LRM: LARGE RECONSTRUCTION MODEL FOR SINGLE IMAGE TO 3D

Anonymous authors
Paper under double-blind review

ABSTRACT

We propose the first Large Reconstruction Model (LRM) that predicts the 3D model of an object from a single input image within just 5 seconds. In contrast to many previous methods that are trained on small-scale datasets such as ShapeNet in a category-specific fashion, LRM adopts a highly scalable transformer-based architecture with 500 million learnable parameters to directly predict a neural radiance field (NeRF) from the input image. We train our model in an end-to-end manner on massive multi-view data containing around 1 million objects, including both synthetic renderings from Objaverse and real captures from MVImgNet. This combination of a high-capacity model and large-scale training data empowers our model to be highly generalizable and produce high-quality 3D reconstructions from various testing inputs including real-world in-the-wild captures and images from generative models. Video demos and interactable 3D meshes can be found on this anonymous website: https://scalei3d.github.io/LRM.

1 INTRODUCTION

Imagine if we could instantly create a 3D shape from a single image of an arbitrary object. Broad applications in industrial design, animation, gaming, and AR/VR have strongly motivated relevant research in seeking a generic and efficient approach towards this long-standing goal. Due to the underlying ambiguity of 3D geometry in a single view, early learning-based methods usually perform well on specific categories, utilizing the category data prior to infer the overall shape (Yu et al., 2021). Recently, advances in image generation, such as DALL-E (Ramesh et al., 2021) and Stable Diffusion (Rombach et al., 2022), have inspired research that leverages the remarkable generalization capability of 2D diffusion models to enable multi-view supervision (Liu et al., 2023b; Tang et al., 2023). However, many of these methods require delicate parameter tuning and regularization, and their results are limited by the pre-trained 2D generative models. Meanwhile, there are many approaches that rely on per-shape optimization (*e.g.* optimize a NeRF (Mildenhall et al., 2021; Chan et al., 2022; Chen et al., 2022a; Müller et al., 2022; Sun et al., 2022)) to construct a consistent geometry; this process is often slow and impractical.

On the other hand, the great success in natural language processing (Devlin et al., 2018; Brown et al., 2020; Chowdhery et al., 2022) and image processing (Caron et al., 2021; Radford et al., 2021; Alayrac et al., 2022; Ramesh et al., 2022) can be largely credited to three critical factors: (1) using highly scalable and effective neural networks, such as the Transformers (Vaswani et al., 2017), for modeling the data distribution, (2) enormous datasets for learning generic priors, as well as (3) self-supervised-like training objectives that encourage the model to discover the underlying data structure while maintaining high scalability. For instance, the GPT (generative pre-trained transformer) series (Radford et al., 2019; Brown et al., 2020; OpenAI, 2023) build large language models with huge transformer networks, large-scale data, and the simple next-word prediction task. In light of this, we pose the same question for 3D: given sufficient 3D data and a large-scale training framework, ***is it possible to learn a generic 3D prior for reconstructing an object from a single image?***

In this paper, we propose a **L**arge **R**econstruction **M**odel (LRM) for single-image to 3D. Our method adopts a large transformer-based encoder-decoder architecture for learning 3D representations of objects from a single image in a data-driven manner. Our method takes an image as input and regresses a NeRF in the form of a triplane representation (Chan et al., 2022). Specifically, LRM utilizes the

pre-trained visual transformer DINO (Caron et al., 2021) as the image encoder to generate the image features, and learns an image-to-triplane transformer decoder to project the 2D image features onto the 3D triplane via cross-attention and model the relations among the spatially-structured triplane tokens via self-attention. The output tokens from the decoder are reshaped and upsampled to the final triplane feature maps. Afterwards, we can render the images at an arbitrary view by decoding the triplane feature of each point with an additional shared multi-layer perception (MLP) to get its color and density and performing volume rendering.

The overall design of LRM maintains high scalability and efficiency. In addition to the use of a fully transformer-based pipeline, a triplane NeRF is a concise and scalable 3D representation since it is computationally friendly compared to other representations such as volumes and point clouds. It also has a better locality with respect to the image input compared to tokenizing the NeRF's model weights as in Shap-E (Jun & Nichol, 2023). Moreover, our LRM is trained by simply minimizing the difference between the rendered images and ground truth images at novel views, without excessive 3D-aware regularization or delicate hyper-parameter tuning, allowing the model to be very efficient in training and adaptable to a wide range of multi-view image datasets.

To the best of our knowledge, LRM is the first *large-scale 3D reconstruction model*; it contains more than 500 million learnable parameters, and it is trained on approximately one million 3D shapes and video data across diverse categories (Deitke et al., 2023; Yu et al., 2023); this is substantially larger than recent methods that apply relatively shallower networks and smaller datasets (Chang et al., 2015; Reizenstein et al., 2021; Downs et al., 2022). Through experiments, we show that LRM can reconstruct high-fidelity 3D shapes from a wide range of images captured in the real world, as well as images created by generative models. LRM is also a highly practical solution for downstream applications since it can produce a 3D shape in just five seconds[1] without post-optimization.

2 Related work

Single Image to 3D Reconstruction Extensive efforts have been devoted to address this problem, including early learning-based methods that explore point clouds (Fan et al., 2017; Wu et al., 2020), voxels (Choy et al., 2016; Tulsiani et al., 2017; Chen & Zhang, 2019), and meshes (Wang et al., 2018; Gkioxari et al., 2019), as well as various approaches that learn implicit representations such as SDFs (Park et al., 2019; Mittal et al., 2022), occupancy networks (Mescheder et al., 2019), and NeRF (Jang & Agapito, 2021; Müller et al., 2022). Leveraging 3D templates (Roth et al., 2016; Goel et al., 2020; Kanazawa et al., 2018; Kulkarni et al., 2020), semantics (Li et al., 2020), and poses (Bogo et al., 2016; Novotny et al., 2019) as shape priors have also been widely studied in category-specific reconstruction. Category-agnostic methods show great generalization potential (Yan et al., 2016; Niemeyer et al., 2020), but they often unable to produce fine-grained details even when exploiting spatially-aligned local image features (Xu et al., 2019; Yu et al., 2021).

Very recently, there is an emerging trend of using pre-trained image/language models (Radford et al., 2021; Li et al., 2022; 2023; Saharia et al., 2022; Rombach et al., 2022), to introduce semantics and multi-view guidance for image-to-3D reconstruction (Liu et al., 2023b; Tang et al., 2023; Deng et al., 2023; Shen et al., 2023b; Melas-Kyriazi et al., 2023; Metzer et al., 2023; Xu et al., 2023; Qian et al., 2023). For instance, Zero-1-to-3 fine-tunes the Stable Diffusion model to generate novel views by conditioning on the input image and camera poses (Liu et al., 2023b); its view consistency and reconstruction efficiency have been further improved by Liu et al. (2023a). Make-It-3D (Tang et al., 2023) uses BLIP to generate text descriptions for the input image (which is applied to guide the text-to-image diffusion) and trains the model with score distillation sampling loss (Poole et al., 2022) and CLIP image loss to create geometrically and semantically plausible shapes.

In contrast to all these methods, our LRM is a purely data-driven approach that learns to reconstruct arbitrary objects in the wild. It is trained with minimal and extensible 3D supervision (*i.e.*, rendered or captured 2D images of 3D objects) and does not rely on any guidance from pre-trained vision-language contrastive or generative models.

[1]Five seconds per shape on a single NVIDIA A100 GPU, including around 1.14 seconds image-to-triplane feed-forward time, 1.14 seconds to query resolution of $384 \times 384 \times 384$ points from the triplane-NeRF, and 1.91 seconds mesh extraction time using Marching Cubes (Lorensen & Cline, 1998).

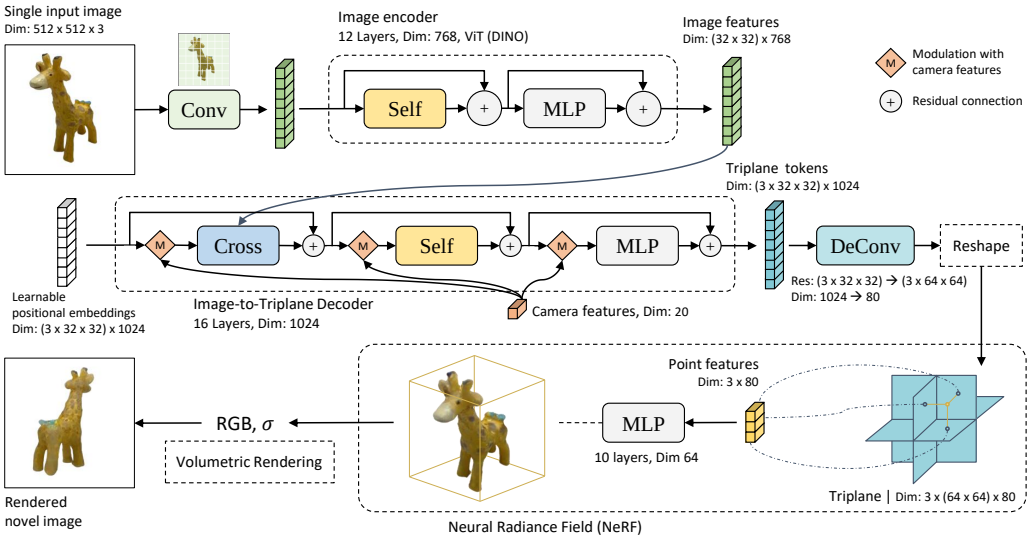

Figure 1: The overall architecture of LRM, a fully-differentiable transformer-based encoder-decoder framework for single-image to NeRF reconstruction. LRM applies a pre-trained vision model (DINO) to encode the input image (Sec. 3.1), where the image features are projected to a 3D triplane representation by a large transformer decoder via cross-attention (Sec. 3.2), followed by a multi-layer perceptron to predict the point color and density for volumetric rendering (Sec. 3.3). The entire network is trained end-to-end on around a million of 3D data (Sec. 4.1) with simple image reconstruction losses (Sec. 3.4).

Learning 3D Representations from Images 3D reconstruction from a single image is an ill-posed problem that has been frequently addressed by models with generative properties. Many previous works apply an encoder-decoder framework to model the image-to-3D data distribution (Choy et al., 2016; Yan et al., 2016; Dai et al., 2017; Xu et al., 2019; Wu et al., 2020; Müller et al., 2022), where a compact latent code is trained to carry the texture, geometry, and pose details of the target. However, learning such an expressive representation usually requires a capable network and abundant 3D data which is very expensive to acquire. Hence most of these methods only focus on a few categories and produce very coarse results. GINA-3D (Shen et al., 2023a) implements a model that applies a visual transformer encoder and cross-attention (instead of the transformer decoder in LRM) to translate images to triplane representations. But the model and training are much smaller in scale, and their work has a different focus on category-specific generation. Recent data-driven approach MCC (Wu et al., 2023) trains a generalizable transformer-based decoder with CO3D-v2 data (Reizenstein et al., 2021) to predict occupancy and color from the input image and its unprojected point cloud. Although MCC can handle real and generated images and scenes, the results are usually over-smooth and lose details.

Multimodal 3D Motivated by the great advances in 2D multimodal learning (Tan & Bansal, 2019; Chen et al., 2020; 2022b; Yu et al., 2022; Singh et al., 2022; Wang et al., 2022; Alayrac et al., 2022; Girdhar et al., 2023), LRM considers 3D as a new modality and directly grounds 2D feature maps onto 3D triplane via cross-attention. There are early attempts in this direction that minimize the difference between encoded image and 3D representations (Girdhar et al., 2016; Mandikal et al., 2018), as well as recent research, ULIP (Xue et al., 2023) and CLIP^2 (Zeng et al., 2023), which bridges 3D, language, and images via contrastive learning. LERF (Kerr et al., 2023) learns a language field inside NeRF by rendering CLIP embeddings along training rays. In contrast, our method focuses on generic single image-to-3D reconstruction. We would like to mention the concurrent work Cap3D (Luo et al., 2023) that produces descriptions for 3D shapes by applying BLIP (Li et al., 2023) to generate captions of different views, uses GPT-4 (OpenAI, 2023) to summarize them, and then employs these language-3D pairs for training text-to-3D generative models (Nichol et al., 2022; Poole et al., 2022; Jun & Nichol, 2023). There are also recent works in connecting 3D and large language models (Hong et al., 2023; Yang et al., 2023).

3 METHOD

In this section, we detail the proposed LRM architecture (Fig. 1). LRM contains an image encoder that encodes the input image to patch-wise feature tokens (Sec. 3.1), followed by an image-to-triplane decoder that projects image features onto triplane tokens via cross-attention (Sec. 3.2). The output triplane tokens are upsampled and reshaped into the final triplane representation, which is used to query 3D point features. Lastly, the 3D point features are passed to a multi-layer perception to predict RGB and density for volumetric rendering (Sec. 3.3). The training objectives and data are described in Sec. 3.4 and Sec. 4.1.

3.1 IMAGE ENCODER

Given an RGB image as input, LRM first applies a pre-trained visual transformer (ViT) (Dosovitskiy et al., 2020) to encode the image to patch-wise feature tokens $\{\boldsymbol{h}_i\}_{i=1}^{n} \in \mathbb{R}^{d_E}$, where i denotes the i-th image patch, n is the total number of patches, and d_E is the latent dimension of the encoder. Specifically, we use DINO (Caron et al., 2021), a model trained with self-distillation that learns interpretable attention over the structure and texture of the salient content in images. Compared to other semantic-oriented representations such as the visual features from ImageNet-pretrained ResNet (He et al., 2016) or CLIP (Radford et al., 2021), the detailed structural and texture information in DINO is more important in our case since LRM can use it to reconstruct the geometry and color in 3D space. As a result, instead of only using the ViT pre-defined class token `[CLS]` that aggregates patch-wise features, we also utilize the entire feature sequence $\{\boldsymbol{h}_i\}_{i=1}^{n}$ to better preserve this information[2].

3.2 IMAGE-TO-TRIPLANE DECODER

We implement a transformer decoder to project image and camera features onto learnable spatial-positional embeddings and translate them to triplane representations. This decoder can be considered as a prior network that is trained with large-scale data to provide necessary geometric and appearance information to compensate for the ambiguities of single-image reconstruction.

Camera Features We construct the camera feature $\boldsymbol{c} \in \mathbb{R}^{20}$ of the input image by flattening out the 4-by-4 camera extrinsic matrix $\boldsymbol{E}$ (that represents the camera-to-world transformation) and concatenate it with the camera focal length foc and principal point pp as $\boldsymbol{c} = [\boldsymbol{E}_{1\times 16}, foc_x, foc_y, pp_x, pp_y]$. Moreover, we normalize the camera extrinsic $\boldsymbol{E}$ by similarity transformations so that all the input cameras are aligned on the same axis (with the lookup direction aligned with the z-axis). Note that, LRM does not depend on a canonical pose of the object, and the ground truth $\boldsymbol{c}$ is only applied in training. Conditioning on normalized camera parameters greatly reduces the optimization space of triplane features and facilitates model convergence (see details in Sec. 4.2). To embed the camera feature, we further implement a multi-layer perceptron (MLP) to map the camera feature to a high-dimensional camera embedding $\tilde{\boldsymbol{c}}$. The intrinsics (focal and principal point) are normalized by the image's height and width before sending to the MLP layer.

Triplane Representation We follow previous works (Chan et al., 2022; Gao et al., 2022) to apply triplane as a compact and expressive feature representation of the reconstruction subject. A triplane $\boldsymbol{T}$ contains three axis-aligned feature planes $\boldsymbol{T}_{XY}$, $\boldsymbol{T}_{YZ}$ and $\boldsymbol{T}_{XZ}$. In our implementation, each plane is of dimension $(64\times64)\times d_T$ where 64×64 is the spatial resolution, and d_T is the number of feature channels. For any 3D point in the NeRF object bounding box $[-1, 1]^3$, we can project it onto each of the planes and query the corresponding point features $(\boldsymbol{T}_{xy}, \boldsymbol{T}_{yz}, \boldsymbol{T}_{xz})$ via bilinear interpolation, which is then decoded by an MLP^{nerf} into the NeRF color and density (Sec. 3.3).

To obtain the triplane representation $\boldsymbol{T}$, we define learnable spatial-positional embeddings $\boldsymbol{f}^{init}$ of dimension $(3\times32\times32)\times d_D$ which guide the image-to-3D projection and are used to query the image features via cross-attention, where d_D is the hidden dimension of the transformer decoder. The number of tokens in $\boldsymbol{f}^{init}$ is smaller than the number of final triplane tokens ($3\times64\times64$); we will upsample the output of the transformer $\boldsymbol{f}^{out}$ to the final $\boldsymbol{T}$. In the forward pass, conditioning on the camera features $\tilde{\boldsymbol{c}}$ and image features $\{\boldsymbol{h}_i\}_{i=1}^{n}$, each layer of our image-to-triplane transformer

[2]For simplicity, we use $\{\boldsymbol{h}_i\}_{i=1}^{n}$ in the following to denote the concatenated sequence of the encoded `[CLS]` token and patch-wise features

decoder gradually updates the initial positional embedding $\boldsymbol{f}^{init}$ to the final triplane features via modulation and cross-attention, respectively. The reason for applying two different conditional operations is that the camera controls the orientation and distortion of the whole shape, whereas the image features carry the fine-grained geometric and color information that need to be embedded onto the triplane. Details of the two operations are explained below.

Modulation with Camera Features Our camera modulation is inspired by DiT (Peebles & Xie, 2022) which implements an adaptive layer norm (adaLN) to modulate image latents with denoising timesteps and class labels. Suppose $\{\boldsymbol{f}_j\}$ is a sequence of vectors in transformer, we define our modulation function $\mathrm{ModLN_c}(\boldsymbol{f}_j)$ with camera feature $\boldsymbol{c}$ as

$$\gamma, \beta = \mathrm{MLP}^{\mathrm{mod}}(\tilde{\boldsymbol{c}}) \tag{1}$$

$$\mathrm{ModLN_c}(\boldsymbol{f}_j) = \mathrm{LN}(\boldsymbol{f}_j) \cdot (1 + \gamma) + \beta \tag{2}$$

where γ and β are the scale and shift (Huang & Belongie, 2017) output by $\mathrm{MLP}^{\mathrm{mod}}$ and LN is the Layer Normalization (Ba et al., 2016). Such modulation is applied to each attention sub-layer which will be specified next.

Transformer Layers Each transformer layer contains a cross-attention sub-layer, a self-attention sub-layer, and a multi-layer perceptron sub-layer (MLP), where the input tokens to each sub-layer are modulated by the camera features. Suppose feature sequence $\boldsymbol{f}^{in}$ is the input of an transformer layer, we can consider $\boldsymbol{f}^{in}$ as the triplane hidden features since they are corresponding to the final triplane features $\boldsymbol{T}$. As shown in the decoder part of Fig. 1, the cross-attention module firstly attends from the triplane hidden features $\boldsymbol{f}^{in}$ to the image features $\{\boldsymbol{h}_i\}_{i=1}^{n}$, which can help linking image information to the triplane. Note that we here do not explicitly define any spatial alignment between the 2D images and 3D triplane hidden features, but consider 3D as an independent modality and ask the model to learn the 2D-to-3D correspondence by itself. The updated triplane hidden features will be passed to a self-attention sub-layer that further models the intra-modal relationships across the spatially-structured triplane entries. Then, a multi-layer perceptron sub-layer (MLP^{tfm}) follows as in the original Transformer (Vaswani et al., 2017) design. Lastly, the output triplane features $\boldsymbol{f}^{out}$ will become the input to the next transformer layer.

Such a design is similar to the Perceiver network (Jaegle et al., 2021) while our model maintains a high-dimensional representation across the attention layers instead of projecting the input to a latent bottleneck. Overall, we can express this process for each j-th triplane entry in each layer as

$$\boldsymbol{f}_j^{cross} = \mathrm{CrossAttn}(\mathrm{ModLN}_c(\boldsymbol{f}_j^{in}); \{\boldsymbol{h}_i\}_{i=1}^{n}) + \boldsymbol{f}_j^{in} \tag{3}$$

$$\boldsymbol{f}_j^{self} = \mathrm{SelfAttn}(\mathrm{ModLN}_c(\boldsymbol{f}_j^{cross}); \{\mathrm{ModLN}_c(\boldsymbol{f}_{j'}^{cross})\}_{j'}) + \boldsymbol{f}_j^{cross} \tag{4}$$

$$\boldsymbol{f}_j^{out} = \mathrm{MLP}^{tfm}(\mathrm{ModLN}_c(\boldsymbol{f}_j^{self})) + \boldsymbol{f}_j^{self} \tag{5}$$

The ModLN operators in sub-layers (*i.e.*, $\mathrm{CrossAttn}$, $\mathrm{SelfAttn}$, MLP^{tfm}) use different set of learnable parameters in the layer normalization and the modulation MLP^{mod}. We do not add additional superscript to differentiate them for clarity.

The transformer layers are processed sequentially. After all the transformer layers, we obtain the output triplane features $\boldsymbol{f}^{\mathrm{out}}$ from the last layer as the output of the decoder. This final output is upsampled by a learnable de-convolution layer and reshaped to the final triplane representation $\boldsymbol{T}$.

3.3 TRIPLANE-NERF

We employ the triplane-NeRF formulation (Chan et al., 2022) and implement an MLP^{nerf} to predict RGB and density σ from the point features queried from the triplane representation $\boldsymbol{T}$. The MLP^{nerf} contains multiple linear layers with ReLU (Nair & Hinton, 2010) activation. The output dimension of the MLP^{nerf} is 4 where the first three dimensions are RGB colors and the last dimension corresponds to the density of the field. We refer to the Appendix for the details of NeRF volumetric rendering.

3.4 TRAINING OBJECTIVES

LRM produces the 3D shape from a single input image and leverages additional side views to guide the reconstruction during training. For each shape in the training data, we consider $(V-1)$ randomly chosen side views for supervision; we apply simple image reconstruction objectives between the V rendered views $\hat{\boldsymbol{x}}$ and the ground-truth views $\boldsymbol{x}^{GT}$ (include the input view and side views). More precisely, for every input image $\boldsymbol{x}$, we minimize:

$$\mathcal{L}_{\text{recon}}(\boldsymbol{x}) = \frac{1}{V}\sum_{v=1}^{V}\left(\mathcal{L}_{\text{MSE}}(\hat{\boldsymbol{x}}_v, \boldsymbol{x}_v^{GT}) + \lambda\mathcal{L}_{\text{LPIPS}}(\hat{\boldsymbol{x}}_v, \boldsymbol{x}_v^{GT})\right) \tag{6}$$

where $\mathcal{L}_{\text{MSE}}$ is the normalized pixel-wise L2 loss, $\mathcal{L}_{\text{LPIPS}}$ is the perceptual image patch similarity (Zhang et al., 2018) and λ is a customized weight coefficient.

4 EXPERIMENTS

4.1 DATA

LRM relies on abundant 3D data from Objaverse (Deitke et al., 2023) and MVImgNet (Yu et al., 2023), consisting of synthetic 3D assets and videos of objects in the real world respectively, to learn a generalizable cross-shape 3D prior. For each 3D asset in Objaverse, we normalize the shape to the box $[-1, 1]^3$ in world space and render 32 random views with the same camera pointing toward the shape at arbitrary poses. The rendered images are of resolution 1024×1024, and the camera poses are sampled from a ball of radius $[1.5, 3.0]$ and with height in range $[-0.75, 1.60]$[3]. For each video, we utilize the extracted frames from the dataset. Since the target shape in those frames can be at random positions, we crop and resize all of them using the predicted object mask[4] so that the object is at the center of the resulting frames; we adjust the camera parameters accordingly. Note that, our method does not model background, hence we render images from Objaverse with a pure white background, and use an off-the-shelf package[4] to remove the background of video frames. In total, we pre-processed 730,648 3D assets and 220,219 videos for training.

To evaluate the performance of LRM on arbitrary images, we collected novel images from Objaverse (Deitke et al., 2023), MvImgNet (Yu et al., 2023), ImageNet (Deng et al., 2009), Google Scanned Objects (Downs et al., 2022), Amazon Berkeley Objects (Collins et al., 2022), captured new images in the real world, and generated images with Adobe Firefly[5] for reconstruction. We visualize their results in Sec. 4.3.1 and Appendix. To numerically study the design choices of our approach, we randomly acquired 50 unseen 3D shapes from the Objaverse and 50 unseen videos from the MvImgNet dataset, respectively. For each shape, we pre-process 15 reference views and pass five of them to our model one by one to reconstruct the same object, and evaluate the rendered images using all 15 reference views (see analyses in Appendix).

4.2 IMPLEMENTATION DETAILS

Camera Normalization We normalize the camera poses corresponding to the input images to facilitate the image-to-triplane modeling. Specifically, for the images rendered from synthetic 3D assets in Objaverse, we normalize the input camera poses to position $[0, -2, 0]$, with the camera vertical axis aligned with the upward z-axis in the world frame. For the video data, since the camera can be at an arbitrary distance from the target and the object is not at the image center, we only normalize the camera pose to $[0, -dis, 0]$ where dis is the original distance between world origin and camera origin.

Network Architecture We apply the ViT-B/16 model of pre-trained DINO as the image encoder, which takes 512×512 RGB images as input and produces 1025 feature tokens (1024 patch-wise features plus one `[CLS]` features) (Caron et al., 2021). The output dimension of ViT is 768 (d_E). The image-to-triplane decoder and the MLP^{nerf} are of 16 and 10 layers with hidden dimensions

[3]Most of Objaverse assets have consistent z-axis up.

[4]Rembg package, a tool to remove image background: https://pypi.org/project/rembg

[5]Adobe Firefly, a text-to-image generation tool: https://firefly.adobe.com

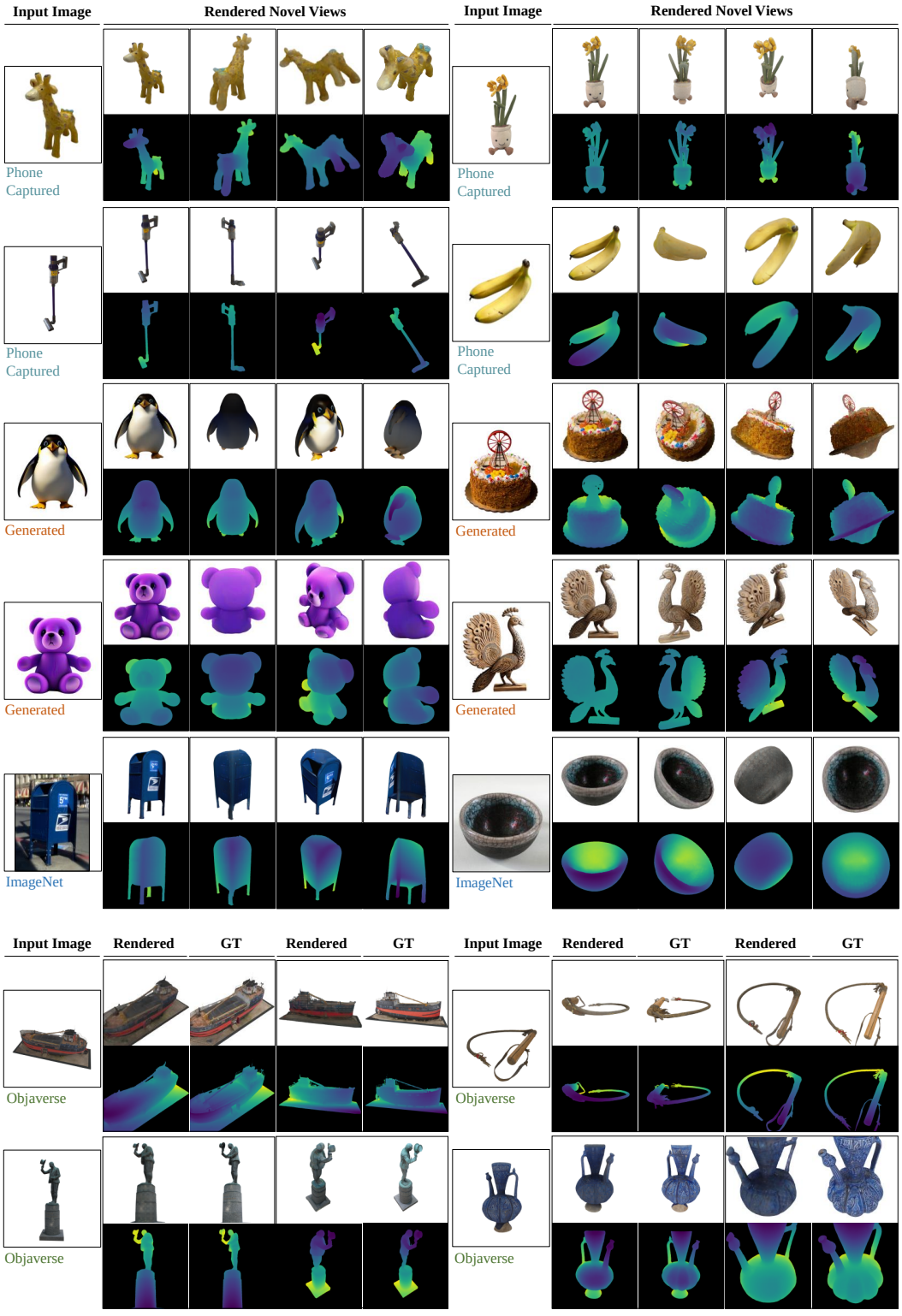

Figure 2: Rendered novel views (RGB and depth) of shapes reconstructed by our LRM from single images. None of the images are observed by the model during training. Generated images are created using Adobe Firefly. The last two rows compare our results to the rendered ground truth images of Objaverse objects (GT). Please zoom in for clearer visualization.

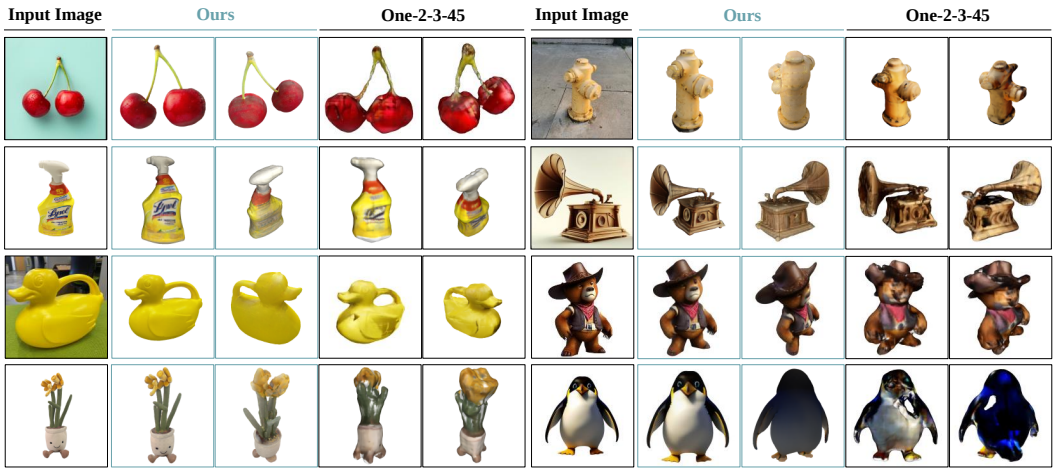

Figure 3: Comparison to One-2-3-45 (Liu et al., 2023a). To avoid cherry-picking, input images in the first three rows are selected from the examples provided in One-2-3-45's paper or demo page. None of the images are observed by our model during training. Please zoom in for clearer visualization.

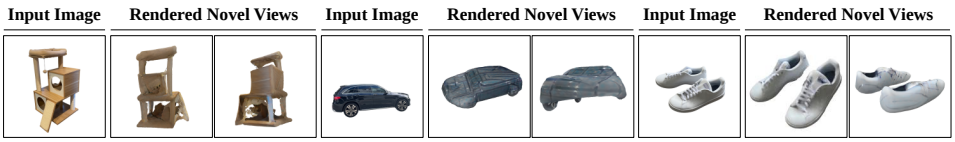

Figure 4: Failure cases of our method. All three examples show blurry textures for occluded regions, and distortion due to the largely inaccurate assumption of the camera parameters.

1024 (d_D) and 64, respectively. The triplane dimension is 80 (d_T). For neural rendering, we sample 128 points for each ray and render 128×128 resolution images for supervision. We also use the deferred back-propagation technique in the ARF work (Zhang et al., 2022) to save GPU memory.

Training We train LRM on 128 NVIDIA (40G) A100 GPUs with batch size 1024 (1024 different shapes per iteration) for 30 epochs, taking about 3 days to complete. Each epoch contains one copy of the rendered image data from Objaverse and three copies of the video frame data from MvImgNet to balance the amount of synthetic and real data. For each sample, we use 3 randomly chosen side views (*i.e.*, the total views $V = 4$) to supervise the shape reconstruction, and we set the coefficient λ=2.0 for $\mathcal{L}_{\text{LPIPS}}$. We apply the AdamW optimizer (Loshchilov & Hutter, 2017) and set the learning rate to 4×10^{-4} with a cosine schedule (Loshchilov & Hutter, 2016). We numerically analyze the influence of data, training, and model hyper-parameters in Appendix.

Inference During inference, LRM takes an arbitrary image as input (squared and background removed) and assumes the unknown camera parameters to be the normalized cameras that we applied to train the Objaverse data. We query a resolution of $384\times384\times384$ points from the reconstructed triplane-NeRF and extract the mesh using Marching Cubes (Lorensen & Cline, 1998). This entire process only takes less than 5 seconds to complete on a single NVIDIA A100 GPU.

4.3 Results

We visualize the novel views of shapes reconstructed from real, generated, and rendered images from various datasets (Fig. 2), compare our method with a concurrent work (Liu et al., 2023a) (Fig. 3), and summarize some failure cases of our method (Sec. 4.3.2). Numerical analyses of data, model architecture, and supervision can be found in the Appendix.

4.3.1 VISUALIZATION

Figure 2 visualizes some examples of the shapes reconstructed from single images. Overall, the results show very high fidelity for diverse inputs, including real, generated, and rendered images of various subjects with distinct textures. Not only is complex geometry correctly modeled (*e.g.* chair, flagon, and wipe), but also the high-frequency details, such as the texture of the wood peafowl, are preserved; both reflecting the great generalization ability of our model. From the asymmetric examples, giraffe, penguin, and bear, we can see that LRM can infer semantically reasonable occluded portion of the shapes, which implies effective cross-shape priors have been learned.

In Figure 3, we compare LRM with One-2-3-45, a concurrent work to ours that achieves state-of-the-art single image to 3D reconstruction by generating multi-view images with 2D diffusion models (Liu et al., 2023a). To avoid cherry-picking, we directly test our method on the example images provided in their paper or demo page[6]. We can see that our method produces much sharper details and consistent surfaces. In the last row of the figure, we test One-2-3-45 with two examples used in Figure 2, showing much worse reconstruction results.

4.3.2 LIMITATIONS

Despite the high-quality single-image-to-3D results we have shown, our method still has a few limitations. First, our LRM tends to produce blurry textures for occluded regions, as shown in Figure 4. Our conjecture is that this is due to the fact that the single-image-to-3D problem is inherently probablistic, i.e., multiple plausible solutions exist for the unseen region, but our model is deterministic and is likely producing averaged modes of the unseens. Second, during inference time, we assign a set of fixed camera intrinsics and extrinsics (same as our Objaverse training data) to the test images. These camera parameters may not align well with the ground-truth, especially when the images are cropped and resized, causing large changes to Field-of-View (FoV) and principal points. Figure 4 shows that incorrect assumption of the camera parameters can lead to distorted shape reconstruction. Third, we only address images of objects without background; handling the background (Zhang et al., 2020; Barron et al., 2022), as well as complex scenes, is beyond the scope of this work. Finally, we assume Lambertian objects and omit the view-dependent modelling (Mildenhall et al., 2021) in our predicted NeRF. Therefore we cannot faithfully reconstruct the view-dependent appearance of some real-world materials, *e.g.*, shiny metals, glossy ceramics, etc.

5 CONCLUSION

In this paper, we propose LRM, the first large transformer-based framework to learn an expressive 3D prior from a million 3D data to reconstruct objects from single images. LRM is very efficient in training and inference; it is a fully-differentiable network that can be trained end-to-end with simple image reconstruction losses and only takes five seconds to render a high-fidelity 3D shape, thus enabling a wide range of real-world applications. In the era of large-scale learning, we hope our idea can inspire future research to explore data-driven 3D large reconstruction models that generalize well to arbitrary in-the-wild images.

Future Directions The future directions of our research are mainly twofold; (1) Scaling up the model and training data: with the simplest transformer-based design and minimal regularization, LRM can be easily scaled to a larger and more capable network, including but not limited to applying a larger image encoder, adding more attention layers to the image-to-triplane decoder, and increasing the resolution of triplane representations. On the other hand, LRM only requires multi-view images for supervision, hence a wide range of 3D, video, and image datasets can be exploited in training. We expect both approaches to be promising in improving the model's generalization ability and the quality of reconstruction. (2) Extension to multimodal 3D generative models: LRM model builds a pathway for generating novel 3D shapes from language by leveraging a text-to-image generation model to first create 2D images. But more interestingly, we suggest the learned expressive triplane representations could be applied to directly bridge language descriptions and 3D to enable efficient text-to-3D generation and editing (*e.g.*, via latent diffusion (Rombach et al., 2022)). We will explore this idea in our future research.

[6]One-2-3-45 demo page: https://huggingface.co/spaces/One-2-3-45/One-2-3-45.

ETHICS STATEMENT

LRM proposed in this paper is a deterministic model in which, given the same image as input, the model will infer the identical 3D shape. Unlike generative models that can be used to easily synthesize various undesirable contents (*e.g.*, from language inputs), LRM requests the specific 2D content to exist in the first place. LRM is trained on Objaverse (Deitke et al., 2023) and MvImgNet (Yu et al., 2023) data, which mostly contain ethical content. However, given an unethical or misleading image, LRM could produce unethical 3D objects or 3D disinformation that may be more convincing than the 2D input images (although the reconstructed objects are less realistic than real-world objects).

Image-to-3D reconstruction models like LRM hold the potential to automate tasks currently performed by 3D designers. However, it's worth noting that these tools also have the capacity to foster growth and enhance accessibility within the creative industry.

REPRODUCIBILITY STATEMENT

Our LRM is built by integrating the publicly available codebases of threestudio[7] (Guo et al., 2023), x-transformers[8], and DINO[9] (Caron et al., 2021), and the model is trained using publicly available data from Objaverse (Deitke et al., 2023) and MvImgNet (Yu et al., 2023). We include very comprehensive data pre-processing, network architecture, and training details in this paper, which greatly facilitate reproducing our LRM.

REFERENCES

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in Neural Information Processing Systems*, 35:23716–23736, 2022.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

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

Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*, pp. 561–578. Springer, 2016.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16123–16133, 2022.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[7] threestudio's GitHub page: https://github.com/threestudio-project/threestudio.

[8] x-transformers's GitHub page: https://github.com/lucidrains/x-transformers.

[9] DINO's GitHub page: https://github.com/facebookresearch/dino.

Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022a.

Jun Chen, Han Guo, Kai Yi, Boyang Li, and Mohamed Elhoseiny. Visualgpt: Data-efficient adaptation of pretrained language models for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18030–18040, 2022b.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pp. 104–120. Springer, 2020.

Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5939–5948, 2019.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, pp. 628–644. Springer, 2016.

Jasmine Collins, Shubham Goel, Kenan Deng, Achleshwar Luthra, Leon Xu, Erhan Gundogdu, Xi Zhang, Tomas F Yago Vicente, Thomas Dideriksen, Himanshu Arora, et al. Abo: Dataset and benchmarks for real-world 3d object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21126–21136, 2022.

Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5868–5877, 2017.

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*, pp. 13142–13153, 2023.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20637–20647, 2023.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

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)*, pp. 2553–2560. IEEE, 2022.

Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 605–613, 2017.

Jun Gao, Tianchang Shen, Zian Wang, Wenzheng Chen, Kangxue Yin, Daiqing Li, Or Litany, Zan Gojcic, and Sanja Fidler. Get3d: A generative model of high quality 3d textured shapes learned from images. *Advances In Neural Information Processing Systems*, 35:31841–31854, 2022.

Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI 14*, pp. 484–499. Springer, 2016.

Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15180–15190, 2023.

Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh r-cnn. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9785–9795, 2019.

Shubham Goel, Angjoo Kanazawa, and Jitendra Malik. Shape and viewpoint without keypoints. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, pp. 88–104. Springer, 2020.

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.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus), 2023.

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 3d-llm: Injecting the 3d world into large language models. *arXiv*, 2023.

Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 1501–1510, 2017.

Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pp. 4651–4664. PMLR, 2021.

Wonbong Jang and Lourdes Agapito. Codenerf: Disentangled neural radiance fields for object categories. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12949–12958, 2021.

Heewoo Jun and Alex Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023.

Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 371–386, 2018.

Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. *arXiv preprint arXiv:2303.09553*, 2023.

Nilesh Kulkarni, Abhinav Gupta, David F Fouhey, and Shubham Tulsiani. Articulation-aware canonical surface mapping. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 452–461, 2020.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pp. 12888–12900. PMLR, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

Xueting Li, Sifei Liu, Kihwan Kim, Shalini De Mello, Varun Jampani, Ming-Hsuan Yang, and Jan Kautz. Self-supervised single-view 3d reconstruction via semantic consistency. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pp. 677–693. Springer, 2020.

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. *arXiv preprint arXiv:2306.16928*, 2023a.

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. *arXiv preprint arXiv:2303.11328*, 2023b.

William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*, pp. 347–353. 1998.

Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. Scalable 3d captioning with pre-trained models. *arXiv preprint arXiv:2306.07279*, 2023.

Priyanka Mandikal, KL Navaneet, Mayank Agarwal, and R Venkatesh Babu. 3d-lmnet: Latent embedding matching for accurate and diverse 3d point cloud reconstruction from a single image. *arXiv preprint arXiv:1807.07796*, 2018.

Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8446–8455, 2023.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4460–4470, 2019.

Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12663–12673, 2023.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 306–315, 2022.

Norman Müller, Andrea Simonelli, Lorenzo Porzi, Samuel Rota Bulo, Matthias Nießner, and Peter Kontschieder. Autorf: Learning 3d object radiance fields from single view observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3971–3980, 2022.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. doi: 10.1145/3528223.3530127. URL https://doi.org/10.1145/3528223.3530127.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814, 2010.

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.

Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3504–3515, 2020.

David Novotny, Nikhila Ravi, Benjamin Graham, Natalia Neverova, and Andrea Vedaldi. C3dpo: Canonical 3d pose networks for non-rigid structure from motion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7688–7697, 2019.

OpenAI. Gpt-4 technical report, 2023.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 165–174, 2019.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022.

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

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.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pp. 8821–8831. PMLR, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10901–10911, 2021.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Joseph Roth, Yiying Tong, and Xiaoming Liu. Adaptive 3d face reconstruction from unconstrained photo collections. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4197–4206, 2016.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

Bokui Shen, Xinchen Yan, Charles R Qi, Mahyar Najibi, Boyang Deng, Leonidas Guibas, Yin Zhou, and Dragomir Anguelov. Gina-3d: Learning to generate implicit neural assets in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4913–4926, 2023a.

Qiuhong Shen, Xingyi Yang, and Xinchao Wang. Anything-3d: Towards single-view anything reconstruction in the wild. *arXiv preprint arXiv:2304.10261*, 2023b.

Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. Flava: A foundational language and vision alignment model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15638–15650, 2022.

Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022.

Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.

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. *arXiv preprint arXiv:2303.14184*, 2023.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Shubham Tulsiani, Tinghui Zhou, Alexei A Efros, and Jitendra Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2626–2634, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *Proceedings of the European conference on computer vision (ECCV)*, pp. 52–67, 2018.

Yi Wang, Kunchang Li, Yizhuo Li, Yinan He, Bingkun Huang, Zhiyu Zhao, Hongjie Zhang, Jilan Xu, Yi Liu, Zun Wang, et al. Internvideo: General video foundation models via generative and discriminative learning. *arXiv preprint arXiv:2212.03191*, 2022.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Chao-Yuan Wu, Justin Johnson, Jitendra Malik, Christoph Feichtenhofer, and Georgia Gkioxari. Multiview compressive coding for 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9065–9075, 2023.

Rundi Wu, Yixin Zhuang, Kai Xu, Hao Zhang, and Baoquan Chen. Pq-net: A generative part seq2seq network for 3d shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 829–838, 2020.

Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Yi Wang, and Zhangyang Wang. Neurallift-360: Lifting an in-the-wild 2d photo to a 3d object with 360deg views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4479–4489, 2023.

Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. *Advances in neural information processing systems*, 32, 2019.

Le Xue, Mingfei Gao, Chen Xing, Roberto Martín-Martín, Jiajun Wu, Caiming Xiong, Ran Xu, Juan Carlos Niebles, and Silvio Savarese. Ulip: Learning a unified representation of language, images, and point clouds for 3d understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1179–1189, 2023.

Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. *Advances in neural information processing systems*, 29, 2016.

Jianing Yang, Xuweiyi Chen, Shengyi Qian, Nikhil Madaan, Madhavan Iyengar, David F. Fouhey, and Joyce Chai. Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent, 2023.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4578–4587, 2021.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

Xianggang Yu, Mutian Xu, Yidan Zhang, Haolin Liu, Chongjie Ye, Yushuang Wu, Zizheng Yan, Chenming Zhu, Zhangyang Xiong, Tianyou Liang, et al. Mvimgnet: A large-scale dataset of multi-view images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9150–9161, 2023.

Yihan Zeng, Chenhan Jiang, Jiageng Mao, Jianhua Han, Chaoqiang Ye, Qingqiu Huang, Dit-Yan Yeung, Zhen Yang, Xiaodan Liang, and Hang Xu. Clip2: Contrastive language-image-point pretraining from real-world point cloud data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15244–15253, 2023.

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

Kai Zhang, Nick Kolkin, Sai Bi, Fujun Luan, Zexiang Xu, Eli Shechtman, and Noah Snavely. Arf: Artistic radiance fields. In *European Conference on Computer Vision*, pp. 717–733. Springer, 2022.

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.

APPENDICES

A BACKGROUND FOR MODEL COMPONENTS

A.1 NERF

We adopt NeRF (Mildenhall et al., 2021), specifically the compact triplane NeRF variant (Chan et al., 2022), as our 3D representation to predict in LRM. NeRF, when coupled with differentiable volume rendering, can be optimized with just image reconstruction losses.

At the core of NeRF (Mildenhall et al., 2021) and its variants (Chan et al., 2022; Chen et al., 2022a; Müller et al., 2022; Sun et al., 2022) is a spatially-varying color (modeling appearance) and density (modeling geometry) field function. [10] Given a 3D point $\mathbf{p}$, the color and density field $(\mathbf{u}, \sigma)$ can be written as:

$$(\mathbf{u}, \sigma) = \mathrm{MLP}^{nerf}(f_\theta(\mathbf{p})), \tag{7}$$

where the spatial encoding f_θ is used to facilitate the MLP^{nerf} to learn high-frequency signals. Different NeRF variants (Chan et al., 2022; Chen et al., 2022a; Müller et al., 2022; Sun et al., 2022) typically differ from each other in terms of the choice of the spatial encoding and the size of the MLP. In this work, we use the triplane spatial encoding function proposed by EG3D (Chan et al., 2022), because of its low tokenization complexity ($O(N^2)$ as opposed to a voxel grid's $O(N^3)$ complexity, where N is spatial resolution).

Images are rendered from NeRF using volume rendering that's trivially differentiable. In detail, for each pixel to render, we cast a ray $\mathbf{r}$ through a NeRF, and use finite point samples $\mathbf{p}_i$ along the ray to compute the volume rendering integral to get the rendered color $\mathbf{u}(\mathbf{r})$:

$$\mathbf{u}(\mathbf{r}) = \sum_i T_i(1 - \exp(-\sigma_i \delta_i))\mathbf{u}_i, \tag{8}$$

$$T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j), \tag{9}$$

where $(\mathbf{u}_i, \sigma_i) = \mathrm{MLP}_\phi(f_\theta(\mathbf{p}_i))$ and δ_i is the distance between point $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$.

A.2 TRANSFORMER LAYERS

In this subsection, we provide the details of the layers used in the transformer decoder (Vaswani et al., 2017) as a background. For the Vision Transformer encoder, please refer to the original DINO paper (Caron et al., 2021) for implementation details.

Attention operator Attention operator is an expressive neural operator which converts an input feature x with condition to a sequence of other features $\{y_i\}$. It first computes the attention score α_i by using the dot product between the input x and each condition feature y_i. An additional $\mathrm{softmax}$ is added after the dot products to normalize the weights to a summation of 1. This attention score measures the relationship between input and conditions. Then the output is the weighted summation of the conditions $\{y_i\}$ with respect to the attention score α_i.

$$\alpha_i = \mathrm{softmax}_i\{x^\top y_i\} \tag{10}$$

$$\mathrm{Attn}(x; \{y_i\}_i) = \sum_i \alpha_i y_i \tag{11}$$

For some specific cases (*e.g.*, in the transformer attention layer below), the attention operator wants to differentiate the vectors used in calculating the attention score and the vectors for final outputs. Thus it will introduce another set of 'value' vectors $\{z_i\}_i$, and treat the $\{y_i\}_i$ as corresponding 'key' vectors. Taking this into consideration, the formula would become

$$\alpha_i = \mathrm{softmax}_i\{x^\top y_i\} \tag{12}$$

$$\mathrm{Attn}(x; \{y_i\}_i, \{z_i\}_i) = \sum_i \alpha_i z_i \tag{13}$$

[10] To simplify the discussion, we ignore the view-dependent modeling in NeRF (Mildenhall et al., 2021).

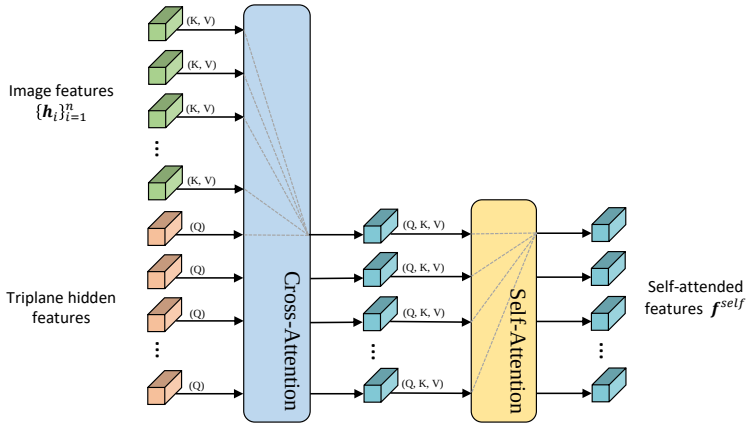

Figure 5: Visual illustration of the cross-attention and self-attention in LRM's image-to-triplane decoder.

Multi-head Attention The attention operator described above only attends to the condition features once to get the attention vector. However, the actual attention might contain multiple modes. Thus, the multi-head attention (Vaswani et al., 2017) is proposed. The multi-head attention is implemented by first splitting the input features into smaller queries.

$$[x^1, \dots, x^{nh}] = x \tag{14}$$

where nh is the number of heads. Meanwhile, y_i and z_i are split into $\{y_i^k\}_k$ and $\{z_i^k\}_k$ in a similar way. After that, the output of each head is computed independently and the final output is a concatenation of heads' outputs.

$$out^k = \text{Attn}(x^k; \{y_i^k\}_i, \{z_i^k\}_i) \tag{15}$$

$$\text{MultiHeadAttn}(x; \{y_i\}_i, \{z_i\}_i) = [out^1, \dots, out^{nh}] \tag{16}$$

Attention Layers in Transformer The detailed attention layers in transformer utilize the above multi-head attention with more linear layers. Here are the formulas for the self-attention layer (see the right yellow 'Self-Attention' block in Fig. 5). The layer first projects the input feature sequence $f = \{f_j\}_j$ to query q, key k, and value v vectors with linear layers. Then the multi-head attention is applied. There is one more linear layer over the output. We also follow the recent papers (Chowdhery et al., 2022; Touvron et al., 2023) to remove the bias terms in the attention layers.

$$q_j = W_{\text{q}} f_j \tag{17}$$

$$k_i = W_{\text{k}} f_i \tag{18}$$

$$v_i = W_{\text{v}} f_i \tag{19}$$

$$o_j = \text{MultiHeadAttn}(q_j; \{k_i\}_i, \{v_i\}_i) \tag{20}$$

$$\text{SelfAttn}(f_j; \{f_j\}_j) = W_{\text{out}} o_j \tag{21}$$

$$\tag{22}$$

The cross-attention layer is defined similarly (see the left blue 'Cross-Attention' block in Fig. 5). The only difference to the self-attention layer is that the W_{k} and W_{v} is applied to the condition vectors (*e.g.*, the image features h in our example).

MLP layers in Transformer The Transformer model architecture applies the MLP layer (multi-layer perceptron) to do channel mixing (*i.e.*, mix the information from different feature dimensions). We follow the original transformer paper (Vaswani et al., 2017) for the implementation. The MLP layer contains two linear layers with a GELU (Hendrycks & Gimpel, 2023) activation in between. The intermediate hidden dimension is 4 times of the model dimension.

Layer Normalization We take the default LayerNorm (LN) implementation in PyTorch (Paszke et al., 2019). Besides the LN layers in ModLN as in Sec. 3.2, we follow the Pre-LN architecture to also apply LN to the final output of transformers, *e.g.*, the output of ViT and also the output of transformer decoder.

Positional Encoding The positional embedding in ViT (Dosovitskiy et al., 2020) is bilinearly upsampled from its original resolution (14x14 for input 224x224) to match our higher input resolution (32x32 for input 512x512).

B Training Setup

We specify the training setup of our LRM. Apart from the information that we provided in Sec. 4.2, we apply a cosine schedule (Loshchilov & Hutter, 2016) with 3000 warm-up iterations. We set the second beta parameter (β_2) of the AdamW optimizer (Loshchilov & Hutter, 2017) to be 0.95. We apply a gradient clipping of 1.0 and a weight decay of 0.05. The weight decay are only applied on the weights that are not bias and not in the layer normalization layer. We use BF16 precision in in the mixed precision training. To save computational cost in training, we resize the reference novel views from 512×512 to a randomly chosen resolution between 128×128 and 384×384 and only ask the model to reconstruct a randomly selected 128×128 region. With this design, we can possibly increase the effective resolution of the model.

C Analyses

We evaluate the effect of data, model hyper-parameters, and training methods on the performance of LRM, measuring by PSNR, CLIP-Similarity (Radford et al., 2021), SSIM (Wang et al., 2004) and LPIPS (Zhang et al., 2018) of the rendered novel views. Note that due to the large training cost of our final model, the following analytic experiments use a much smaller version of LRM model as the baseline (indicated by shaded rows in the tables). Specifically, we scale down the image-to-triplane decoder to 12 cross-attention layers, and change the input image resolution to 256, triplane latent dimension to 32, rendering resolution in training to 64, and use 96 samples per ray for rendering 64×64 images for supervision. We only train each model on 32 NVIDIA A100 GPUs for 15 epochs and the resulting difference can be seen in Table 1. We are aware that some observations might change if we scale up the model, but most of the conclusions should be general and consistent.

Table 1: Comparison between the final model and the baseline for analysis.

Models	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
Final	**20.1**	**91.0**	**79.7**	**16.0**
Baseline	19.0	87.8	77.4	19.1

C.1 Synthetic vs. Real Data

Table 2 compares the influence of using synthetic 3D data from the Objaverse (Deitke et al., 2023) and real video data from the MvImgNet (Yu et al., 2023) in training. Results show that removing real data causes an obvious drop for all the metrics, despite the fact our synthetic 3D dataset contains $3\times$ more shapes than MvImgNet. One potential reason is that the real data have much more variation in the lighting, the size of the target, and the camera poses, which effectively benefits the learning. Future work could augment the rendering of synthetic shapes to adequately utilize those abundant data. Nevertheless, combining the two datasets leads to substantially better results than training on any one of them alone.

C.2 Number of Views in Training Data

In Table 3, we conduct experiments with all data but limited the number of training views per shape. For example, for 'Train Views = 8', we only use a random subset of 8 views per shape. During

training, we still randomly sample 4 views from the above view subset for each training step. From the results, we can see that more views can lead to better results possibly because of more diversed data. However, the growth is saturated at 16 views but more views do not go worse.

Table 2: Influence of training datasets.

Data	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
Synthetic (Objaverse)	15.5	84.7	70.3	29.3
Real (MvImgNet)	17.5	85.7	75.7	22.0
Synthetic+Real	**19.0**	**87.8**	**77.4**	**19.1**

Table 3: Effect of the number of different views per shape in training. 32+ indicates some video data in MvImgNet contain more than 32 views per shape, which we apply all of them for training.

Train Views	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
4	18.8	86.7	77.5	19.8
8	18.9	87.3	77.5	19.4
16	**19.1**	**87.9**	**77.6**	**19.0**
32+	19.0	87.8	77.4	19.1

C.3 Model Hyper-parameters

Table 4 presents the results of having a different number of cross-attention layers in the image-to-triplane decoder. There is a slight trend indicating that the scores can be improved by having a deeper model, especially for the latent semantic and perceptual similarity measurements CLIP and LPIPS, implying that the network models better representations for reconstructing higher-quality images.

We also evaluate the influence of the number of MLP layers in NeRF (Table 5). Results show that it is unnecessary to have a very large network, and there seems to be a sweet spot around two to four layers. This observation is consistent with EG3D (Chan et al., 2022) where the information of shapes is encoded by the triplane and such MLP is only a shallow model for projecting triplane features to color and density.

As shown in Table 6, we found that increasing the triplane resolution leads to better image quality. Note that, in this experiment, we only use a deconvolution layer to upsample the $32\times32\times32$ triplane produced by LRM's decoder, whereas we suspect a large improvement could be seen by increasing the quantity of input spatial-positional embeddings to query more fine-grained image details. However, such an approach will dramatically increase the computational cost, we leave this idea to future research.

Table 4: Effect of the number of cross-attention layers in image-to-triplane decoder.

CrossAttn Layers	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
6	19.0	87.7	**77.6**	19.1
16	19.0	87.8	77.4	19.1
24	**19.1**	**88.0**	**77.6**	**18.9**

C.4 Camera Pose

As we have discussed in the Main Paper, normalizing camera poses in training has a huge impact on the generalization of input views. We can see from Table 7 that when no modification is applied (*None*), LRM produces the worst results. Augmenting camera poses with a *Random* rotation greatly improves the results since the model learns a more general image-to-triplane projection via decoupled views and camera poses. However, such unconstrained projection is very difficult to

Table 5: Effect of the number of MLP layers in NeRF.

NeRF MLP Layers	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
2	**19.2**	87.7	**77.8**	**18.9**
6	19.1	**88.0**	77.6	19.0
12	19.0	87.8	77.4	19.1
14	19.1	87.2	77.6	19.0

Table 6: Effect of the resolution of triplane. For *64up* and *128up*, we apply additional 2×2 and 4×4 deconvolution layers, respectively, to upsample a *Res.* 32 triplane.

Triplane Res.	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
32	18.9	86.3	77.2	19.7
64up	**19.0**	87.8	77.4	19.1
128up	**19.0**	**88.3**	**77.5**	**19.0**

learn. We therefore *Normalized* all camera poses so that all images are projected onto the triplane from the same direction, allowing the model to adequately learn and utilize the cross-shape prior for reconstruction.

Table 7: Effect of camera pose normalization.

Camera Pose	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
None	15.3	83.4	70.1	28.9
Random	18.0	85.6	75.7	21.1
Normalized	**19.0**	**87.8**	**77.4**	**19.1**

C.5 Image Quantity and Resolution

Table 8 and Table 9 study the influence of the number of side views supervision for each sample and the effect of image rendering resolution in training. Results indicate that as the quantity of side views increases, the reconstructed image quality improves. Having more views allows the model to better correlate the appearance and geometry of different parts of the same shape, and facilitates inferring multi-view consistent results. Moreover, using a higher rendering resolution of images in training largely improves the results, as the model is encouraged to learn more high-frequency details.

Table 8: Influence of the number of side views applied for each training sample.

Side Views	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
1	18.7	87.7	77.2	19.7
2	18.7	87.5	77.2	19.6
3	19.0	**87.8**	77.4	19.1
4	**19.1**	**87.8**	**77.6**	**18.9**

Table 9: Influence of the rendering resolution of images in training.

Render Res.	Unseen Evaluation			
	PSNR↑	CLIP-Similarity↑	SSIM↑	LPIPS↓
32	18.8	86.3	77.0	20.1
64	19.0	87.8	77.4	19.1
128	**19.4**	**89.0**	**78.3**	**18.0**

C.6 LPIPS Loss

Lastly, we found that our LPIPS objective (Zhang et al., 2018) has a huge impact on the results. Removing it from training will decrease the CLIP-Similarity, SSIM, and LPIPS scores to 74.7, 76.4, and 29.4, respectively.

D Visualizations

We present more visualizations of the reconstructed 3D shapes in the following pages. The input images include photos captured by our phone camera, images from Objaverse (Deitke et al., 2023), MvImgNet (Yu et al., 2023), ImageNet (Deng et al., 2009), Google Scanned Objects (Downs et al., 2022), Amazon Berkeley Objects (Collins et al., 2022), and images generated by the Adobe Firefly[11]. We implement a heuristic function to pre-process the camera-captured images, generated images, and images from MvImgNet and ImageNet. The function removes the image background with an off-the-shelf package[12], followed by cropping out the target object, rescaling the target to a suitable size and centering the target on a square white figure. All input images are never seen by the model in training. Please visit our anonymous webpage https://scalei3d.github.io/LRM for video demonstrations and interactable 3D meshes.

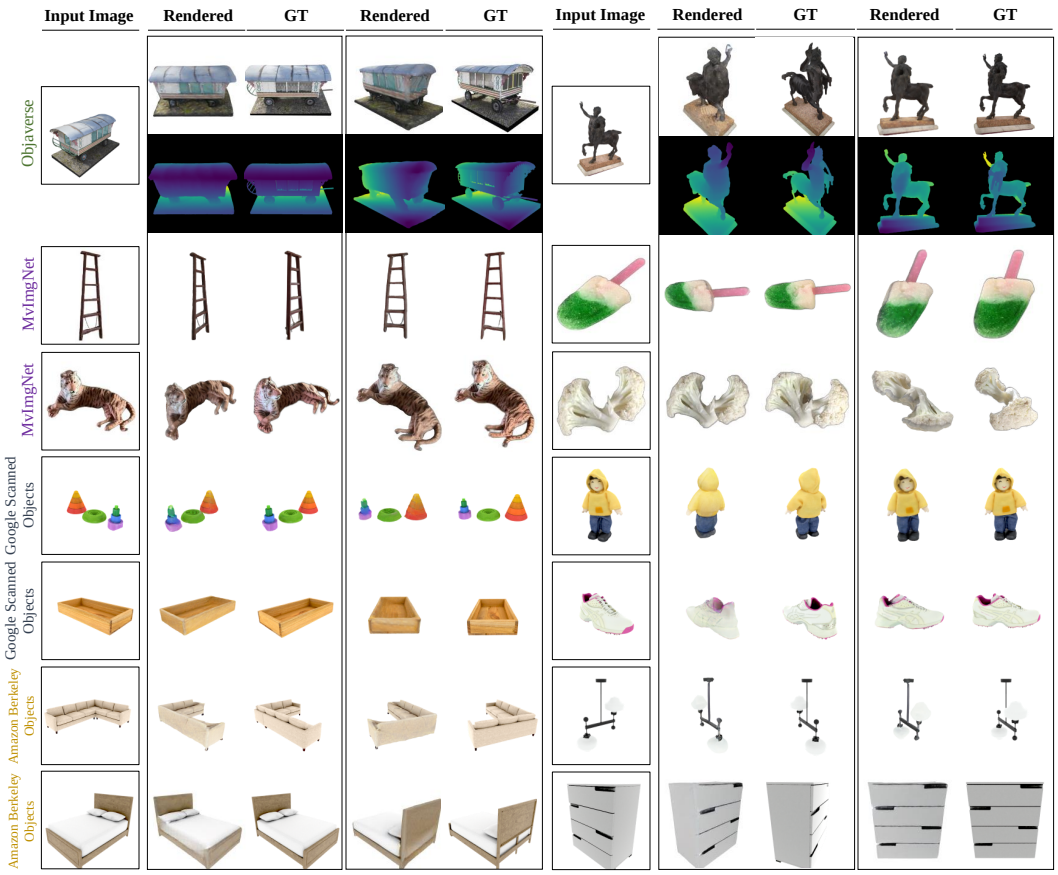

Figure 6: Comparison between LRM rendered novel views and the ground truth images (GT). None of the images are observed by the model during training. The GT depth images of Objaverse are rendered from the 3D models. Please zoom in for clearer visualization.

[11] Adobe Firefly, a text-to-image generation tool: https://firefly.adobe.com.

[12] Rembg package, a tool to remove image background: https://pypi.org/project/rembg

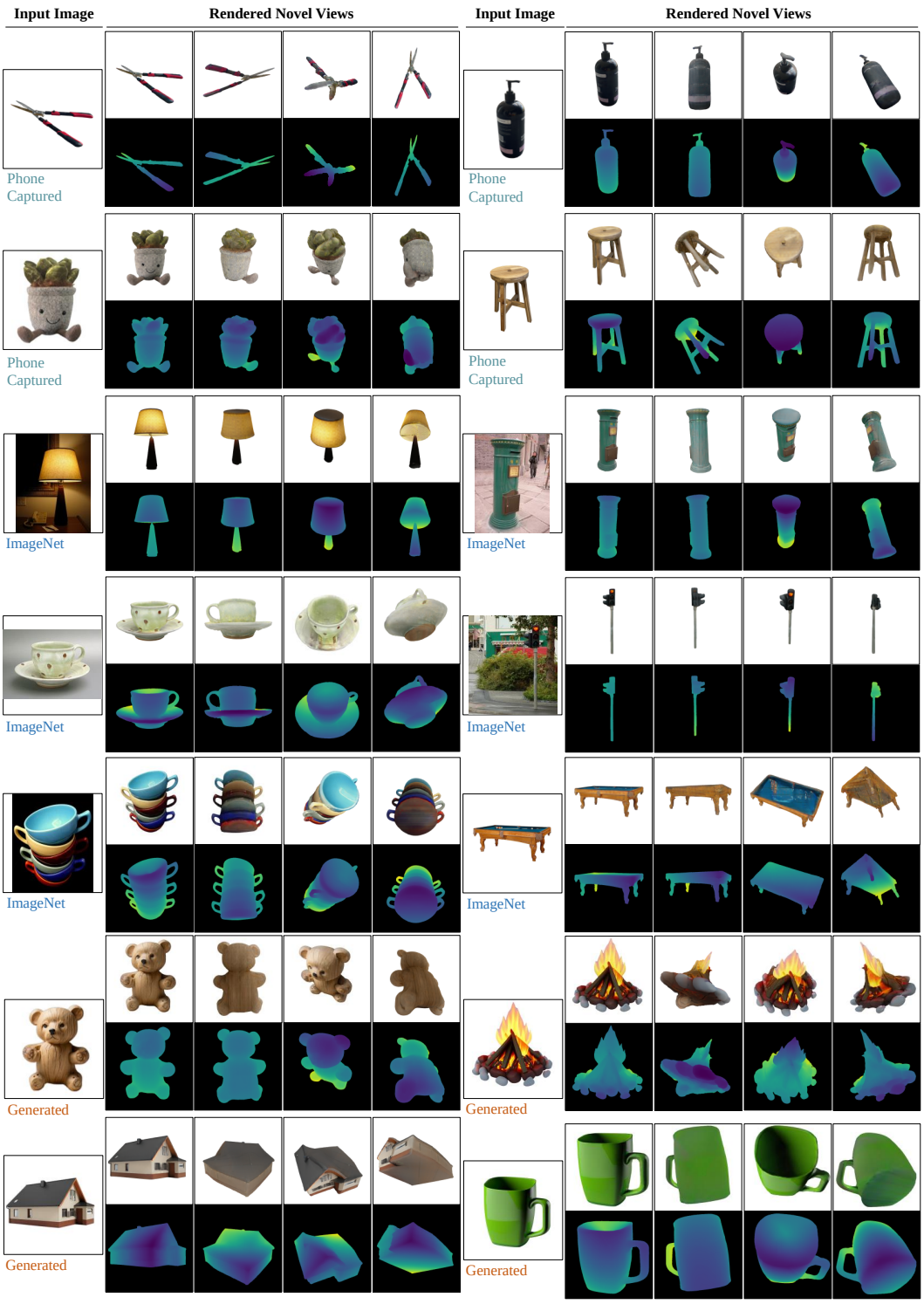

Figure 7: Rendered novel views (RGB and Depth) of shapes reconstructed by our LRM from single images. None of the images are observed by the model during training. Generated images are created by the Adobe Firefly. Please zoom in for clearer visualization.