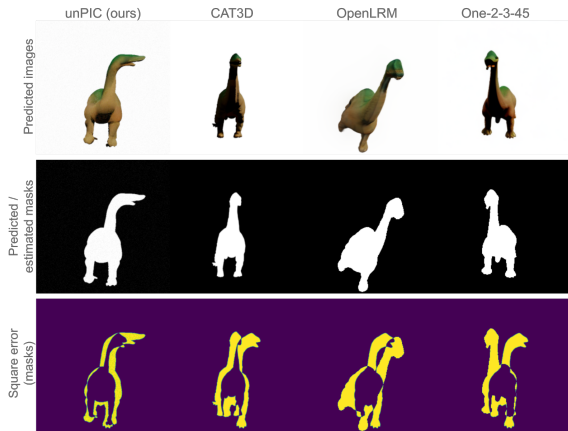

Figure 5. While all models produce plausible images, their shapes and poses can be off. We compare masks with the ground-truth.

Figure 6. **Diversity of unPIC outputs** at a particular novel view (90-degree rotation) from an ambiguous image (top-left). We show the first samples without cherry-picking.

can visualize and interpret the intermediate features—for instance, visualizing CROCS would reveal whether 3D reconstruction errors arise due to geometry or texturing; (c) it prevents a collapse of output diversity in the prior—this would be likelier if the modules were cotrained, as the prior would be spurred to output only the optimal intermediate representation for the final RGB loss; (d) in the multiview case, unPIC allows expressing relationships between views at a higher level of abstraction than pixels.

Our pointmap-based representation is both capable of coordinating multiview synthesis, but also inherently predictable by construction. By training a prior to predict CROCS annotations from a given image, we can achieve diverse and valid instantiations of 3D shapes (see Fig 6 for an example). More broadly, modeling geometric features as opposed to conditioning on them directly in generative models eases the problem of controllability when the desired controls are tedious to express.

Limitations & Future work. A salient shortcoming is that we do not model scene backgrounds. This may be feasible using CROCS for room-like scenes where the background geometry is uniform, and can be scaled to a unit cube. We are currently also limited to training on images of 3D assets rendered at controllable poses, rather than in-the-wild images. This could be eased by using pretrained 3D models (e.g., NeRFs) of real scenes to render images and depth maps, which then could be used to estimate CROCS.

A straightforward extension of our study would be to allow multiple random source views rather than one. One of them could be designated the primary view. The remaining "reference" views could optionally be annotated with camera poses to avoid correspondence ambiguity.

The unPIC framework is not limited to one prior, e.g., it could be fruitful to predict multiview texture features to condition the decoder. Or it may also be possible to enhance multi-object handling by breaking CROCS into two components: a scene-level pointmap like ours (that colors objects based on their positions in the scene), and a pointmap that colors objects individually, independent of their position.

Conclusion. We introduced the unPIC framework for multiview synthesis from a single image via multiview features. One choice of intermediate features, based on NOCS, encodes scale-free geometry and point-to-point correspondence across views. We showed it is an effective choice over several geometric and non-geometric alternatives. We introduced a version of NOCS called CROCS that is predictable irrespective of the camera pose of the source image.

Our two-tier system helps address the underspecification of image-to-3D tasks by allowing probabilistic outputs at each level. At the same time, we encourage multiview consistency within each sample using cyclic inductive biases. Our geometry-based, hierarchical approach outperforms three SoTA baselines on reconstruction

metrics, especially in terms of shape and pose accuracy. It learns shape priors that generalize well beyond the digital assets it was trained on, paving the way for more geometry-driven novel view synthesis.

References

[1] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 510–517. IEEE, 2015. 4, 13

[2] Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, Matthieu Guillaumin, and Jitendra Malik. Abo: Dataset and benchmarks for real-world 3d object understanding. *CVPR*, 2022. 6

[3] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023. 4

[4] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. *Advances in Neural Information Processing Systems*, 36, 2024. 4, 18

[5] Maximilian Denninger, Dominik Winkelbauer, Martin Sundermeyer, Wout Boerdijk, Markus Knauer, Klaus H. Strobl, Matthias Humt, and Rudolph Triebel. Blenderproc2: A procedural pipeline for photorealistic rendering. *Journal of Open Source Software*, 8(82):4901, 2023. 18

[6] 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, 2022. 6

[7] Mohamed El Banani, Amit Raj, Kevis-Kokitsi Maninis, Abhishek Kar, Yuanzhen Li, Michael Rubinstein, Deqing Sun, Leonidas Guibas, Justin Johnson, and Varun Jampani. Probing the 3d awareness of visual foundation models. In *CVPR*, pages 21795–21806, 2024. 3

[8] Ruiqi Gao*, Aleksander Holynski*, Philipp Henzler, Arthur Brussee, Ricardo Martin-Brualla, Pratul P. Srinivasan, Jonathan T. Barron, and Ben Poole*. Cat3d: Create anything in 3d with multi-view diffusion models. *arXiv*, 2024. 1, 2, 5, 13

[9] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 1

[10] Jing He, Haodong Li, Wei Yin, Yixun Liang, Leheng Li, Kaiqiang Zhou, Hongbo Liu, Bingbing Liu, and Ying-Cong Chen. Lotus: Diffusion-based visual foundation model for high-quality dense prediction. *arXiv preprint arXiv:2409.18124*, 2024. 3

[11] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022. 3

[12] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33:6840–6851, 2020. 5

[13] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *ICLR*, 2024. 1, 2, 3, 6, 7, 13

[14] Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In *International Conference on Machine Learning*, pages 13213–13232. PMLR, 2023. 5

[15] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019.

[16] Drew A Hudson, Daniel Zoran, Mateusz Malinowski, Andrew K Lampinen, Andrew Jaegle, James L McClelland, Loic Matthey, Felix Hill, and Alexander Lerchner. SODA: Bottleneck diffusion models for representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23115–23127, 2024. 2

[17] Allan Jabri, Sjoerd van Steenkiste, Emiel Hoogeboom, Mehdi S. M. Sajjadi, and Thomas Kipf. Dorsal: Diffusion for object-centric representations of scenes et al. In *ICLR*, 2024. 3

[18] Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions, 2023. 3

[19] Rishabh Kabra, Loic Matthey, Alexander Lerchner, and Niloy Mitra. Leveraging VLM-based pipelines to annotate 3d objects. In *Forty-first International Conference on Machine Learning*, 2024. 18

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

[21] Xin Kong, Shikun Liu, Xiaoyang Lyu, Marwan Taher, Xiaojuan Qi, and Andrew J. Davison. Eschernet: A generative model for scalable view synthesis. In *CVPR*, pages 9503–9513, 2024. 2, 5

[22] Akshay Krishnan, Abhijit Kundu, Kevis-Kokitsi Maninis, James Hays, and Matthew Brown. Omninocs: A unified nocs dataset and model for 3d lifting of 2d objects. In *European Conference on Computer Vision*, pages 127–145. Springer, 2025. 4

[23] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 13

[24] 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*, pages 9298–9309, 2023. 1, 2, 6

[25] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view syn-

thesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[26] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. 1

[27] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 5

[28] Muyao Niu, Xiaodong Cun, Xintao Wang, Yong Zhang, Ying Shan, and Yinqiang Zheng. Mofa-video: Controllable image animation via generative motion field adaptions in frozen image-to-video diffusion model. *ECCV*, 2024. 3

[29] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 5

[30] Xiaqing Pan, Nicholas Charron, Yongqian Yang, Scott Peters, Thomas Whelan, Chen Kong, Omkar Parkhi, Richard Newcombe, and Carl Yuheng Ren. Aria digital twin: A new benchmark dataset for egocentric 3d machine perception, 2023. 7

[31] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *ICLR*, 2023. 1, 2

[32] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. pages 8748–8763. PMLR, 2021. 5

[33] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1 (2):3, 2022. 1

[34] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2021. 6

[35] Kyle Sargent, Zizhang Li, Tanmay Shah, Charles Herrmann, Hong-Xing Yu, Yunzhi Zhang, Eric Ryan Chan, Dmitry Lagun, Li Fei-Fei, Deqing Sun, and Jiajun Wu. Zeronvs: Zero-shot 360-degree view synthesis from a single image. In *CVPR*, pages 9420–9429, 2024. 2

[36] Roger N Shepard and Jacqueline Metzler. Mental rotation of three-dimensional objects. *Science*, 171(3972):701–703, 1971. 13

[37] Xiaoyu Shi, Zhaoyang Huang, Fu-Yun Wang, Weikang Bian, Dasong Li, Yi Zhang, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, et al. Motion-i2v: Consistent and controllable image-to-video generation with explicit motion

modeling. In *ACM SIGGRAPH 2024 Conference Papers*, pages 1–11, 2024. 3

[38] Steven G Vandenberg and Allan R Kuse. Mental rotations, a group test of three-dimensional spatial visualization. *Perceptual and motor skills*, 47(2):599–604, 1978. 13

[39] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *CVPR*, pages 2642–2651, 2019. 1, 4

[40] 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*, pages 12619–12629, 2023. 2

[41] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, pages 20697–20709, 2024. 3, 11

[42] 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 *Advances in Neural Information Processing Systems*, pages 8406–8441. Curran Associates, Inc., 2023. 1

[43] Daniel Watson, William Chan, Ricardo Martin Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel view synthesis with diffusion models. In *The Eleventh International Conference on Learning Representations*, 2023. 2

[44] Rundi Wu, Ben Mildenhall, Philipp Henzler, Keunhong Park, Ruiqi Gao, Daniel Watson, Pratul P. Srinivasan, Dor Verbin, Jonathan T. Barron, Ben Poole, and Aleksander Ho?y?ski. Reconfusion: 3d reconstruction with diffusion priors. In *CVPR*, pages 21551–21561, 2024. 2

[45] 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. *arXiv preprint arXiv:2406.09292*, 2024. 3

[46] Chao Xu, Ang Li, Linghao Chen, Yulin Liu, Ruoxi Shi, Hao Su, and Minghua Liu. Sparp: Fast 3d object reconstruction and pose estimation from sparse views. *ECCV*, 2024. 3, 4

[47] Yinghao Xu, Hao Tan, Fujun Luan, Sai Bi, Peng Wang, Jiahao Li, Zifan Shi, Kalyan Sunkavalli, Gordon Wetzstein, Zexiang Xu, et al. Dmv3d: Denoising multi-view diffusion using 3d large reconstruction model. *ICLR*, 2024. 2

[48] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 1, 2, 3

[49] Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Tianyou Liang, Guanying Chen, Shuguang Cui, and Xiaoguang Han. Mvimgnet: A large-scale dataset of multi-view images. In *CVPR*, 2023. 6

[50] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. 3

Probabilistic Inverse Cameras: Image to 3D via Multiview Geometry

Supplementary Material

A. Layout

The appendix is organized as follows: we describe CROCS in detail in Appendix B, discussing in particular the geometrical properties which led to our algorithmic choices. We then present additional qualitative comparisons and quantitative analyses in Appendix C. In particular, we assess multiview consistency across all models in Appendix C.2; we break down unPIC's hierarchical error in Appendix C.3; and we ablate the effect of classifier-free guidance in Appendix C.4. Finally, we describe all training and evaluation details for unPIC in Appendix D.

B. Geometric Subtleties

B.1. CROCS Rescaling

Recall that we pre-render multiview NOCS images using a fixed reference frame. This reference frame can be labeled using the source-view camera position $(\theta, \phi, r) = (0, 0, 1)$ that captures the object in its default creator-intended pose at a default camera distance. At training time, we would like to set the source camera to arbitrary locations to ensure a rich training distribution. To change the reference frame from $\theta = 0$ to a new source-camera location θ' (we will treat the other parameters in Appendix B.2), we simply rotate the *ground-plane axes* of all pre-rendered NOCS maps using the SO2 rotation matrix $[[\cos\theta', -\sin\theta'], [\sin\theta', \cos\theta']]$. The ground-plane axes are the Red and Green channels of the NOCS maps in our case.

Our on-the-fly NOCS-to-CROCS canonicalization comes with a challenge, arising from the fact that the object was scaled to fit the NOCS cube at $\theta = 0$. When we rotate the RGB reference cube to follow the primary camera to θ', then the object may no longer be bounded by the rotated cube. See Fig. 7 where we visualize this issue for a fixed object scaled to fit the reference cube at $\theta = 0$.

In the upper row of Fig. 7a, we get some CROCS values outside the range $[0, 1]$, because the object exceeds the boundaries of the $[0, 1]^2$ CROCS reference square. Some values of θ' such as 45, 135, 215, and 305 degrees are affected the most, because they lead to the greatest object overhang. This is not ideal—the scale of CROCS values which the model needs to predict depends on the reference frame. Since the reference frame (i.e., source camera position) is not actually supplied to the model, the model can only learn to predict the variable scale by overfitting. We aim to avoid any dependence on the reference frame. To this end, we introduce a CROCS rescaling operation to eliminate the dependence (Fig. 7a, lower row).

Taking a concrete example, Fig. 7b shows a cube-like object from different angles, along with the pre-rendered NOCS images (second row), and canonicalized CROCS maps without and with rescaling (third and fourth rows). Here are two observations from this figure: **1)** The front-facing edge of the cube, visible at $\theta' = 45$ as yellow in the second row, disappears in the third row because the CROCS values exceed the $[0, 1]$ range (and the color map is defined on the same range). If we rescale the values in the third row, we see the edge reappears in the fourth row. **2)** The maximum overhang (in the third row) occurs at $\theta' = 45$, where the range of observed y values expands to $[-0.2, 1.2]$. The length of this expanded interval is nearly $\sqrt{2} \approx 1.41$, which can also be deduced geometrically—it equals the length of the diagonal of the fixed square in Fig. 7a.
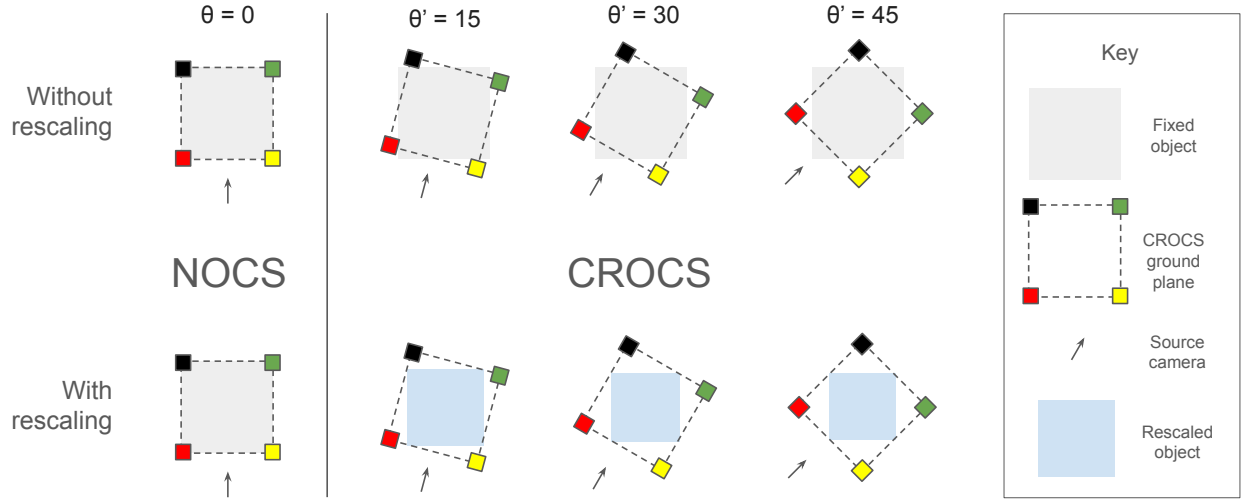
Geometrically, we expect no object overhang at all if a given object is bounded by a sphere of diameter 1 (a tighter bound than the unit cube). We believe a lot of objects fall in this category. For this reason, our model does reasonably well even if we train it to predict CROCS without rescaling. But we observed a noticeable gain in the prior's performance on CROCS with rescaling. The latter setting removes the reference-frame-dependent scale that the model needs to predict in some cases (objects like a cube).

Note that for a given object, the rescaling needs to be performed across all multiview pointmaps (K=8 target views in our case) simultaneously, since any given view only shows a particular facet of the object. (A single pointmap may entirely miss the variation along its depth component.) Our rescaling is reminiscent of the pairwise pointmap renormalization performed by DUSt3R (see Eq. 3 in [41]). All the results in this work use CROCS with rescaling.
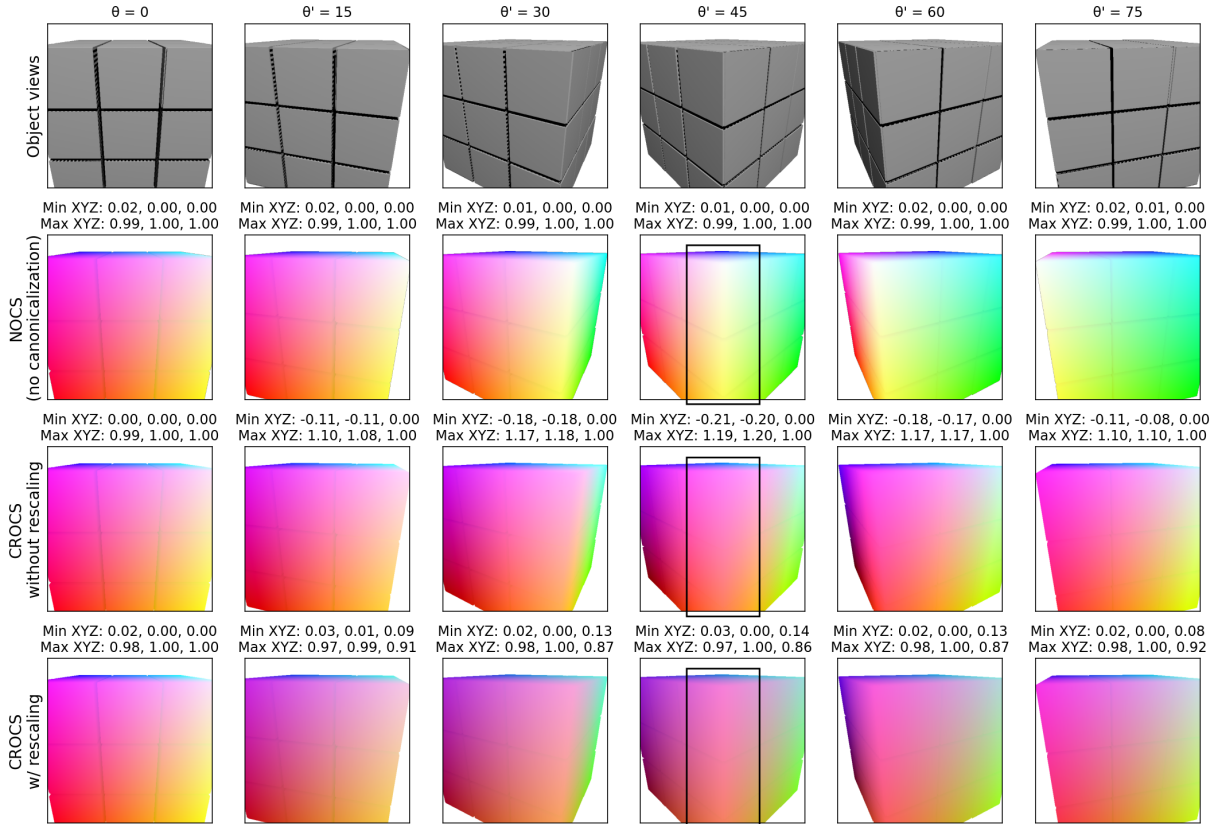
B.2. No Canonicalization for Camera Elevation or Distance

In Appendix B.1 we discussed how to canonicalize NOCS when switching the source camera from θ to θ'. Here, we discuss the remaining camera extrinsic parameters. Since all cameras are oriented to face the center of the object, we do not need to consider their rotations. In fact, we do not need to canonicalize for changes in the camera distance r. It only serves as a zoom parameter to augment the training data.

The only remaining parameter is the elevation angle ϕ, which determines the source-camera height. Recall that the target cameras are also placed at the same camera height. In fact, ϕ determines the difficulty of the multiview prediction problem. For instance, at $\phi = 90$ (a straight-down view of the object), the source and target camera locations converge

(a) **The general case** (simplified to 2D). **Top row:** An object that fits the NOCS reference square exactly at $\theta = 0$ is no longer bounded by the CROCS reference square at $\theta' \in \{15, 30, 45\}$. Due to object overhang, some CROCS values lie outside $[0, 1]$. The extent of the overhang depends on θ'. **Bottom row:** We rescale CROCS based on the observed range of multiview values for any θ'. This ensures the object is once again bounded tightly by $[0, 1]^2$.

(b) **A concrete example in 3D.** We show the minimum and maximum NOCS/CROCS values observed above each pointmap image. At $\theta' = 45$, we see that the front-facing edge of the cube appears squashed without rescaling, because its X and Y values lie outside the range $[0, 1]$.

Figure 7. **CROCS rescaling.** We examine the effect of rotating the RGB reference cube used to paint the object's surface, following the source camera as it moves around a fixed object. θ (the azimuthal angle) denotes the default camera position, while θ' denotes a new position. We consider the largest possible objects—a square in 2D or cube in 3D.
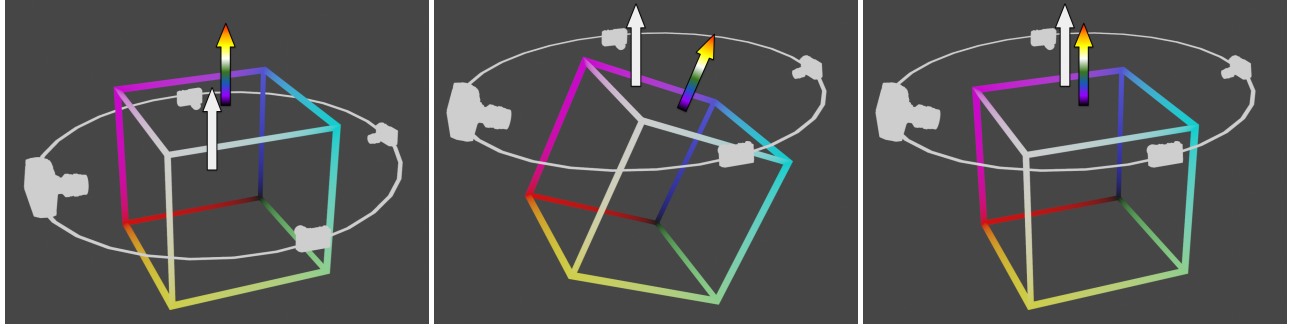
Figure 8. **CROCS when varying the camera elevation angle** ϕ. As in Fig. 3, the large camera denotes the source view; the smaller cameras denote 3 of 7 novel views. The white arrow denotes the normal direction w.r.t. the camera plane (drawn as a white circle connecting the cameras). The colored arrow denotes the normal of the CROCS color cube. **Left:** at the default angle $\phi = 0$, each camera is pointed at a particular face of the RGB color cube, and has a consistent color distribution across examples. **Middle:** when the source camera is raised to ϕ', we could hypothetically rotate the color cube by the same degree to ensure the source camera still points at the same face of the cube. The issue is—tilting the color cube toward the source camera tilts it away from the opposite camera, and also rotates the colors seen by the remaining target cameras (by ϕ' and $-\phi'$ in their respective image planes). The color shifts experienced by the four cameras are clearly inconsistent. **Right:** If we choose not to tilt the cube through ϕ', all cameras undergo a small and consistent shift in the distribution of colors toward the up-axis color (blue). If the model can infer ϕ', it can also predict the shift. We follow the *Right* approach.

to the same point—the apex of the hemisphere. In this case, the prediction task is reduced to rotating the object in the image plane.

Say we raise/lower the source camera from ϕ to ϕ' (Fig. 8). This would reveal more/less of the object's upper/lower surface, and hence more/less of the NOCS color representing the up-axis (Blue in our case). We could potentially correct for this by tilting the reference cube so the source camera is pointed at the same cube face as before at ϕ. This would help maintain the same color bias for the source camera. However, the target cameras would no longer point at the same cube faces respectively—rather, their distribution of colors would shift (toward the up-axis/away from it in the case of the opposite target camera) or rotate (in the image plane for other target cameras). Consequently, the distribution shifts in the CROCS colors would be inconsistent across target cameras.

Based on this geometrical observation, we choose not to canonicalize for changes in the camera elevation ϕ'. This has the effect of leaving it to the model to infer the camera elevation from the source image (implicitly in contrast to One-2-3-45 [23]). It differs from our treatment of θ', which leaves the model oblivious to the azimuthal rotation of the cameras.

Our decision not to canonicalize for ϕ is forced by the design choice of our target camera poses, which follow the camera height of the source camera. An alternative choice would be to tilt the camera plane and the CROCS cube together, leaving the cameras at different heights in Fig. 8-Middle[1]. Our choice of target poses is based on how hu-

mans perceive and reason about objects. We tend to see side or top-down views of *level* objects. On mental rotation tasks (e.g., when asked what an object looks like from "behind"), we tend to imagine a rotation of the object either in the image plane, or in depth about the world's vertical axis [36, 38]. We find it harder to imagine a potentially bottom-facing view. In addition to aligning with human perception, modeling object rotations at typical views/poses (e.g., top-down or side angles) could also be more useful for downstream applications.

C. Additional Results

C.1. Comparisons

We present further qualitative results comparing unPIC with CAT3D and One-2-3-45 in Figs. 9 to 12.

Besides its multiview inconsistency, One-2-3-45 also has the flaw that it rescales any given object to a canonical size in the image plane. This is visible from the model's source-pose output (top-left cell in each 2x4 output grid); it is resized compared to the source-pose outputs of other methods. Our approach can cope with arbitrarily sized objects in images despite the fact that it relies on a scale-free 3D shape representation (CROCS).

Note that both CAT3D and our method predict gray values at background pixels. But unPIC produces RGBA outputs rather than RGB—the alpha mask hides the gray background pixels after compositing. While we do compute masks using posthoc background removal for CAT3D, we

---

[1]This alternative is equivalent to using a stationary multiview camera rig, but orienting the object randomly before taking pictures, as in CAT3D

[8], likely LRM [13], and datasets like YCB [1]. Changing the object's orientation decouples the object's up-axis from the world's up-axis. It can leave the object in unnatural, physically unstable poses.
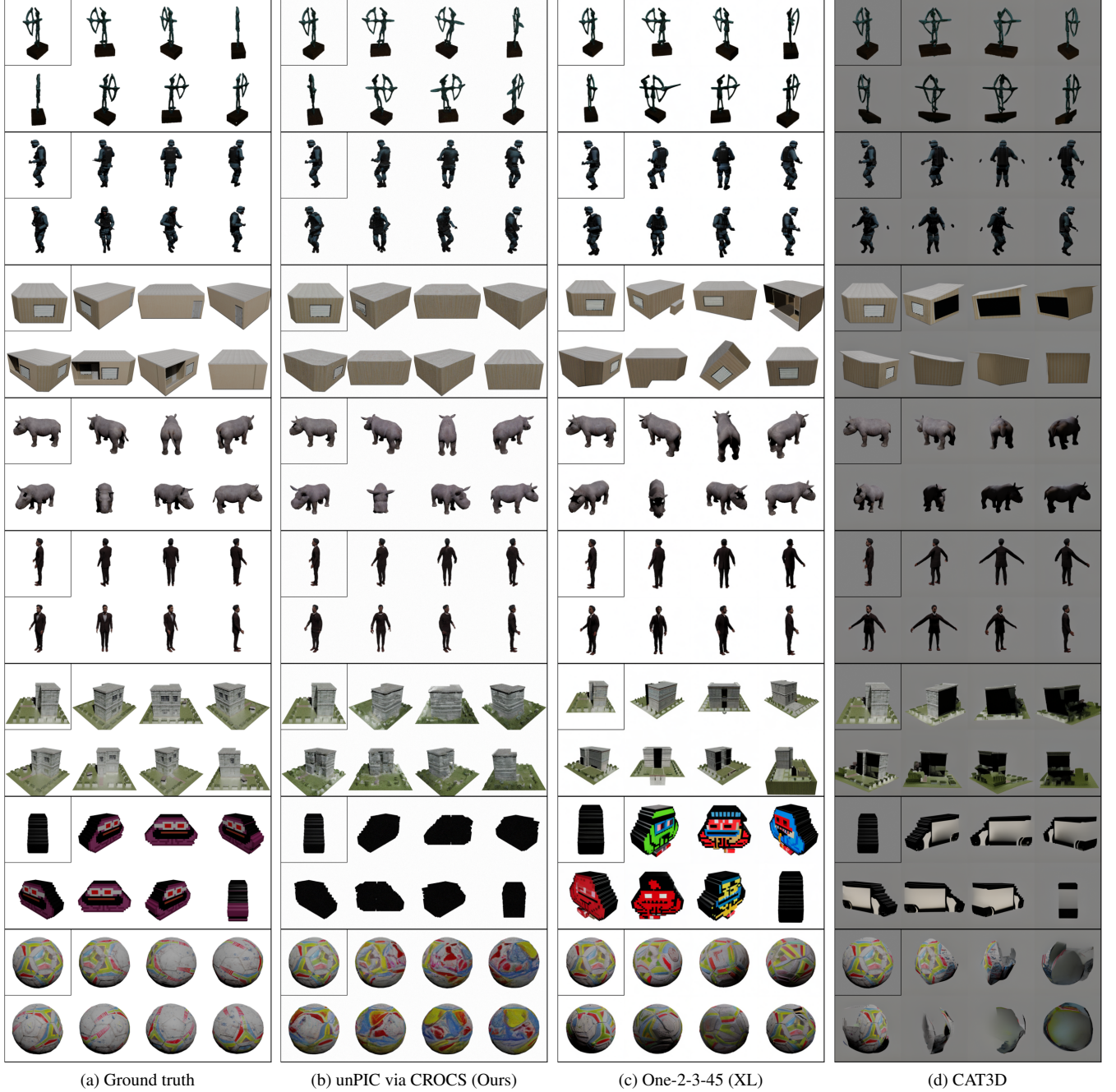
Figure 9. **Additional qualitative comparison on Objaverse-XL holdouts.**

(a) Ground truth   (b) unPIC via CROCS (Ours)   (c) One-2-3-45 (XL)   (d) CAT3D

only use these masks to compute the IoU metric. The remaining methods (One-2-3-45 and OpenLRM) output RGB images with white background pixels. We present untampered results for all methods in Figs. 9 to 12.

C.2. Multiview Consistency

Qualitatively, we have seen that models like One-2-3-45 and CAT3D can be inconsistent (in terms of appearance as well as 3D geometry) in the images the generate for a given object. The issue is partly due to their lack of geometrical priors, and potentially exacerbated in One-2-3-45 by single-view synthesis. To assess multiview consistency systematically, we introduce the following metric: we compute CLIP embeddings (using the open-source ViT L/14 336px model) for all K generated images in a given example, then compute the mean of the (K x K) pairwise distance matrix of
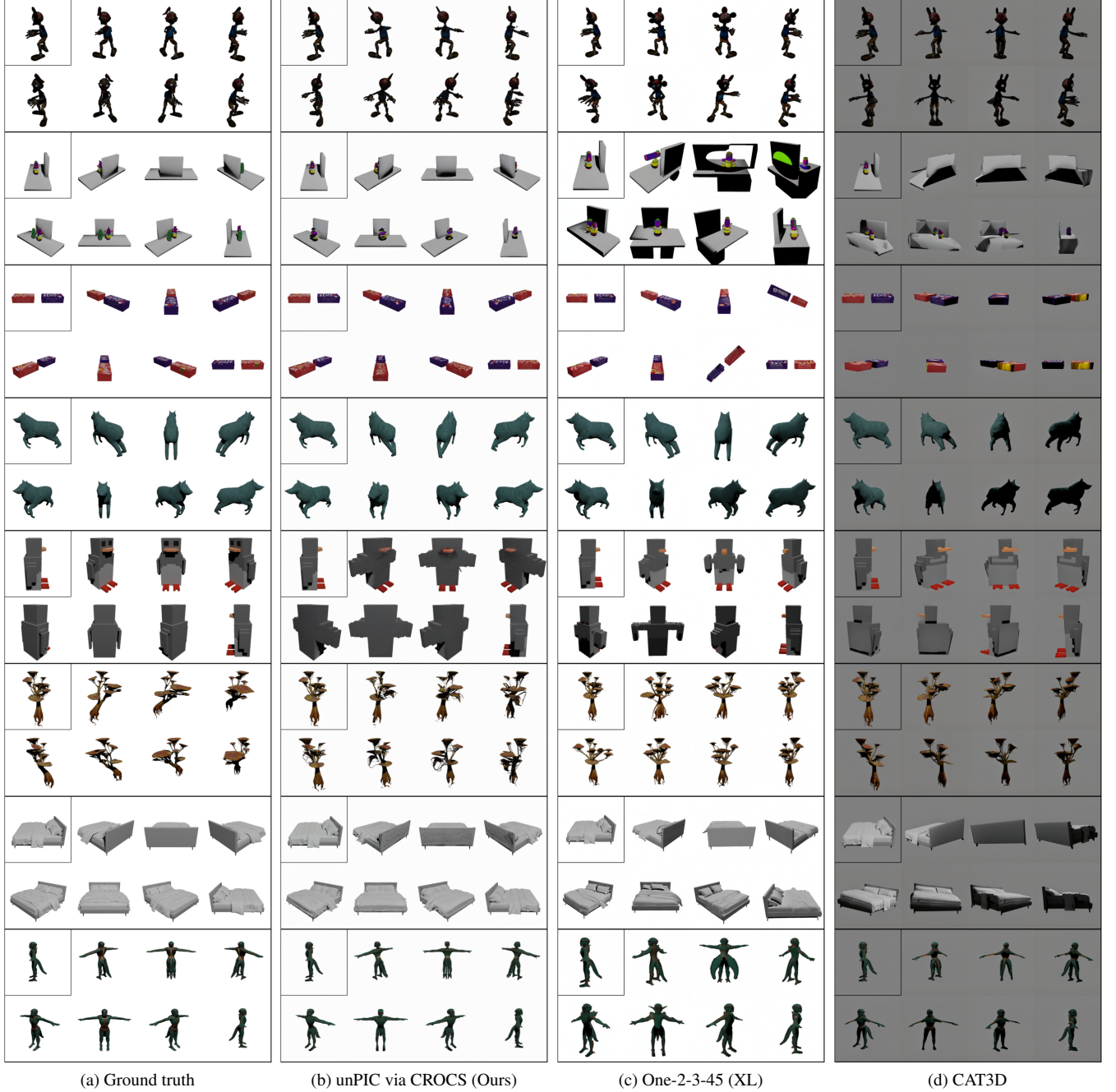
Figure 10. **Additional qualitative comparison on Objaverse-XL holdouts.**

the CLIP embeddings. This metric avoids any comparison with ground-truth images, and merely probes the internal consistency of a set of images. We use the same model outputs that we evaluated in Sec 4.3.

We also report the metric for ground-truth images to validate that the metric is sensible. The ground-truth numbers are not lower bounds, because a model could optimize multiview consistency by generating the same image K times.

Rather, the ground-truth numbers help validate that the metric is reasonable—it could hypothetically penalize models with better 3D geometry for producing less 2D-consistent views. But we see in Tab. 5 that the metric is indeed reasonable, with the ground-truth images generally producing the best score. We also find that unPIC outperforms all other methods consistently, including the geometry-driven OpenLRM.
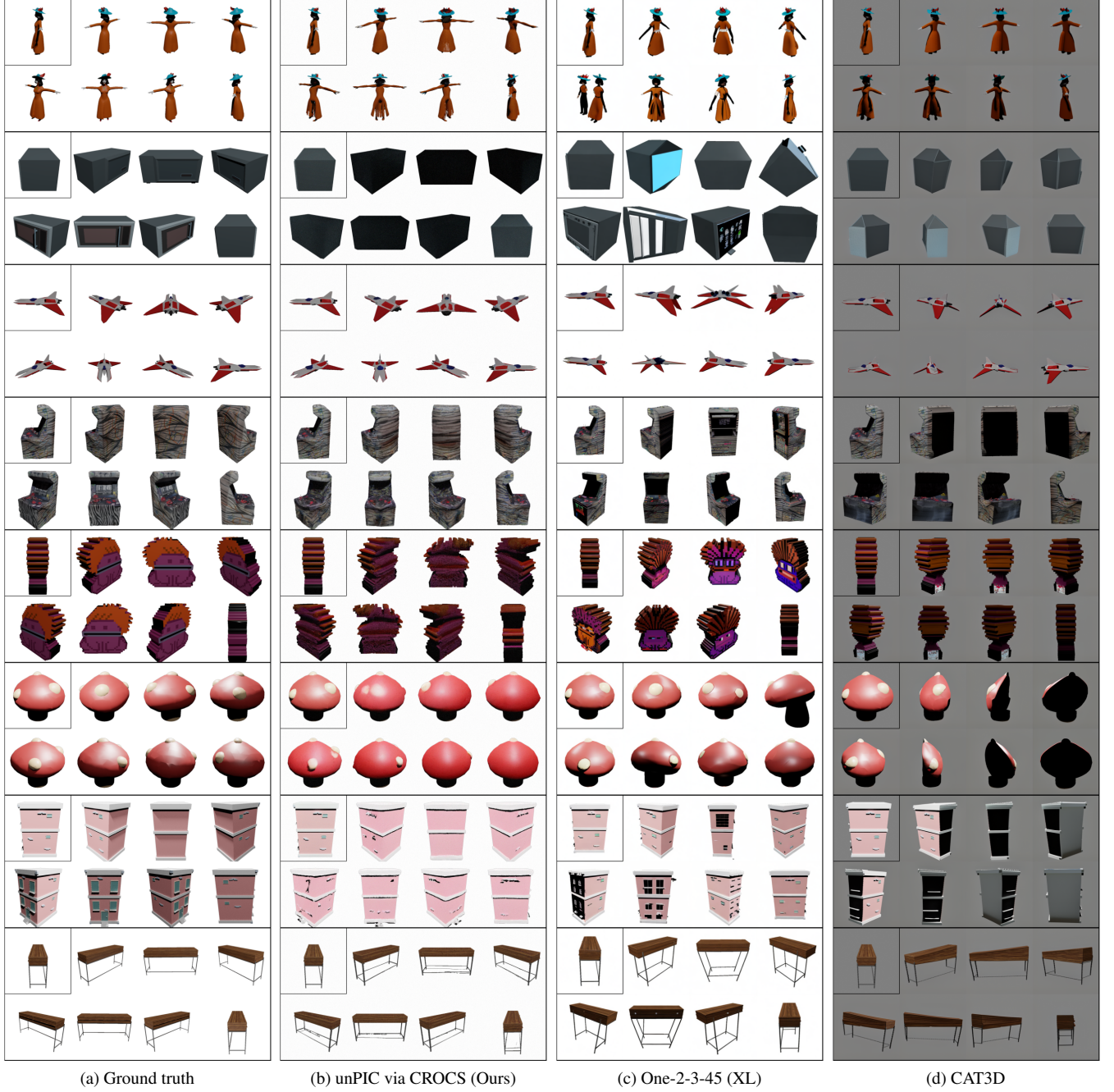
15

Figure 11. **Additional qualitative comparison on Objaverse-XL holdouts.**

C.3. Decomposing unPIC's hierarchical error

In Sec 4.3 we reported metrics from the full hierarchical setup, i.e., on the final output of the prior and decoder stacked together. Since we train the prior and decoder modules separately, we can further analyze the following components of the total error:

1. The error of the prior alone. This describes the geometrical inaccuracy (when the intermediate representation is CROCS) in predicting the 3D shape and pose of a given object.

2. The error of the decoder when it is fed ground-truth CROCS rather than the output of the prior. This describes the difficulty of rendering the object (e.g., predicting the object's texture) at novel views when extrapolating from a single source image.
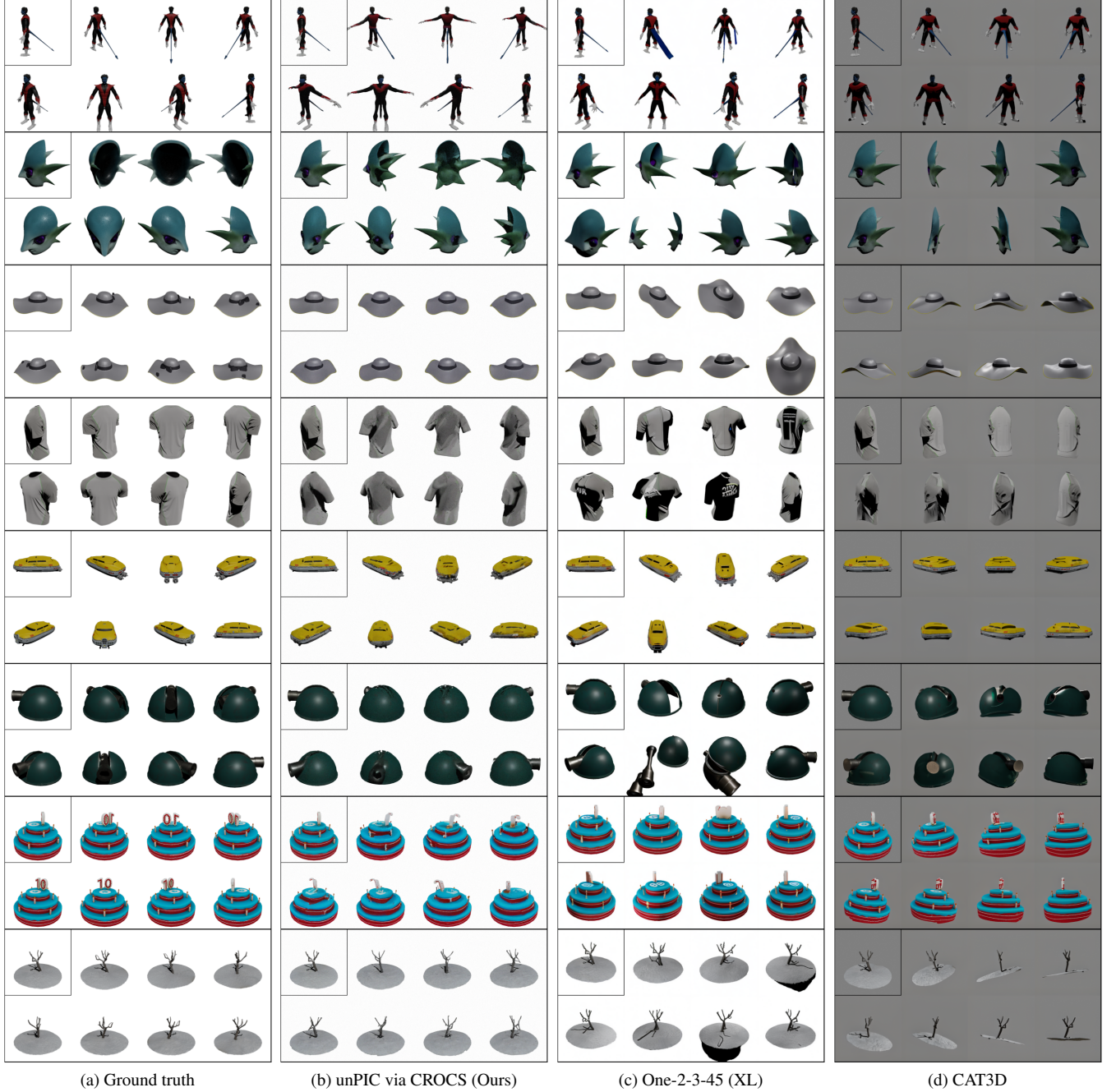
16

Figure 12. **Additional qualitative comparison on Objaverse-XL holdouts.**

We report these components along with total hierarchical error in Tab. 6. We find that the prior's error (column 1) is nearly twice that of the oracle decoder (column 2). This highlights the importance of getting the geometry correct. Improving the geometrical prior would help reduce the total error (column 3) more than improving the final rendering (i.e., the decoder module). With geometry predictions as good as the ground-truth CROCS, our full

multiview-synthesis error could be 4-5x lower (depending on the dataset).

C.4. Effect of Classifier-Free Guidance

We run a hyperparameter sweep on the weight of the classifier-free guidance applied to the prior or decoder at sampling time. A weight greater than 1.0 has the effect of pushing the sampler in the conditional direction over the un-

17

Table 5. **Assessing multiview consistency using pairwise CLIP distances** ($\downarrow$). For each method, we take all $K = 8$ generated views per example, embed them using CLIP, and compute their mean pairwise distance. We also report the same metric for the ground-truth images on each dataset. All numbers are 1e-4.

	CAT3D	One-2-3-45	OpenLRM	unPIC via CROCS (ours)	GT images
ObjXL	2.44	2.55	2.38	<u>2.16</u>	**2.10**
GSO	3.08	2.74	2.93	<u>2.63</u>	**2.61**
ABO	2.37	2.22	2.55	**1.74**	<u>1.77</u>
DTC	2.70	1.98	2.40	<u>1.81</u>	**1.49**

Table 6. **A breakdown of unPIC's hierarchical error (256x256).** We report Mean Square Errors (1e-3) for the prior module, the decoder module when using ground-truth CROCS, and the full error running stacked inference. The MSE is calculated on all 7 novel views.

	Image $\rightarrow$ CROCS	GT CROCS $\rightarrow$ Pixels	Image $\rightarrow$ CROCS $\rightarrow$ Pixels
ObjXL	4.58	2.84	10.52
ABO	3.46	1.65	7.60
GSO	2.69	3.09	7.63

Table 7. **The effect of classifier-free guidance** while sampling from the prior and decoder modules (on ObjXL holdouts). We apply the CFG weights only on the source image in both modules respectively.

Prior CFG	Decoder CFG	PSNR $\uparrow$	IoU $\uparrow$	FID $\downarrow$	LPIPS $\downarrow$
1.0	1.0	23.891	0.770	110.497	0.490
1.0	2.0	23.891	0.770	110.470	0.490
1.0	4.0	23.891	0.770	110.470	0.490
1.0	8.0	23.891	0.770	110.497	0.490
2.0	1.0	**24.027**	**0.789**	104.920	**0.480**
2.0	2.0	**24.027**	**0.789**	**104.895**	**0.480**
2.0	4.0	**24.027**	**0.789**	104.897	**0.480**
2.0	8.0	**24.027**	**0.789**	**104.895**	**0.480**
4.0	1.0	23.401	0.783	104.583	0.482
4.0	2.0	23.401	0.783	104.541	0.482
4.0	4.0	23.403	0.783	104.539	0.482
4.0	8.0	23.406	0.783	104.605	0.482
8.0	1.0	22.175	0.769	111.273	0.496
8.0	2.0	22.179	0.769	110.948	0.496
8.0	4.0	22.175	0.769	111.273	0.496
8.0	8.0	22.179	0.769	110.913	0.496

conditional direction. Based on the empirical performance of the CFG weights in Tab. 7, we use fixed weights of 2.0 for our remaining experiments and samples.

D. More Training and Evaluation Details

Dataset. We use the following sets of assets from Objaverse: 1) the LVIS subset expanded to 83k examples from the same categories. We use object type labels predicted with high confidence from [19] to expand the LVIS subset. 2) The KIUI subset [2] comprising 101k additional assets. 3) From the Objaverse-XL alignment subset [4], we used a combination of GLB, OBJ, and FBX assets after filtering out certain (a) terrains and HDRI environment maps, (b) layouts and rooms, and (c) textureless FBX assets that rendered as pink. In total, we used 620k assets (Objaverse and Objaverse-XL) for training. In addition, we held out 50k assets randomly sampled from Objaverse-XL after the filtering step.

To render NOCS images, we adapted a script from BlenderProc[3] [5] which creates a special NOCS material for the surface of a given object. We also use their Blender settings (the CYCLES engine with 1 diffuse bounce, 0 glossy bounces, and 0 ambient occlusion bounces) for rendering NOCS. We export them in the EXR format to ensure linear-light values, preserving the continuity of NOCS values on the object's surface.

Model. We assume an RGB source image, but predict RGBA CROCS and RGBA output images. The architecture for the prior and decoder has just under 150M parameters. It takes about 6 minutes to run the prior and decoder each for 2000 denoising steps, with classifier-free guidance, on an A100. As efficiency wasn't a core goal for this work, we expect more efficient sampling strategies and better architectures could certainly help reduce the runtime. We use an exponential moving average of the model parameters for evaluation, although this is not essential.

---

[2]https://github.com/ashawkey/objaverse_filter
[3]https://github.com/DLR-RM/BlenderProc/