GOEmbed: Gradient Origin Embeddings for Representation Agnostic 3D Feature Learning

Animesh Karnewar[1,2], Andrea Vedaldi[2], Roman Shapovalov[2], Tom Monnier[2], Niloy J. Mitra[1], and David Novotny[2]

[1] University College London, United Kingdom
[2] MetaAI London, United Kingdom

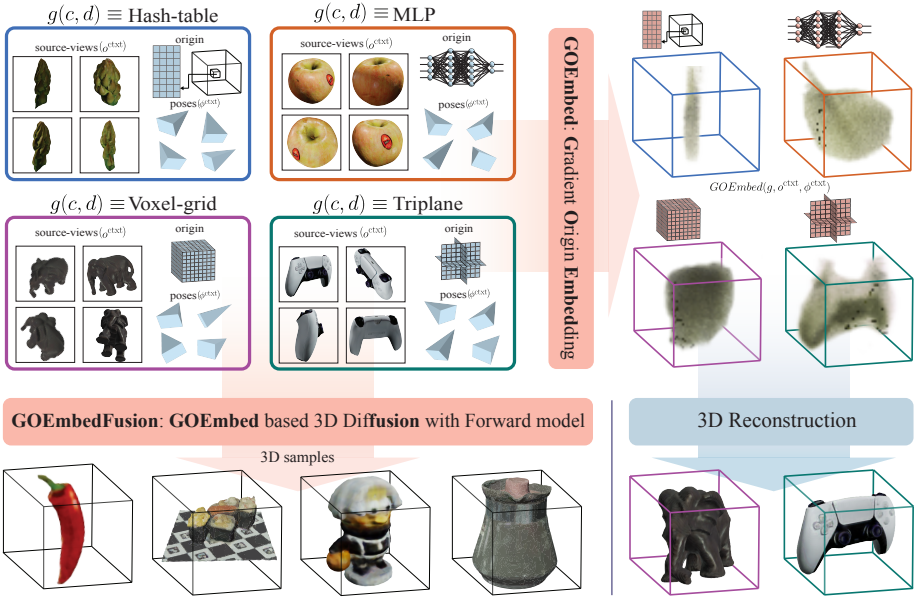

Fig. 1: We propose the **GOEmbed** (**G**radient **O**rigin **Embed**ding) mechanism that encodes source views (o^{ctxt}) and camera parameters (ϕ^{ctxt}) into arbitrary 3D Radiance-Field representations $g(c, d)$ (sec. 3). We show how these general-purpose GOEmbeddings can be used in the context of 3D DFMs (Diffusion with Forward Models) (sec. 5) and for sparse-view 3D reconstruction (sec. 6).

Abstract. Encoding information from 2D views of an object into a 3D representation is crucial for generalized 3D feature extraction and learning. Such features then enable various 3D applications, including reconstruction and generation. We propose GOEmbed: Gradient Origin Embeddings that encodes input 2D images into any 3D representation, *without* requiring a pre-trained image feature extractor; unlike typical prior approaches in which input images are either encoded using 2D features extracted from large pre-trained models, or customized features are designed to handle different 3D representations; or worse, encoders may not yet be available for specialized 3D neural representations such as MLPs

and Hash-grids. We extensively evaluate our proposed general-purpose GOEmbed under different experimental settings on the OmniObject3D benchmark. First, we evaluate how well the mechanism compares against prior encoding mechanisms on multiple 3D representations using an illustrative experiment called Plenoptic-Encoding. Second, the efficacy of the GOEmbed mechanism is further demonstrated by achieving a new SOTA FID of 22.12 on the OmniObject3D generation task using a combination of GOEmbed and DFM (Diffusion with Forward Models), which we call GOEmbedFusion. Finally, we evaluate how the GOEmbed mechanism bolsters sparse-view 3D reconstruction pipelines.

1 Introduction

The rate of progress in 3D Computer Vision research has risen in the last decade due to increased interest in various AR (Augmented Reality), VR (Virtual Reality) and MR (Mixed Reality) applications [20,64,71,72,87]. Many 3D Computer Vision problems are newly being viewed in the light of Deep-Learning based solutions. The process of encoding the information in 2D images into deep features over the chosen 3D representation can be found in the Deep-Learning solutions to almost all these problems, for instance, consider various solutions to long-standing problems such as MVS (Multi-View Stereo) [31, 53, 95], NVS (Novel-View Synthesis) or IBR (Image Based Rendering) [17, 30, 45, 76, 92], and 3D reconstruction [60, 79, 80], as well as to the nascent 3D problems such as 3D synthesis [14, 36, 73], and 3D distillation [33, 42]. Surprisingly, despite being such a critical operation, no systematic standalone study of this 2D to 3D encoding operation exists (to the best of our knowledge).

3D scenes/assets do not have a *de-facto* data representation, and different representations are utilized depending upon the requirements of the applications. For instance, just for representing Radiance Fields of static 3D assets, various neural data representations such as MLPs [4, 49, 66], Triplanes [12, 15], Feature-voxel-grids [43, 44], Hash-grids [51, 70], as well as non-neural ones like ReLU-Fields [34], Plenoxels [90], DVGo [68], 3DGS [38] are being utilized in various 3D applications. Interestingly, none of these is *foolproof* and each has certain pros-and-cons. For instance, on the one hand NeRF MLPs can very compactly ($\sim$ 5MB) represent extensive 3D spatial scenes, but on the other require excessively long times for training ($\sim$1-2 days) and rendering ($\sim$1-2 mins). Thus, given this disarray around 3D scene representations, it is a key challenge to come up with a 2D-to-3D encoding method that can: (i) generalize to arbitrary 3D representations, (ii) while being able to capture maximum information in the 3D features from the 2D images.

Existing methods of encoding can be grouped into two categories. (i) In the first category, the methods are similar to cost-volume-construction-like approaches where 2D deep features are extracted from the images, and then these 2D features are un-projected into the 3D space. On top of the deep-feature extraction network, this un-projection operation can require predicted depths using off-the-shelf depth-estimation models [5, 7, 89] for 3D representations such

as point-clouds [2, 58, 81]. For 3D representations such as feature-voxel-grids, *per-voxel-costs* in the form of variance of the per-2D-view features implicitly encodes 3D depths but requires large amount of compute-memory [53, 95]. What is further limiting is that the un-projection operation, is non-trivial and challenging to extend to specialized 3D neural representations such as MLPs and Hash-grids, and hence the approaches from this category are mostly limited to only certain 3D representations. (ii) In the second category, the methods entirely circumvent all forms of 3D inductive biases and directly inject the 2D features into the problem specific backbone networks. For example, methods like NeRF-former [59] and LRM [30] use cross-attention to directly inject 2D features into the 3D sparse-view-reconstruction backbone network. Apart from the limitation of requiring the memory-heavy cross-attention operation, given the results of our sparse-view-3D-reconstruction experiments (sec. 6), we hypothesize that these learned-feature-injection approaches may not be learning sufficient 3D-priors and are specializing only to the domains of training (albeit on large-scale).

To overcome the aforementioned limitations, we propose **GOEmbed**: **G**radient **O**rigin **Embed**dings (fig. 1) to encode the information in 2D images into any 3D representation which exists currently or which will be proposed in the future; as long as the differentiable *render* operation can be defined on it. Succinctly, the GOEmbed defines the 3D embeddings as the gradient of the mean-squared-error between renders (of the origin 3D representation) and the G.T. 2D views wrt. a predefined origin over the chosen 3D representation (fig. 2 and sec. 3). In most cases (except for MLPs due to symmetry-breaking), a simple zero-feature initialization is sufficient to define the origin. Apart from the 3D representation agnostic general-purpose applicability, our GOEmbeddings are light-weight since they do not require memory-heavy large pretrained 2D feature-extraction networks, and they try to maximize the information transfer between 2D and 3D. In summary, our contributions are:

- We propose the **GOEmbed** as a generalized encoding mechanism of 2D source views into different 3D representations (sec. 3, eq. 1).
- We propose a novel 3D diffusion pipeline by combining our GOEmbed with DFM (Diffusion with Forward Model) to achieve the state-of-the-art score on the OmniObject3D generation benchmark (sec. 5, eq. 3 - 6).
- We evaluate the efficacy of GOEmbed for extracting different 3D representations from source images in the illustrative Plenoptic Encoding experiment (sec. 4); and also evaluate its utility in the sparse-view 3D reconstruction task (sec. 6).

2 Related Work

Our proposed method is primarily an attempt at a systematic study of 2D-to-3D encoding mechanism, thus we firstly cover the prior encoding methods in subsec. 2.1. Secondly, since our method can encode 2D into arbitrary 3D representations, we cover the works related to neural 3D scene representations in subsec 2.2. And lastly, since we propose a new realisation of the DFM framework, we cover prior

3D generative modeling works in subsec 2.3. Please refer to the survey works of Xie et al. [87] and Shi et al. [64] for more exhaustive coverage on 3D neural representations and 3D generative models respectively.

2.1 2D-to-3D Encoding

Most of the 3D encoding mechanisms proposed till now have been in the context of a larger problem such as MVS or NVS, and there has also not been a standalone principled study of the encoding mechanisms yet. Nevertheless, the early works [23] constructed pixel-disparity based 3D cost-volumes for MVS (Multi-View Stereo) problems. These raw-pixel based cost-volumes were soon superseded by 2D deep image feature based ones [31, 33, 36, 53, 95] due their memory-compactness and information expressiveness. These approaches typically proceed as: first obtain per-image-per-pixel features using large pretrained image networks such as ResNet [25] or DinoV2 [56]; then these features are un-projected into the 3D space using the camera parameters associated with the views. The un-projected features are then accumulated into the feature-voxel grid yielding the cost-volume to be used in various 3D problem contexts. Apart from voxel-grids, recent works also splat these features either on Triplanes [24], or on the surface of proxy meshes. The most similar work to our proposed GOEmbed in terms of idea is the one from Bond-Taylor and Willcocks titled GON (Gradient Origin Networks) [8], although their proposal has no context of 3D encodings. We take inspiration from GONs, but our proposed GOEmbedings are different from them in that: (i) while the purpose of GON is to obtain a compressed latent space, our GOEmbed is aimed at obtaining a coarse partial estimate of the plenoptic 3D scene given 2D image observations (sec. 3); and (ii) our GOEmbed encodings are much more local compared to the latent embeddings obtained using the GON. Please refer to the section 1.1 of the supplementary for a discussion of GON.

2.2 3D Neural Representations

Neural Fields movement in the data representations research was spear-headed by the works such as OccupancyNetworks [55] and SRNs [66] that proposed to use MLPs for representing the 3D shapes as occupancy fields and SDFs (signed-distance fields) respectively. But, most notably, the work NeRF (Neural Radiance Fields) [49] made this idea popular by demonstrating the use of MLPs in the context of an application as complex as 3D Novel-view-synthesis. Concurrently, another important work called SIREN [65] showed that sinusoid-activated MLPs could also represent images, videos, MRIs, etc. The research progress exploded after these two works, leading to various works [35, 69, 90, 91] proposing voxel-based modifications to the NeRF scene representation; others [21,52,75] speeding up the training and inference of NeRFs. Numerous works such as Acorn [47], NSVF [41], Instant-NGP [52], Variable Bitrate Neural Fields [70], TensoRF [16], and Triplanes [11] mark some of the key hybrid explicit-implicit versions of Neural 3D scene representations.

2.3 3D Generative Models

The earliest forms of deep learning based 3D generative models can be traced back to 3D-GAN [82]; where prior non-learning based methods of 3D shape reconstruction or synthesis involved template based permutation approaches [32]. Methods evolved to PlatonicGAN [26] and PrGAN [18] that generates 3D shapes from an unstructured corpus of uni-category images. More interesting is the approach of HoloGAN [54] that adds a 3D inductive bias in the form of a 3D feature-grid and a neural renderer to the conventional generator architecture of 2D GAN. Utilizing the differentiable Emission-Absorption renderer component of the NeRF [49], works like GRAF [61], VoxGRAF [62], and Pi-GAN [13] train 3D GANs with better view-consistency using category-specific image datasets. Orthogonal to these approaches, StyleNeRF [22] proposed a hybrid approach which utilised a core-3D graf-like base model, followed by a HoloGAN-like neural renderer to upsample the rendered resolution. Surprisingly, due to carefully crafted architecture and strong regularization, StyleNeRF generated samples hold formidable 3D view-consistency. Simultaneously various works [37, 39, 67, 83, 84, 94] proposed GAN and Diffusion based 3D generative models on single-3D scene setting. This progress continued steadily till the recent milestone works in 3D generative models such as EG3D [12], which trains a 3D GAN on newly proposed 3D Triplane neural representation; GET3D [19], which uses DMTet [63] module in the GAN to directly generate 3D textured meshes; DiffRF [50], which proposed the first preliminary 3D diffusion model on 3D voxels similar to 3D-GAN [82]; works such as RenderDiffusion [3], HoloDiffusion [36], DFM [73] which train 3D diffusion models only using 2D images; until the most recent DiffTF [10] that adapted the DiffRF training pipeline to Triplanes and improved it further to achieve the state-of-the art results on the OmniObject3D [86] dataset.

3 Method

3.1 GOEmbed: Gradient Origin Embeddings

Let $g(c, d)$ represent a static 3D scene as a Radiance-Field such that $c = [x, y, z]$ denotes the 3D coordinates of a point in the Euclidean space, $d = [\theta, \gamma]$[3] denotes the spherical polar coordinates of an outgoing direction from the point, and the function $g : \mathbb{R}^5 \rightarrow \mathbb{R}^4$ maps each 3D coordinate and a particular outgoing direction to four values (σ, R, G, B). Here, σ represents density and $[R, G, B]$ represents the outgoing radiance. Let $\mathcal{R}(g, \phi)$ denote the rendering *functional* that converts the Radiance-Field function into an image of a certain resolution from a certain camera viewpoint, as described by the camera parameters ϕ. We use the differentiable Emission-Absorption Volume Raymarching algorithm for

[3] We deviate from the more common use of $[\theta, \phi]$ for spherical polar coordinates in favour of $[\theta, \gamma]$ to avoid confusion with ϕ^{ctxt} and ϕ^{trgt} used to denote context and target camera parameters respectively.

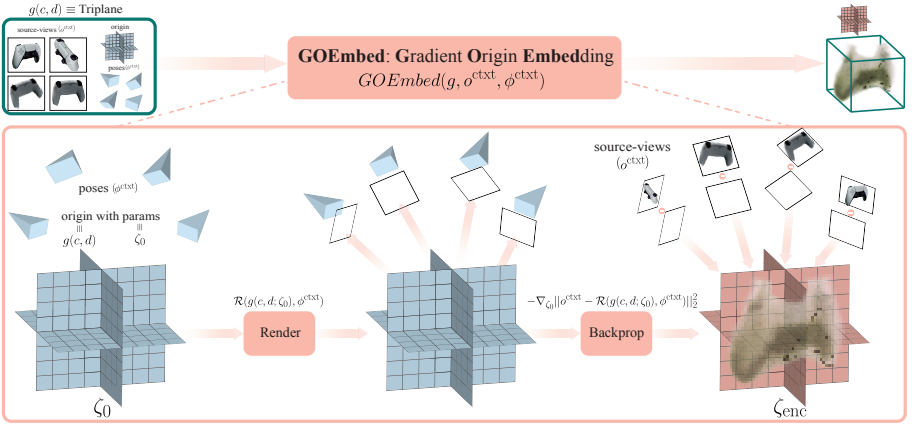

Fig. 2: GOEmbed illustration. We demonstrate the mechanism here using the Triplane representation for $g(c, d)$, but note that this can be applied to other representations as well. The GOEmbed mechanism (eq. 1) consists of two steps. First we render the origin ζ_0 from the context-poses ϕ^{ctxt}; then we compute the gradient of the MSE between the renders and the source-views o^{ctxt} wrt. the origin ζ_0 which gives us the GOEmbed encodings ζ_{enc}.

rendering [48,49]. For the encoding mechanism to be unified and general-purpose, it must have the following three properties:

(i) It should be able to encode one or more views alike.
(ii) It should generalize to any parameterization/realization of the function g, i.e., the encoding mechanism should be applicable irrespective of whether the Radiance-Field is represented as an MLP [49], a Hash-grid [51], a Triplane [12,77], or a Voxel-grid [35,69,90].
(iii) It should try to maximize the information transfer from the 2D views to the 3D embedding.

We introduce the Gradient Origin Embeddings, where we define the encoding of the observations (2D views) as the gradient of the mean-squared error between the rendered and ground truth 2D views with respect to a predefined origin (zero vector or features) 3D representation.

Without any loss of generality, assuming that ζ are the parameters of the g function (i.e., the features/weights of the 3D Radiance-Field) and ζ_0 denotes the origin (zero features/weights), we define the encodings ζ_{enc} as (fig. 2):

$$\zeta_{\text{enc}} := GOEmbed(g, o^{\text{ctxt}}, \phi^{\text{ctxt}}) \\ \boxed{:= -\nabla_{\zeta_0}||o^{\text{ctxt}} - \mathcal{R}(g(c,d;\zeta_0),\phi^{\text{ctxt}})||_2^2} \tag{1}$$

where, o^{ctxt} and ϕ^{ctxt} are the G.T. 2D views and their camera parameters, respectively, which are to be encoded into the 3D representation ζ_{enc}. Note that

the proposed encoding function GOEmbed backpropagates through the differentiable rendering functional $\mathcal{R}$. By construction, such an encoding satisfies the formerly stated properties (i) and (ii). Further, we minimize the loss function

$$\mathcal{L}^{\text{GOEmbed}}(o^{\text{ctxt}}, o^{\text{trgt}}) := ||o^{\text{ctxt}} - \hat{o}^{\text{ctxt}}||_2^2 + ||o^{\text{trgt}} - \hat{o}^{\text{trgt}}||_2^2 \\ \text{where, } \hat{o} = \mathcal{R}(g(c, d; \zeta_{\text{enc}}), \phi), \tag{2}$$

for maximizing the information content in the encodings ζ_{enc} to satisfy property (iii). Here, ϕ^{ctxt} are the context views used for encoding while ϕ^{trgt} are some target views of the same scene but different from the source ones. Intuitively, we repurpose the backward pass of the rendering functional to encode the information in the source views o^{ctxt} into the parameters of the 3D scene representation ζ. Thus, as long as a mathematically differentiable rendering operator is possible on it, any 3D scene representation can be encoded using our GOEmbed encoder.

3.2 Experimental Evaluation Rubric

We evaluate the generality of GOEmbed mechanism along three axes: firstly, to measure the information transfer, we run experiments in a Plenoptic Encoding setting (sec. 4); secondly, we train the GOEmbedFusion model to learn a 3D generative model using 2D images to evaluate its 3D generative capability (sec. 5); and finally, we also evaluate the GOEmbed mechanism in a sparse-view 3D reconstruction setting (sec. 6).

Dataset. We perform all our experiments on the recently released OmniObject3D dataset [86] containing ∼6K 3D scans of real world objects from daily life. The dataset contains a large-vocabulary of ∼200 categories of daily life classes having some overlap with COCO [40]. Only for our non-forward diffusion baseline, we also use the text-captions recently released by the OmniObject3d authors at their GitHub page [85].

Metrics. For the experiments in the Plenoptic-Encoding setting (sec. 4), we use the standard image reconstruction metrics **PSNR**, **LPIPS** [93] and **SSIM** [78], while for the quantitative analysis in generative modeling experiments (sec. 5) we use the **FID** [27] and **KID** [6] metrics following prior works. And, lastly for the 3D reconstruction experiments (sec. 6) we again use the standard image reconstruction metrics similar to the Plenoptic-Encoding experiments. We color code all the scores as first, second, and third.

4 Plenoptic Encoding

To evaluate the information transfer from the source views to the encoded 3D representation, we train the standalone GOEmbed component on its own before

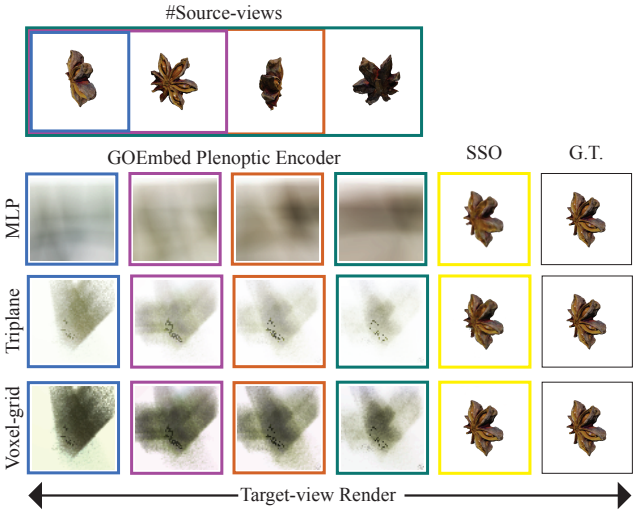

Fig. 3: Plenoptic Encoding Qualitative Evaluation. The rows MLP, Triplane and Voxel-grid show the renders of the GOEmbed encoded representations from the target-view respectively. The colour-coded columns demonstrate the effect of varying the number-of-source views (1, 2, 3, 4) used in the GOEmbed encoding. The SSO column shows the target render of the single-scene-overfitted representation while the G.T. column shows the mesh-render from the dataset (repeated for clarity).

Table 1: Plenoptic Encoding Quantitative Evaluation. PSNR(↑), LPIPS(↓) and SSIM(↑) reported on three different representations of the 3D Radiance-Field g, namely, Triplanes, Voxel-Grids and MLPs. All the metrics are evaluated for target views (different from the source views) against the G.T. mesh renders from the dataset. The SSO (Single Scene Overfitting) scores denote the case of individually fitting the representations to the 3D scenes.

Method	Triplane			Voxel-Grid			MLP		
	PSNR (↑)	LPIPS (↓)	SSIM (↑)	PSNR (↑)	LPIPS (↓)	SSIM (↑)	PSNR (↑)	LPIPS (↓)	SSIM (↑)
DinoV2 1 source-view	14.182	1.286	0.425	14.419	1.151	0.485	N/A	N/A	N/A
DinoV2 2 source-views	14.482	1.235	0.435	15.330	0.925	0.526	N/A	N/A	N/A
DinoV2 3 source-views	14.664	1.213	0.430	15.685	0.864	0.546	N/A	N/A	N/A
DinoV2 4 source-views	14.711	1.199	0.425	15.834	0.857	0.551	N/A	N/A	N/A
(Ours) 1 source-view	15.711	1.025	0.477	15.514	1.008	0.479	11.208	1.124	0.496
(Ours) 2 source-views	15.513	1.068	0.468	15.319	1.067	0.480	11.406	1.197	0.489
(Ours) 3 source-views	15.672	1.129	0.456	15.590	1.110	0.484	11.490	1.202	0.500
(Ours) 4 source-views	15.919	1.150	0.472	15.755	1.145	0.486	11.599	1.195	0.505
SSO	28.165	0.087	0.941	32.061	0.067	0.9582	27.382	0.119	0.918

using it in different contexts. Specifically, given the dataset $D = \{(\mathcal{I}_i^j, \phi_i^j) | i \in [1, N]$ and $j \in [1, C]\}$ of N 3D scenes where each scene contains C images and camera parameters, we define the Plenoptic Encoding as a mechanism which encodes k source views and camera parameters, of a certain 3D scene, into the representation g (whose parameters are ζ). The encoded scene representation

should be such that the rendered views from the same source cameras, and some l different target cameras, should be as close as possible to the G.T. images $\mathcal{I}$, i.e., the PlenopticEncoder $PE : \mathbb{R}^{k\times h\times w\times c} \times \mathbb{R}^{k\times 4\times 4} \rightarrow \mathbb{R}^{(k+l)\times h\times w\times c}$ should minimize the following mean squared error objective:

$$\mathcal{L}^{\text{PE-MSE}} := \mathbb{E}_{(\mathcal{I},\phi)\sim D} \|\mathcal{I} - PE(\mathcal{I}, \phi)\|_2^2.$$

Although this experimental setup is quite similar to typical MVS/NVS or 3D reconstruction or 3D prior learning setups, in form and essence; we note here that the plenoptic encoder PE is neither targeted to do Multi-View Stereo nor 3D reconstruction. To set the correct expectations, we note that the main and the only goal here is to evaluate how much information is transferred from the 2D images to the chosen 3D representation. We evaluate the GOEmbed encodings on three different 3D Radiance-Field representations, g, namely Triplanes [12], Feature-Voxel grids [35, 36, 69, 90], and MLPs [49] and compare them to cost-volume like approaches where possible.

All the evaluations in Table 1 are done on 100 randomly chosen *test* scenes from the OmniObject3D dataset on 1, 2, 3, and 4, number of source views. Although the loss is optimized on both source and target camera views, the reported scores are for *target-views* only, since we are interested in measuring the "3D" encoding ability, as opposed to overfitting the source views into the representation. If we consider only the source-view metrics, there is a possibility of the degenerate solution where the encoder copies all the source views as-it-is disregarding any 3D structure. Also, all the scores are computed against the renders of the G.T. meshes, but since some quality can be lost by the choice of the representation itself, the SSO (Single-Scene-Overfitting) version, where we fit the representation directly to the scene, is also provided for a more grounded comparison.

Table 1 summarizes the scores obtained in this setup of experiments. We compare against cost-volume construction approach using DinoV2 [56], where we first un-project the per-view image features into a feature-voxel grid and then accumulate the per-view features into the cost-volume similar to StereoMachines [31] and many others [17, 36, 42, 53, 95]. We further splat the voxel-features into the mutually orthogonal planes in case of the Triplanes based cost-volume baseline. As apparent from the table, GOEmbed outperforms the DinoV2 cost-volume-approach for Triplanes and comes very close to the Voxel-Grids one. Note that the cost-volume approach is not trivial to come-up with for MLPs, whereas our GOEmbed can handle this representation by design and obtains formidable scores in this setup. Apart from this advantage, GOEmbed only has learnable parameters ($\sim$17K) as part of the renderer functional $\mathcal{R}$ (usually an MLP), while the DinoV2 has excess of $\sim$1B parameters, albeit pretrained. Finally, what is most surprising is that our light-weight GOEmbed approach is actually able to obtain almost 50% of the SSO scores which is the practically achievable performance for the chosen 3D representations (last row of table 1). Figure 3 illustrates this phenomenon visually.

5 3D Generation

Fig. 4: 3D Generation Qualitative Evaluation. 3D samples generated by our GOEmbedFusion compared to the prior GAN, and Diffusion based baselines.

Although prior works like RenderDiffusion [3] and HoloDiffusion [36] were already proposed to train 3D diffusion models using only 2D images, recently, the notable work of Tewari et al. [73] proposed a unified mathematical framework for the *stochastic-inverse* setting of generative modeling where we only have access to partial observations of the underlying ground-truth signals, but never the underlying signals themselves; in which 3D inverse graphics is a special case. Please refer to section 1.2 of the supplementary for a discussion on the mathematical framework of DFM (Diffusion with Forward Models). Despite this, the proposed vanilla realization of the DFM framework for 3D generative modeling has certain limitations; first, it is only specific to the feature-point-cloud 3D representation and does not generalize to other 3D representations; second, it cannot generate purely unconditional 3D samples since it is a 2D view conditioned 3D diffusion model; and third, it requires a computationally expensive auto-regressive process for sampling the underlying 3D scenes.

We propose a novel realization of the DFM framework called GOEmbedFusion where our proposed GOEmbed drives the diffusion with forward model training pipeline; while overcoming all the limitations as mentioned earlier. Our unobserved samples here correspond to the ζ parameters since we are interested

in modeling the generative probability distribution over the 3D Radiance-Fields. A single forward pass through our proposed GOEmbedFusion pipeline is defined through the following equations:

$$\zeta_{\text{enc}} := GOEmbed(g, o^{\text{ctxt}}, \phi^{\text{ctxt}}) \tag{3}$$

$$\hat{\zeta}_0 := \mathcal{D}_\theta(\zeta_T, T; \zeta_{\text{enc}}) \tag{4}$$

$$\hat{\zeta}_t \sim q(\hat{\zeta}_t|\hat{\zeta}_{t-1}) = q(\hat{\zeta}_0) \prod_{k=1}^{t} q(\hat{\zeta}_k|\hat{\zeta}_{k-1}) \tag{5}$$

$$\hat{o}^{\text{trgt}} := \mathcal{R}(g(c, d; \mathcal{D}_\theta(\hat{\zeta}_t, t; \zeta_{\text{enc}})), \phi^{\text{trgt}}), \tag{6}$$

where T corresponds to the highest timestep and $\zeta_T \sim \mathcal{N}(0, I)$ is a sample of the standard Gaussian noise distribution. Equation 3 first encodes the context observations and their parameters $(o^{\text{ctxt}}, \phi^{\text{ctxt}})$ into the GOEmbed encoding ζ_{enc} using eq. 1. Then, eq. 4 uses the denoiser network to predict a pseudo-deterministic estimate of the clean 3D scene $\hat{\zeta}_0$ conditioned on the GOEmbed encodings ζ_{enc}. While the noise can impart some degree of stochasticity to the output, the network can, in theory, completely ignore the noise. We let the network learn what to do based on the training and the data complexity, and hence call this step "pseudo"-deterministic. This is followed by obtaining a noisy version of the 3D scene $\hat{\zeta}_t$ through standard forward diffusion corruption (eq. 5). We estimate the process of drawing from the distribution $q(\hat{\zeta}_t|\hat{\zeta}_{t-1})$ using the standard DDPM [28] closed-form equation $\hat{\zeta}_t = \sqrt{\alpha_t}\hat{\zeta}_0 + \sqrt{1-\alpha_t}\epsilon$, where ϵ is pure Gaussian noise and α_t denotes the schedule of diffusion. And finally, we predict any target observations by rendering the output of the denoiser network $\mathcal{D}_\theta$ for $\hat{\zeta}_t$ conditioned on the GOEmbed encodings ζ_{enc} and the timestep t using eq. 6.

We note here that the network $\mathcal{D}_\theta$ operates in the `x_start` diffusion formulation in contrast to the popular `epsilon` formulation. We require a few different loss functions to be able to train this pipeline end-to-end such that each component does what it is supposed to:

$$\mathcal{L}^{\text{GOEmbedFusion}} := \mathbb{E}_{t,o^{\text{trgt}}} \|o^{\text{trgt}} - \hat{o}^{\text{trgt}}\|_2^2$$

$$\mathcal{L}^{\text{PSE-DET}} := \mathbb{E}_{t,o^{\text{trgt}}} \|o^{\text{trgt}} - \mathcal{R}(g(c, d; \hat{\zeta}_0)), \phi^{\text{trgt}})\|_2^2.$$

The final objective is simply the sum of the three loss functions as,

$$\mathcal{L}^{\text{total}} := \mathcal{L}^{\text{GOEmbedFusion}} + \mathcal{L}^{\text{PSE-DET}} + \mathcal{L}^{\text{GOEmbed}}.$$

The loss function $\mathcal{L}^{\text{GOEmbed}}$ is required to maximize the information content in the GOEmbed encodings. In contrast, the $\mathcal{L}^{\text{PSE-DET}}$ tries to maximize the reconstruction quality of the pseudo-deterministic output $\hat{\zeta}_0$. The $\mathcal{L}^{\text{GOEmbedFusion}}$ loss function actually trains the GOEmbedFusion pipeline. As apparent, the GOEmbed mechanism enables the GOEmbedFusion model to use any parameterization of the g function as long as it allows differentiable rendering and can define a zero-origin over its parameters ζ. Please refer to the figure 1 in the supplementary

material for an illustration of the GOEmbedFusion model. Empirically, we found that the two-step bootstrapped-denoising as done in eqn. 1 and eqn. 6, similar to prior works [33, 36], is crucial to correctly train this diffusion pipeline in the forward setting. In practice, we train the model using the classifier free guidance training scheme [29], where we dropout the GOEmbed conditioning randomly with a probability of 0.5, to allow for both conditional and unconditional sampling. Lastly, samples can be generated by iterative denoising using the trained model $\mathcal{D}_\theta$ either unconditionally or conditionally using the GOEmbed encodings of certain context observations. Here, we can directly sample (denoise) in the space of the unobserved underlying data samples ζ, eliminating the need for the expensive auto-regressive fusion required by the vanilla realization.

Table 2: 3D Generation Quantitative Evaluation. FID(↓) and KID(↓) scores on the OmniObject3D dataset comparing our GOEmbedFusion with GAN baselines EG3D [12], and GET3D [19]; with the non-forward diffusion baselines DiffRF [50], DiffTF [10], and Our non-forward diffusion baseline; and, with the DFM (Diffusion with Forward Model) [73].

Method	FID (↓)	KID (↓)
EG3D [12]	41.56	1.0
GET3D [19]	49.41	1.5
DiffRF [50]	147.59	8.8
DiffTF [10]	25.36	0.8
DFM [73]	73.51	3.8
Ours non-forward	119.67	8.0
GOEmbedFusion (Ours)	22.12	0.6

As shown in table 2, our GOEmbedFusion sets the new state-of-the-art FID and KID scores on the OmniObject3D dataset in comparison to the prior state-of-the-art DiffTF [10], our implementation of a DiffTF like baseline called Ours non-forward, the DFM [73] model, the DiffRF [50] model, and the prior GANs EG3D [11] and GET3D [19]. We note that our improvements are not only limited to the quality of generation, but also to the benefits provided by our GOEmbedFusion formulation over the vanilla realisation of the DFM (row 5 table 2). While DiffTF [10] uses many architectural modifications and other tricks specific to Triplanes and the OmniObject3D dataset, we only use the base DiT [57] architecture with *no* modifications as our backbone denoiser network $\mathcal{D}_\theta$. Also our proposed GOEmbedFusion training pipeline is a diffusion with forward model, and hence can be trained only using 2D images, unlike DiffTF which first fits ground-truth 3D Triplanes and then trains the diffusion model. Our qualitative samples in figure 4 further support these arguments.

6 Sparse-View 3D Reconstruction

The Plenoptic Encoding experiments (sec. 4) are in an illustrative setup and provide insights into how well our propsed GOEmbed can transfer information from 2D images into various different 3D representations such as Triplanes, MLPs and Voxel-Grids. In this section, we aim to evaluate the utility of the GOEmbeddings in a practical application setup. Thus, although the mathematical experimental setup is similar to the Plenoptic Encoding one, here we input the obtained GOEmbeddings to a backbone sparse-view-3D-reconstruction network. Specifically, we use the DiT based transformer network, which is 12-layers wide as the reconstruction backbone. The end-to-end pipeline is supervised with exactly the same losses as that of the Plenoptic Encoding setup. Intuitively, our GOEmbed reconstruction model replaces the input learned triplane positional encodings with our GOEmbeddings and removes the cross-attention layers from the LRM (Large Reconstruction Model) architecture. Additionally, we also evaluate the reconstruction capability of our GOEmbedFusion model by running the pipeline only till the "pseudo"-deterministic output stage (i.e., eqn. 4).

We compare this transformer-based reconstruction setup of ours to the most recent baseline of LRM (Large Reconstruction Model) [30], which is also based on the transformer architecture. Since LRM's code has not been published, we implement this baseline in our code-base as close to the paper-description as possible, for a fair comparison. Specifically, we train two versions of the LRM, the first one which is 16-layers wide (the base published model), and second smaller version which is 6-layers wide. We introduced the 6-layers version since we found the base-model to be overfitting to the OmniObject3D training subset, which is much smaller in scale than the dataset on which LRM is trained. All the learned network baselines are evaluated on the $\sim$ 200 test scenes of the OmniObject3D dataset while the Triplane-SSO (from the Plenoptic Encoding section 4) is evaluated on 100 scenes. These 100 SSO scenes form a proper subset of the test-set, so the comparison here is fair. Please check the supplemental for more details of our GOEmbed reconstruction architecture.

Table 3: 3D reconstruction Quantitative Evaluation. PSNR(↑), LPIPS(↓) and SSIM(↓) of our GOEmbed reconstruction model, and GOEmbedFusion's "pseudo"-deterministic 3D reconstruction output compared to LRM baselines. We again include the SSO (Single Scene Overfitting) here for comparison.

Method	PSNR (↑)	LPIPS (↓)	SSIM (↑)
LRM (Our) 6 layer	23.788	0.119	0.827
LRM (Our) 16 layer	23.247	0.121	0.813
GOEmbed recon	27.650	0.109	0.900
GOEmbedFusion (PSE-DET)	26.447	0.119	0.890
Triplane *SSO*	28.165	0.087	0.941

As summarized in the table 3, both our GOEmbed reconstruction model as well as the diffusion based GOEmbedFusion model outperforms the LRM baselines. Also, our reconstruction model gets quite close to the practical performance limit as set by the Triplane-SSO experiment. Thus, given these results, we can assert that our proposed GOEmbeddings generalise to larger datasets like OmniObject3D and can strengthen the sparse-view-3D-reconstruction pipelines. Please refer to the section 3 of the supplemetary material for more details and for qualitative samples of these experiments.

7 Limitations, Future Scope, and Societal Impact

Although our framework applies theoretically to any arbitrary 3D Radiance-Field representations, its use in the 3D generative modeling is restricted by the representation's compatibility with existing denoiser network architectures. We believe that Transformers [74] get close to being universally applicable, but it still remains a challenge to adapt certain 3D representations such as Hash-grids [51] as input to Transformers. Nevertheless, we believe that our proposed GOEmbed mechanism makes a substantial research stride and will inspire further interesting applications and theoretical insights. Apart from this, upon close qualitative examination, we find that the samples generated using our GOEmbedFusion model have some peculiar checkerboard artifacts. Similar to the findings of the recent Denoising-ViT [88], we hypothesize that these artifacts in our model are also because of the positional encodings in the DiT architecture. Finding the exact reason for these artifacts and getting rid of them constitutes a future direction to be pursued.

Although our proposed method primarily contributes to general-purpose 3D feature extraction and learning, as shown in our 3D generation experiments, our method could be applied in the context of the generative modeling of real-captured or synthetic 3D assets. Hence, similar to the case of 2D generative models, our proposed GOEmbedFusion model is also prone to the misuse of the synthetically generated media. We note that there is a potential for our method to be used in the creation of fake 3D-consistent videos.

8 Conclusion

We presented the GOEmbed as a general-purpose mechanism for encoding the information in 2D images into arbitrary 3D scene representations and evaluated its information transfer ability with the Plenoptic Encoding experiments. We show that the encodings can be applied to Triplanes, Voxel-grids and MLPs, but note that exploring these in the context of other popular 3D representations (e.g., meshes, point clouds) forms scope for future work. The GOEmbeddings find a practical use in the context of improving the framework of DFM models, which we demostrate through the 3D generation experiments on the OmniObject3D benchmark; and in the context of sparse-view 3D reconstruction as well.

References

1. Guided-diffusion github code (2021), https://github.com/openai/guided-diffusion 5
2. Aliev, K.A., Sevastopolsky, A., Kolos, M., Ulyanov, D., Lempitsky, V.: Neural point-based graphics. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16. pp. 696–712. Springer (2020) 3
3. Anciukevicius, T., Xu, Z., Fisher, M., Henderson, P., Bilen, H., Mitra, N.J., Guerrero, P.: RenderDiffusion: Image diffusion for 3d reconstruction, inpainting and generation. arXiv.cs **abs/2211.09869** (2022) 5, 10
4. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5855–5864 (2021) 2
5. Bhat, S.F., Birkl, R., Wofk, D., Wonka, P., Müller, M.: Zoedepth: Zero-shot transfer by combining relative and metric depth. arXiv preprint arXiv:2302.12288 (2023) 2
6. Bińkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying mmd gans. arXiv preprint arXiv:1801.01401 (2018) 7
7. Birkl, R., Wofk, D., Müller, M.: Midas v3. 1–a model zoo for robust monocular relative depth estimation. arXiv preprint arXiv:2307.14460 (2023) 2
8. Bond-Taylor, S., Willcocks, C.G.: Gradient origin networks. arXiv preprint arXiv:2007.02798 (2020) 4, 1, 2
9. Brito, A.: Blender 3D. Novatec New York (2007) 4
10. Cao, Z., Hong, F., Wu, T., Pan, L., Liu, Z.: Large-vocabulary 3d diffusion model with transformer. arXiv preprint arXiv:2309.07920 (2023) 5, 12
11. Chan, E.R., Lin, C.Z., Chan, M.A., Nagano, K., Pan, B., De Mello, S., Gallo, O., Guibas, L.J., Tremblay, J., Khamis, S., 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) 4, 12
12. Chan, E.R., Lin, C.Z., Chan, M.A., Nagano, K., Pan, B., Mello, S.D., Gallo, O., Guibas, L.J., Tremblay, J., Khamis, S., Karras, T., Wetzstein, G.: Efficient geometry-aware 3D generative adversarial networks. In: Proc. CVPR (2022) 2, 5, 6, 9, 12
13. Chan, E.R., Monteiro, M., Kellnhofer, P., Wu, J., Wetzstein, G.: pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5799–5809 (2021) 5
14. Chan, E.R., Nagano, K., Chan, M.A., Bergman, A.W., Park, J.J., Levy, A., Aittala, M., Mello, S.D., Karras, T., Wetzstein, G.: GeNVS: Generative novel view synthesis with 3D-aware diffusion models. In: arXiv (2023) 2
15. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: TensoRF: Tensorial radiance fields. In: arXiv (2022) 2
16. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. In: European Conference on Computer Vision (ECCV) (2022) 4
17. Chen, A., Xu, Z., Zhao, F., Zhang, X., Xiang, F., Yu, J., Su, H.: Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14124–14133 (2021) 2, 9

18. Gadelha, M., Maji, S., Wang, R.: 3D shape induction from 2D views of multiple objects. In: arXiv (2016) 5
19. Gao, J., Shen, T., Wang, Z., Chen, W., Yin, K., Li, D., Litany, O., Gojcic, Z., Fidler, S.: Get3d: A generative model of high quality 3d textured shapes learned from images. In: Advances In Neural Information Processing Systems (2022) 5, 12
20. Gao, K., Gao, Y., He, H., Lu, D., Xu, L., Li, J.: Nerf: Neural radiance field in 3d vision, a comprehensive review. arXiv preprint arXiv:2210.00379 (2022) 2
21. Garbin, S.J., Kowalski, M., Johnson, M., Shotton, J., Valentin, J.P.C.: FastNeRF: High-fidelity neural rendering at 200FPS. In: arXiv. vol. abs/2103.10380 (2021) 4
22. Gu, J., Liu, L., Wang, P., Theobalt, C.: Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. arXiv preprint arXiv:2110.08985 (2021) 5
23. Gu, X., Fan, Z., Zhu, S., Dai, Z., Tan, F., Tan, P.: Cascade cost volume for high-resolution multi-view stereo and stereo matching. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2495–2504 (2020) 4
24. Gupta, A., Xiong, W., Nie, Y., Jones, I., Oğuz, B.: 3dgen: Triplane latent diffusion for textured mesh generation. arXiv preprint arXiv:2303.05371 (2023) 4
25. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. arXiv preprint arXiv:1512.03385 (2015) 4
26. Henzler, P., Mitra, N.J., Ritschel, T.: Escaping plato's cave using adversarial training: 3D shape from unstructured 2D image collections. In: Proc. ICCV (2019) 5
27. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017) 7
28. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models (2020) 11, 2
29. Ho, J., Salimans, T.: Classifier-free diffusion guidance. arXiv preprint arXiv:2207.12598 (2022) 12
30. Hong, Y., Zhang, K., Gu, J., Bi, S., Zhou, Y., Liu, D., Liu, F., Sunkavalli, K., Bui, T., Tan, H.: Lrm: Large reconstruction model for single image to 3d. arXiv preprint arXiv:2311.04400 (2023) 2, 3, 13
31. Kar, A., Häne, C., Malik, J.: Learning a multi-view stereo machine. Advances in neural information processing systems **30** (2017) 2, 4, 9
32. Kar, A., Tulsiani, S., Carreira, J., Malik, J.: Category-specific object reconstruction from a single image. In: Proc. CVPR (2015) 5
33. Karnewar, A., Mitra, N.J., Vedaldi, A., Novotny, D.: Holofusion: Towards photo-realistic 3d generative modeling. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 22976–22985 (2023) 2, 4, 12
34. Karnewar, A., Ritschel, T., Wang, O., Mitra, N.: Relu fields: The little non-linearity that could. In: ACM SIGGRAPH 2022 Conference Proceedings. pp. 1–9 (2022) 2
35. Karnewar, A., Ritschel, T., Wang, O., Mitra, N.J.: ReLU fields: The little non-linearity that could. arXiv.cs **abs/2205.10824** (2022) 4, 6, 9
36. Karnewar, A., Vedaldi, A., Novotny, D., Mitra, N.: Holodiffusion: Training a 3D diffusion model using 2D images. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (2023) 2, 4, 5, 9, 10, 12
37. Karnewar, A., Wang, O., Ritschel, T., Mitra, N.: 3ingan: Learning a 3d generative model from images of a self-similar scene. arXiv preprint arXiv:2211.14902 (2022) 5
38. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (July 2023), https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/ 2

39. Li, W., Chen, X., Wang, J., Chen, B.: Patch-based 3d natural scene generation from a single example. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 16762–16772 (2023) 5
40. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014) 7
41. Liu, L., Gu, J., Zaw Lin, K., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. Advances in Neural Information Processing Systems **33**, 15651–15663 (2020) 4
42. Liu, M., Xu, C., Jin, H., Chen, L., Varma T, M., Xu, Z., Su, H.: 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) 2, 9
43. Liu, Y., Peng, S., Liu, L., Wang, Q., Wang, P., Theobalt, C., Zhou, X., Wang, W.: Neural rays for occlusion-aware image-based rendering. arXiv.cs **abs/2107.13421** (2022) 2
44. Lombardi, S., Simon, T., Saragih, J., Schwartz, G., Lehrmann, A., Sheikh, Y.: Neural volumes: Learning dynamic renderable volumes from images. arXiv preprint arXiv:1906.07751 (2019) 2
45. Long, X., Lin, C., Wang, P., Komura, T., Wang, W.: Sparseneus: Fast generalizable neural surface reconstruction from sparse views. In: European Conference on Computer Vision. pp. 210–227. Springer (2022) 2
46. Luo, C.: Understanding diffusion models: A unified perspective. arXiv preprint arXiv:2208.11970 (2022) 3
47. Martel, J.N., Lindell, D.B., Lin, C.Z., Chan, E.R., Monteiro, M., Wetzstein, G.: Acorn: Adaptive coordinate networks for neural scene representation. arXiv preprint arXiv:2105.02788 (2021) 4
48. Max, N.: Optical models for direct volume rendering. IEEE Transactions on Visualization and Computer Graphics **1**(2), 99–108 (1995) 6
49. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: Proc. ECCV (2020) 2, 4, 5, 6, 9
50. Müller, N., Siddiqui, Y., Porzi, L., Bulò, S.R., Kontschieder, P., Nießner, M.: Diffrf: Rendering-guided 3d radiance field diffusion. arXiv preprint arXiv:2212.01206 (2022) 5, 12
51. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. In: Proc. SIGGRAPH (2022) 2, 6, 14
52. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. arXiv preprint arXiv:2201.05989 (2022) 4
53. Nguyen, P., Karnewar, A., Huynh, L., Rahtu, E., Matas, J., Heikkila, J.: Rgbd-net: Predicting color and depth images for novel views synthesis. In: 2021 International Conference on 3D Vision (3DV). pp. 1095–1105. IEEE (2021) 2, 3, 4, 9
54. Nguyen-Phuoc, T., Li, C., Theis, L., Richardt, C., Yang, Y.: HoloGAN: Unsupervised learning of 3D representations from natural images. arXiv.cs **abs/1904.01326** (2019) 5
55. Niemeyer, M., Mescheder, L.M., Oechsle, M., Geiger, A.: Occupancy flow: 4d reconstruction by learning particle dynamics. In: Proc. ICCV (2019) 4
56. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al.: Dinov2: Learning robust visual features without supervision. arXiv preprint arXiv:2304.07193 (2023) 4, 9

57. Peebles, W., Xie, S.: Scalable diffusion models with transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 4195–4205 (2023) 12, 5
58. Rakhimov, R., Ardelean, A.T., Lempitsky, V., Burnaev, E.: Npbg++: Accelerating neural point-based graphics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 15969–15979 (2022) 3
59. Reizenstein, J., Shapovalov, R., Henzler, P., Sbordone, L., Labatut, P., Novotny, D.: Common Objects in 3D: Large-scale learning and evaluation of real-life 3D category reconstruction. In: Proc. CVPR (2021) 3
60. Resindra Widya, A., Torii, A., Okutomi, M.: Structure-from-motion using dense cnn features with keypoint relocalization. arXiv e-prints pp. arXiv–1805 (2018) 2
61. Schwarz, K., Liao, Y., Niemeyer, M., Geiger, A.: Graf: Generative radiance fields for 3d-aware image synthesis. Advances in Neural Information Processing Systems **33**, 20154–20166 (2020) 5
62. Schwarz, K., Sauer, A., Niemeyer, M., Liao, Y., Geiger, A.: Voxgraf: Fast 3d-aware image synthesis with sparse voxel grids. ARXIV (2022) 5
63. Shen, T., Gao, J., Yin, K., Liu, M.Y., Fidler, S.: Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis. Advances in Neural Information Processing Systems **34**, 6087–6101 (2021) 5
64. Shi, Z., Peng, S., Xu, Y., Geiger, A., Liao, Y., Shen, Y.: Deep generative models on 3d representations: A survey. arXiv preprint arXiv:2210.15663 (2022) 2, 4
65. Sitzmann, V., Martel, J.N.P., Bergman, A.W., Lindell, D.B., Wetzstein, G.: Implicit neural representations with periodic activation functions. In: Proc. NeurIPS (2020) 4
66. Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene representation networks: Continuous 3D-structure-aware neural scene representations. Proc. NeurIPS (2019) 2, 4
67. Son, M., Park, J.J., Guibas, L., Wetzstein, G.: Singraf: Learning a 3d generative radiance field for a single scene. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8507–8517 (2023) 5
68. Sun, C., Sun, M., Chen, H.T.: Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5459–5469 (2022) 2
69. Sun, C., Sun, M., Chen, H.: Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In: Proc. CVPR (2022) 4, 6, 9
70. Takikawa, T., Evans, A., Tremblay, J., Müller, T., McGuire, M., Jacobson, A., Fidler, S.: Variable bitrate neural fields. In: ACM SIGGRAPH 2022 Conference Proceedings. pp. 1–9 (2022) 2, 4
71. Tewari, A., Fried, O., Thies, J., Sitzmann, V., Lombardi, S., Sunkavalli, K., Martin-Brualla, R., Simon, T., Saragih, J., Nießner, M., et al.: State of the art on neural rendering. In: Computer Graphics Forum. vol. 39, pp. 701–727. Wiley Online Library (2020) 2
72. Tewari, A., Thies, J., Mildenhall, B., Srinivasan, P., Tretschk, E., Yifan, W., Lassner, C., Sitzmann, V., Martin-Brualla, R., Lombardi, S., et al.: Advances in neural rendering. In: Computer Graphics Forum. vol. 41, pp. 703–735. Wiley Online Library (2022) 2
73. Tewari, A., Yin, T., Cazenavette, G., Rezchikov, S., Tenenbaum, J.B., Durand, F., Freeman, W.T., Sitzmann, V.: Diffusion with forward models: Solving stochastic inverse problems without direct supervision. arXiv preprint arXiv:2306.11719 (2023) 2, 5, 10, 12, 1, 3

74. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017) 14
75. Wang, P., Liu, Y., Chen, Z., Liu, L., Liu, Z., Komura, T., Theobalt, C., Wang, W.: F2-nerf: Fast neural radiance field training with free camera trajectories. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4150–4159 (2023) 4
76. Wang, Q., Wang, Z., Genova, K., Srinivasan, P.P., Zhou, H., Barron, J.T., Martin-Brualla, R., Snavely, N., Funkhouser, T.A.: Ibrnet: Learning multi-view image-based rendering. In: Proc. CVPR (2021) 2
77. Wang, T., Zhang, B., Zhang, T., Gu, S., Bao, J., Baltrusaitis, T., Shen, J., Chen, D., Wen, F., Chen, Q., et al.: Rodin: A generative model for sculpting 3d digital avatars using diffusion. arXiv preprint arXiv:2212.06135 (2022) 6
78. Wang, Z., Simoncelli, E.P., Bovik, A.C.: Multiscale structural similarity for image quality assessment. In: The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003. vol. 2, pp. 1398–1402. Ieee (2003) 7
79. Wei, X., Zhang, Y., Li, Z., Fu, Y., Xue, X.: Deepsfm: Structure from motion via deep bundle adjustment. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16. pp. 230–247. Springer (2020) 2
80. Wei, X., Zhang, Y., Ren, X., Li, Z., Fu, Y., Xue, X.: Deepsfm: Robust deep iterative refinement for structure from motion. IEEE Transactions on Pattern Analysis and Machine Intelligence (2023) 2
81. Wiles, O., Gkioxari, G., Szeliski, R., Johnson, J.: Synsin: End-to-end view synthesis from a single image. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7467–7477 (2020) 3
82. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. Advances in neural information processing systems **29** (2016) 5
83. Wu, R., Liu, R., Vondrick, C., Zheng, C.: Sin3dm: Learning a diffusion model from a single 3d textured shape. arXiv preprint arXiv:2305.15399 (2023) 5
84. Wu, R., Zheng, C.: Learning to generate 3d shapes from a single example. arXiv preprint arXiv:2208.02946 (2022) 5
85. Wu, T., Zhang, J., Fu, X., Wang, Y., Ren, J., Pan, L., Wu, W., Yang, L., Wang, J., Qian, C., et al.: Omniobject3d github code and dataset (2023), https://github.com/omniobject3d/OmniObject3D 7
86. Wu, T., Zhang, J., Fu, X., Wang, Y., Ren, J., Pan, L., Wu, W., Yang, L., Wang, J., Qian, C., et al.: Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 803–814 (2023) 5, 7
87. Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., Sridhar, S.: Neural fields in visual computing and beyond. In: Computer Graphics Forum. vol. 41, pp. 641–676. Wiley Online Library (2022) 2, 4
88. Yang, J., Luo, K.Z., Li, J., Weinberger, K.Q., Tian, Y., Wang, Y.: Denoising vision transformers. arXiv preprint arXiv:2401.02957 (2024) 14
89. Yang, L., Kang, B., Huang, Z., Xu, X., Feng, J., Zhao, H.: Depth anything: Unleashing the power of large-scale unlabeled data. arXiv preprint arXiv:2401.10891 (2024) 2
90. Yu, A., Fridovich-Keil, S., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. arXiv preprint arXiv:2112.05131 (2021) 2, 4, 6, 9

91. Yu, A., Li, R., Tancik, M., Li, H., Ng, R., Kanazawa, A.: Plenoctrees for real-time rendering of neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5752–5761 (2021) 4
92. Yu, A., Ye, V., Tancik, M., Kanazawa, A.: PixelNeRF: Neural radiance fields from one or few images. In: Proc. CVPR (2021) 2
93. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018) 7
94. Zhang, Y., Wu, S., Snavely, N., Wu, J.: Seeing a rose in five thousand ways. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 962–971 (2023) 5
95. Zhou, T., Tucker, R., Flynn, J., Fyffe, G., Snavely, N.: Stereo magnification: Learning view synthesis using multiplane images. arXiv preprint arXiv:1805.09817 (2018) 2, 3, 4, 9

Supplementary Material for GOEmbed: Gradient Origin Embeddings for Representation Agnostic 3D Feature Learning

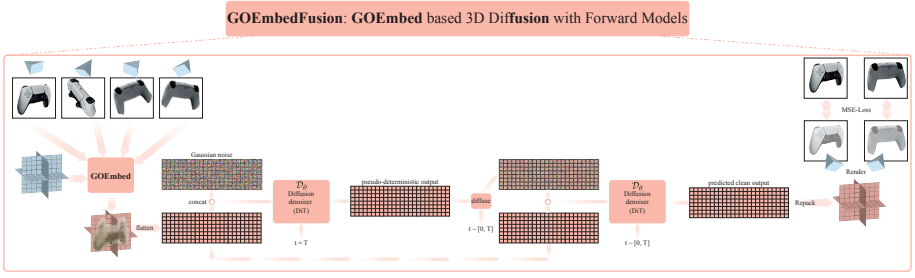

Fig. 1: Illustration diagram for the GOEmbedFusion method. Best viewed zoomed in.

1 Preliminaries

We start with a summary of the required preliminaries for **GONs**: Gradient Origin Networks in sec. 1.1 and **Diffusion with Forward** models in sec. 1.2. Please refer to the papers of Bond-Taylor and Willcocks [8], and Tewari et al. [73] for more details about these methods.

1.1 Preliminaries: GONs (Gradient Origin Networks)

Given a dataset of observed samples $x \sim p_{\text{data}}$, where $x \in \mathbb{R}^m$, a Gradient Origin Network [8] (GON) auto-encodes the input x into reconstructed $\hat{x}$ via a much smaller latent space $z \in \mathbb{R}^k$, where $k \ll m$, without requiring an explicit encoder network. The encoder mapping of the input x to latent z, $F_{\text{enc}} : \mathbb{R}^m \rightarrow \mathbb{R}^k$, is defined as the gradient of the log-likelihood of x under the decoder network $F_{\text{dec}} : \mathbb{R}^k \rightarrow \mathbb{R}^m$ wrt. a known fixed origin latent z_0. In practice, the z_0 is always set to zeros, i.e., $z_0 \in \mathbb{R}^k_{[0]}$ and the mean squared error between $F_{\text{dec}}(z_0)$ and x is used as an estimate of the log-likelihood as follows:

$$\begin{aligned} z &= F_{\text{enc}}(x) \\ &= -\nabla_{z_0} \|x - F_{\text{dec}}(z_0)\|_2^2. \end{aligned} \tag{1}$$

The decoding of z can now simply be obtained as a forward pass of the decoder network F_{dec}:

$$\begin{aligned} \hat{x} &= F_{\text{dec}}(z) \\ &= F_{\text{dec}}(-\nabla_{z_0} \|x - F_{\text{dec}}(z_0)\|_2^2). \end{aligned} \tag{2}$$

Using this encoder-less auto-encoding mechanism, the decoder network can be trained using the standard log-likelihood maximization objective (again estimated via mean squared error):

$$\begin{aligned} L^{\text{MSE}}(x, \hat{x}) &= \|x - \hat{x}\|_2^2 \\ &= \|x - F_{\text{dec}}(-\nabla_{z_0}\|x - F_{\text{dec}}(z_0)\|_2^2)\|_2^2. \end{aligned} \tag{3}$$

In the GON paper [8], Bond-Taylor and Willcocks also proposed a variational version of the GON autoencoder to enable generative sampling of the latent space, a coordinate-based implicit version of the GON as a secondary application, and a multi-step generalisation of the Gradient Origin Networks mechanism. But we only describe the auto-encoder version here, because it is the most relevant one to our proposed GOEmbed encodings (sec 3 of main paper).

1.2 Preliminaries: Diffusion with Forward models

Diffusion models define the generative process through two markov chains, namely forward (diffusion) and reverse (denoising), over the observed data distribution p_{data} [1]. The most popular variant DDPM [28] defines the forward distribution $q(x_{0:T}) = q(x_0, x_1, ..., x_t, ..., x_T)$ as the markov chain of T Gaussian transitions:

$$q(x_{0:T}) = q(x_0)\prod_{1}^{T} q(x_t|x_{t-1})$$

where,

$$q(x_t|x_{t-1}) = \mathcal{N}(\sqrt{\alpha_t}x_{t-1}, (1-\alpha_t)I), \tag{4}$$

where the sequence $(\alpha_{1:T})$ instantiates a monotonically decreasing noise schedule such that the marginal over the last variable approximately equals a standard Gaussian distribution, i.e. $q(x_T|x_{T-1}) \approx q(x_T) \approx \mathcal{N}(0, I)$. Similar to the forward process, the reverse process is also defined as another markov chain of T Gaussian transitions:

$$p(x_{T:0}) = p(x_T)\prod_{1}^{T} p_\theta(x_{t-1}|x_t)$$

where,

$$\begin{aligned} p(x_0) &= q(x_0) = p_{\text{data}}(x) \\ p(x_T) &= q(x_T) = \mathcal{N}(0, I) \\ p_\theta(x_{t-1}|x_t) &= \mathcal{N}(\sqrt{\alpha_{t-1}}\mathcal{D}_\theta(x_t, t), (1-\alpha_{t-1})I), \end{aligned} \tag{5}$$

where the mean of the Gaussian is given by a learned timestep-conditioned denoiser network $\mathcal{D}_\theta : \mathcal{R}^m \to \mathcal{R}^m$. The network $\mathcal{D}_\theta$ can be trained by minimizing

[1] please note that the forward diffusion process here is not to be confused with the forward setting of **Diffusion with Forward** models.

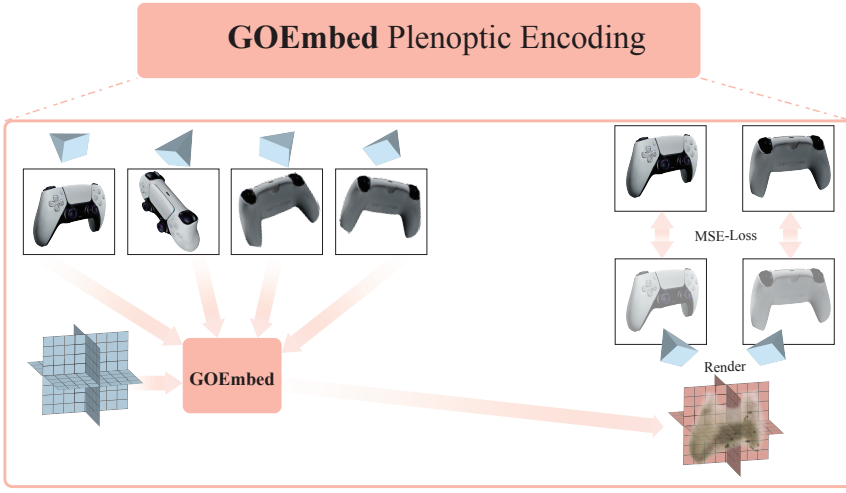

Fig. 2: Illustration diagram for the GOEmbed Plenoptic Encoding experimental setup.

the expectation of the KL divergence between [46] the forward and the reverse transition distributions of a given particular latent x_t over the timesteps t as

$$\mathcal{L}_{\text{diffusion}} = \mathbb{E}_t[D_{KL}(q(x_t|x_{t-1})||p_\theta(x_t|x_{t+1}))], \tag{6}$$

and synthetic samples can be drawn from the trained model by iterating T-times over eq. 5 starting from samples x_T drawn from standard Gaussian distribution $\mathcal{N}(0, I)$. In practice, assuming that the network $D_\theta(x_t, t)$ outputs an estimate of the clean data sample $\hat{x}_0 \approx x_0$ instead of the mean of the Gaussian transition over the previous latent x_{t-1}, the KL divergence objective simplifies to the mean squared error between $\hat{x}_0$ and x_0 [46]

$$\mathcal{L}_{\text{simple}}(x_0, \hat{x}_0) = \mathbb{E}_t[||x_0 - \mathcal{D}_\theta(x_t, t)||_2^2]. \tag{7}$$

DFM models [73] consider the class of *stochastic-inverse* problems, like inverse 3D graphics, which pose a unique challenge where we do not have direct access to the samples $x \sim p_{\text{data}}$, but only have access to partial observations of x obtained through a differentiable forward function $o = \texttt{forward}(x, \phi)$, where ϕ are the parameters for obtaining the observation. The mathematical framework of DFM

learns the conditional distribution $p(x|o, \phi)$ over the unobserved data x given the partial observations (o, ϕ), by modeling the *pushforward* distribution $p_\theta(o^{\text{trgt}}|o^{\text{ctxt}}, \phi^{\text{ct}}$ The observations $(o^{\text{ctxt}}, \phi^{\text{ctxt}})$ form the context while $(o^{\text{trgt}}, \phi^{\text{trgt}})$ are the target

observations.

$$p_\theta(o^{\text{trgt}}_{0:T}|o^{\text{ctxt}}, \phi^{\text{ctxt}}, \phi^{\text{trgt}}) = p(o^{\text{trgt}}_T)\prod_1^T p_\theta(o^{\text{trgt}}_{t-1}|o^{\text{trgt}}_t, o^{\text{ctxt}}, \phi^{\text{ctxt}}, \phi^{\text{trgt}}) \tag{8}$$

In order to implicitly model the conditional distribution over the data x from the pushforward, each of the learned reverse transition $p_\theta(o^{\text{trgt}}_{t-1}|o^{\text{trgt}}_t, o^{\text{ctxt}}, \phi^{\text{ctxt}}, \phi^{\text{trgt}})$ is defined using the following three steps:

$$x_t = \texttt{denoise}(o^{\text{ctxt}}, o^{\text{trgt}}_t, t, \phi^{\text{ctxt}}, \phi^{\text{trgt}}) \tag{9}$$

$$\hat{o}^{\text{trgt}}_t = \texttt{forward}(x_t, \phi^{\text{trgt}}) \tag{10}$$

$$o^{\text{trgt}}_{t-1} \sim \mathcal{N}(\sqrt{\alpha_{t-1}}\hat{o}^{\text{trgt}}_t, (1-\alpha_{t-1})I). \tag{11}$$

Equations 9 and 10 define the same functionality as that of the $\mathcal{D}_\theta$ denoiser network, but also integrate the differentiable `forward` function in the process. The noisy versions of the observed target views o^{trgt}_t can be obtained by the straightforward diffusion process as defined in eq. 4 without any changes. The Forward-Diffusion model can be trained using a conditional version of the KL divergence of eq. 6 as follows:

$$\mathcal{L}_{\text{forward-diffusion}} = \mathbb{E}_t[D_{KL}(q(o^{\text{trgt}}_t|o^{\text{trgt}}_{t-1})||p_\theta(o^{\text{trgt}}_t|o^{\text{trgt}}_{t+1}, o^{\text{ctxt}}, \phi^{\text{ctxt}}), \phi^{\text{trgt}})].$$

And lastly, synthetic samples of the unobserved underlying variable x can be generated in a auto-regressive manner. First we draw a sample o^{trgt} starting from a given set of $(o^{\text{ctxt}}, \phi^{\text{ctxt}})$ context observations by iterating over equations 9, 10, and 11 T times. A partial estimate of the unobserved underlying variable x^0_0 is obtained from the last denoising step. The complete sample x can be generated by merging n different partial estimates $x^{0:n-1}_0$ by repeating the former process n-times and accumulating the drawn o^{trgt}s together as a new context each time.

2 GOEmbedFusion implementation details

Fig. 1 illustrates the pipeline of the proposed GOEmbedFusion method. Although eqs. 3-6 of the main paper describe the model mathematically, more algorithmic details are provided as follows: We do all the experiments with the Triplane 3D representation. For the GOEmbedFusion model, we use 4 source views and 2 target views for training. We train on the OmniObject3D dataset with a batch size of 16 for $1M$ iterations. Of all the data and metadata provided in the dataset, we make use of the RGB, depth and normal-map renders of the textured-meshes available as part of the scans and the 3D camera parameters of the rendered views. All the rendered maps in the dataset are natively rendered at the resolution of `[800 x 800]` in Blender [9]. Each of the 3D scan contains

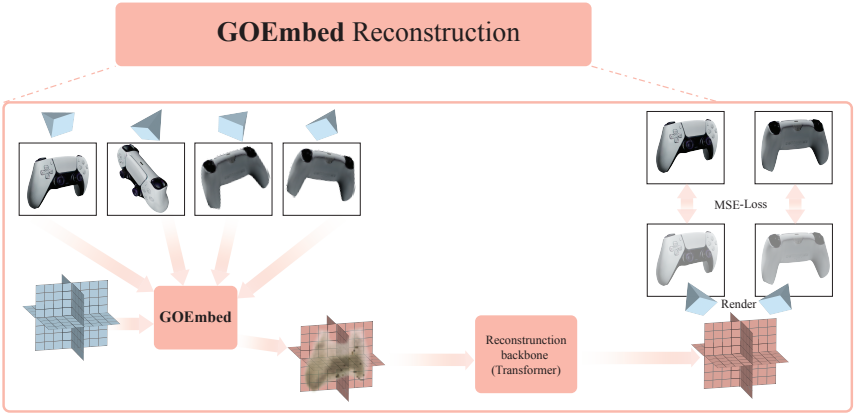

Fig. 3: Illustration diagram for the GOEmbed 3D reconstruction experimental setup.

∼100 renders from random viewpoints on the upper hemisphere of the 3D centered objects. We use *only* the RGB maps for training the model, while use the depth-maps and normal-maps for computing metrics.

For the diffusion details, we base our code on the guided-diffusion code [1], and use the default values of 0.0001 and 0.02 for the `beta-start` and `beta-end` respectively. For the remaining, we use $T = 1000$ timesteps, `cosine` noise schedule and the `x-start` as the model output. Lastly, we use a single learning rate `5e-5` for the entire span of the training.

3 Discussion on 3D reconstruction

Fig. 2 details the setup of the Plenoptic Encoding experiments, while Fig 3 depicts the setup for the sparse-view 3D reconstruction. As we can see, the only difference between these two setups is The presence of a deep learning reconstruction backbone between the two. For this baseline, we train a regression based network which reconstructs the 3D scene using only the GOEmbed encoded 3D representation as input. Note that this baseline doesn't apply noise anywhere, and hence gives purely deterministic output. As can be seen from the scores of table 3 of the main paper, the reconstruction only (regression) model achieves 27.650 dB, while the GOEmbedFusion gets as close as 26.447 dB PSNR by itself. Lastly, the Transformer architecture used by us is based on the `DiT-XL/2` model of Peebles et al. [57].