

Geometry-biased Transformers for Novel View Synthesis

Naveen Venkat^{*1} Mayank Agarwal^{*1} Maneesh Singh Shubham Tulsiani¹

¹Carnegie Mellon University

{nvenkat, mayankag, shubhtuls}@cmu.edu, dr.maneesh.singh@ieee.org

<https://mayankgrwl97.github.io/gbt>

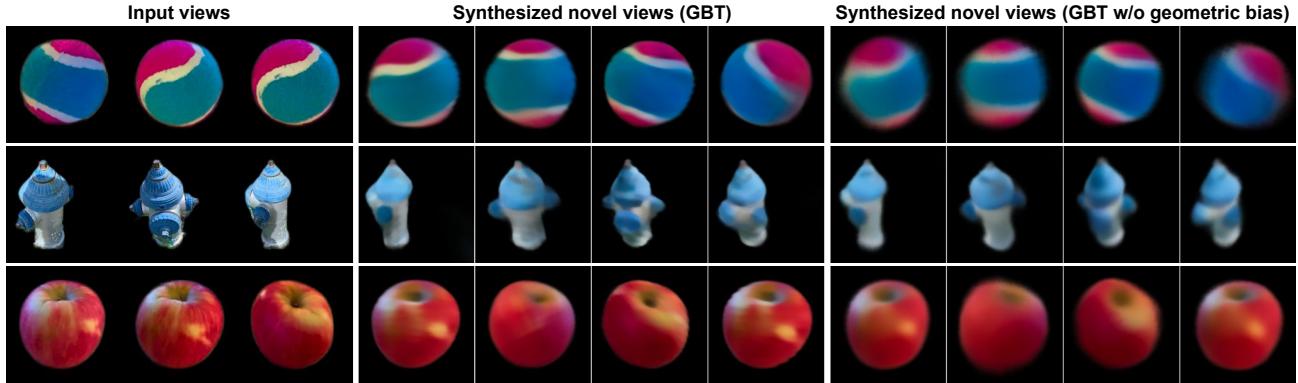


Figure 1. Given a small set of context images with known camera viewpoints (left), our Geometry-biased transformer (GBT) synthesizes novel views from arbitrary query viewpoints (middle). The use of global context ensures meaningful prediction despite large viewpoint variation, while the geometric bias allows more accurate inference compared to a baseline without such bias (right).

Abstract

We tackle the task of synthesizing novel views of an object given a few input images and associated camera viewpoints. Our work is inspired by recent ‘geometry-free’ approaches where multi-view images are encoded as a (global) set-latent representation, which is then used to predict the color for arbitrary query rays. While this representation yields (coarsely) accurate images corresponding to novel viewpoints, the lack of geometric reasoning limits the quality of these outputs. To overcome this limitation, we propose ‘Geometry-biased Transformers’ (GBTs) that incorporate geometric inductive biases in the set-latent representation-based inference to encourage multi-view geometric consistency. We induce the geometric bias by augmenting the dot-product attention mechanism to also incorporate 3D distances between rays associated with tokens as a learnable bias. We find that this, along with camera-aware embeddings as input, allows our models to generate significantly more accurate outputs. We validate our approach on the real-world CO3D dataset, where we train our system over 10 categories and evaluate its view-synthesis ability for novel objects as well as unseen categories. We empirically validate the benefits of the proposed geometric biases and show that our approach significantly improves over prior works.

1. Introduction

Given just a few images depicting an object, we humans can easily imagine its appearance from novel viewpoints. For instance, consider the first image of the hydrant shown in Figure 1 and imagine rotating it slightly anti-clockwise – we intuitively understand that this would move the small outlet towards the front and right. We can also imagine rotating the hydrant further and know that the (currently occluded) central outlet will eventually become visible on the left. These examples serve to highlight that this task of novel-view synthesis requires both reasoning about geometric transformations *e.g.* motion of the visible surfaces, as well as an understanding of the global structure *e.g.* occlusions and symmetries to allow for realistic extrapolations. In this work, we develop an approach that incorporates both these to synthesize accurate novel views given only a sparse set of images of a previously unseen object.

Recent advances in Neural Radiance Fields (NeRFs) [13] have led to numerous approaches that use these representations (and their variants) for obtaining remarkably detailed novel-view renderings. However, such methods typically optimize instance-specific representations using densely sampled multi-view observations, and cannot be directly leveraged for 3D inference from sparse input views.

^{*} indicates equal contribution

To enable generalizable inference from a few views, recent methods seek to instead predict radiance fields using the image projections of a query 3D point as conditioning. While using such geometric reprojection constraints allows accurate predictions in the close vicinity of observed views, this purely local conditioning mechanism fails to capture any global context *e.g.* symmetries or correlated patterns. As a result, these approaches struggle to render views containing unobserved aspects or large viewpoint variations.

Our work is motivated by an alternate approach to generalizable view synthesis, where a geometry-free (global) scene representation is used to predict images from query viewpoints. Specifically, these methods form a set-latent representation from multiple input views and directly infer the color for a pixel for a query view (or equivalently a query ray) using attention-based mechanisms in the scene encoding and ray decoding process. Not only is this direct view synthesis more computationally efficient than volume rendering, but the set-latent representation also allows capturing global context as each ray can attend to all aspects of other views instead of just the projections of points along it. However, this ‘geometry-free’ design comes at the cost of precision – these methods cannot easily capture the details in input views, and while they can robustly capture the coarse structure, do not output high-quality renderings.

In this work, we develop mechanisms to inject geometric biases in these set-latent representation-based approaches. Specifically, we propose Geometry-biased Transformers (GBTs) which consist of a ray-distance-based bias in the attention mechanism in Transformer layers. We show that these help guide the scene encoding and ray decoding stages to pay attention to relevant context, thereby enabling more accurate view synthesis. We benchmark our approach using the Co3D dataset [18] that comprises of challenging real-world captures across diverse categories. We show that our approach outperforms both, projection-based radiance field prediction and set-latent representation-based view synthesis approaches, and also demonstrate our method’s ability to generalize to unseen object categories.

2. Related Work

Instance-specific 3D Representations. Driven by the recent emergence of neural fields [13], a growing number of methods seek to accurately capture the details of a specific object or scene given multiple images. Leveraging either volumetric [1, 2, 5, 9, 13, 14, 16], implicit [17, 27, 31], mesh-based [8, 33], or hybrid [3, 7] representations, these methods learn instance-specific representations capable of synthesizing novel views. However, as these methods do not learn generic data-driven priors, they typically require densely sampled views to be able to infer geometrically consistent underlying representations and are incapable of predicting beyond what they directly observe.

Projection-guided Generalizable View Synthesis. Closer to our goal, several methods have aimed to learn models capable of view-synthesis across instances. While initial attempts [22] used global-variable-conditioned neural fields, subsequent approaches [4, 24, 28, 32] obtained significant improvements by instead using features extracted via projection onto the context views. Reznestein *et al.* [18] further demonstrated the benefits of learning the aggregation mechanisms across the features along a query ray, but the projection-guided features remained the fundamental building blocks. While these projection-based methods are effective at generating novel views by transforming the visible structures, they struggle to deal with large viewpoint changes (as the underlying geometry maybe uncertain), and are fundamentally unable to generate plausible visual information not directly observed in the context views. We argue that this is because these methods lack the mechanisms to learn and utilize contexts globally when generating query views.

Geometry-free View Synthesis. To allow using global context for view synthesis, an alternate class of methods uses ‘geometry-free’ encodings to infer novel views. The initial learning-based methods [23, 30, 34] typically focused on novel-view prediction given a single image via global conditioning. Subsequent approaches [11, 15, 19] improved performance using different architectures *e.g.* Transformers [26], while also allowing for probabilistic view synthesis using VQ-VAEs [25] and VQ-GANs [6]. While this leads to detailed and realistic outputs, the renderings are not 3D-consistent due to stochastic sampling.

Our work is inspired by the recently proposed Scene Representation Transformer (SRT) [20], which uses a set-latent representation that encodes both patch-level and global scene context. This design engenders a fast, deterministic rendering pipeline that, unlike projection-based methods, furnishes plausible hallucinations in the invisible regions. However, these benefits come at the cost of detail – unlike the projection-based methods, this geometry-free approach is unable to capture precise details in the visible aspects. Motivated by this need to improve the detail, we propose mechanisms to inject geometric biases in this framework, and find that this significantly improves the performance while preserving global reasoning and efficiency.

3. Approach

We aim to render novel viewpoints of previously unseen objects from a few posed images. To achieve this goal, we design a rendering pipeline that reasons along the following two aspects: (i) **appearance** - *what is the likely appearance of the object from the queried viewpoint*, and, (ii) **geometry** - *what geometrically-informed context can be derived from the configuration of the given input and query cameras?*

Prior methods address each question in isolation *e.g.* via

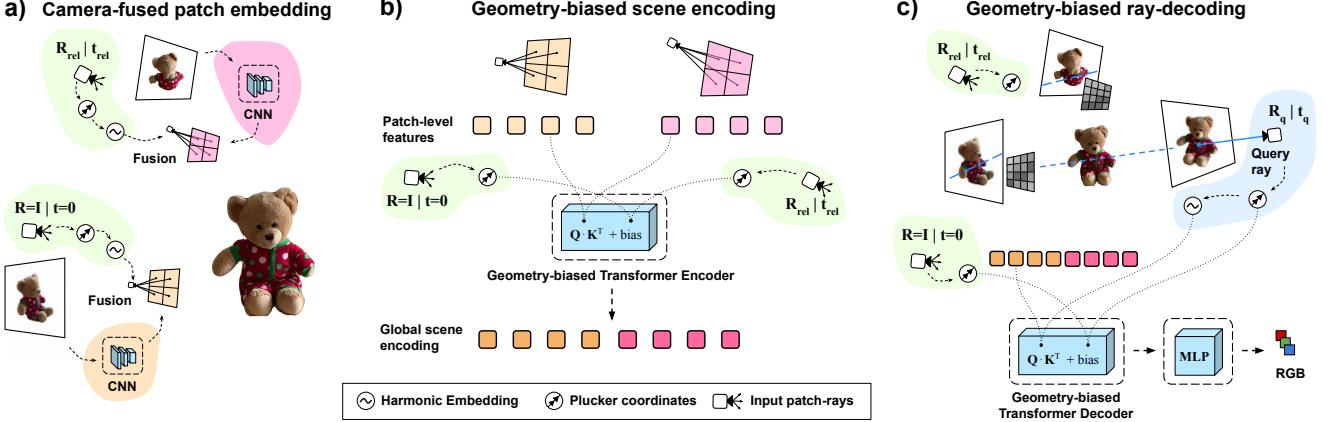


Figure 2. Learning novel view synthesis using Geometry-biased Transformers. Best viewed in color. **a) Camera-fused patch embedding.** Each input image \mathbf{I}_i is processed using a shared CNN backbone F_C and the feature maps are fused with the corresponding input patch-ray embeddings (obtained via \mathbf{p}_i). **b) Geometry-biased scene encoding.** Our proposed Geometry-biased Transformer encoder F_E converts the set of patch-level feature tokens into a scene encoding via self-attention biased with ray distances. **c) Geometry-biased ray-decoding.** To decode pixels for a novel viewpoint, we construct ray queries that are decoded by a geometry-biased transformer decoder F_D by attending into the scene encoding. Finally, an MLP predicts the pixel color using the decoded query token.

global latent representations [11, 20, 22, 29] that address (i) by learning object semantics, or, via reprojections [18, 32] that address (ii) by employing explicit geometric transformations. In contrast to prior works, our method jointly reasons along both these aspects. Concretely, we propose geometry-biased transformers that incorporate geometric inductive biases while learning set-latent representations that help capture global structures with superior quality.

Fig. 2 depicts the Geometry-biased Transformer (GBT) framework which has three components. First, a shared CNN backbone extracts patch-level features which are fused with the corresponding ray embeddings to derive local (pose-aware) features (Fig. 2a). Then, the flattened patch features and the associated rays are fed as input tokens to the GBT encoder that constructs a global set-latent representation via self-attention (Fig. 2b). The attention layers are biased to prioritize both the photometric and the geometric context. Finally, the GBT decoder converts target ray queries to pixel colors by attending to the set-latent representation (Fig. 2c). We now review the preliminary concepts before describing our approach in detail.

3.1. Preliminaries

3.1.1 Ray representations

The fundamental unit of geometric information in our approach is a ray which is used to compute the geometric similarity between two image regions. A naive choice for ray representation is $\mathbf{r} = (\mathbf{o}, \mathbf{d})$, where $\mathbf{o} \in \mathbb{R}^3$ is the origin of the ray, and $\mathbf{d} \in \mathbb{S}^2$ is the normalized ray direction.

In contrast, we use the 4 DoF Plucker coordinates [10, 21], $\mathbf{r} = (\mathbf{d}, \mathbf{m}) \in \mathbb{R}^6$, where $\mathbf{m} = \mathbf{o} \times \mathbf{d}$, that are invari-

ant to the choice of the origin along the ray. Intuitively, this allows us to associate a single color (pixel RGB) to the entire ray, agnostic to its origin. In practice, this simplification mitigates overfitting to the camera origin during training.

3.1.2 Scene Representation Transformers

The overall framework of our approach is inspired by SRT [20] that proposes a transformer encoder-decoder network for novel view synthesis. Given a collection of posed images $\{(\mathbf{I}_i, \mathbf{p}_i)\}_{i=1}^V$ where $\mathbf{I} \in \mathbb{R}^{H \times W \times 3}$ $\mathbf{p}_i \in \mathbb{R}^{3 \times 4}$, and a query ray \mathbf{r} , SRT computes the following:

$$\{\mathbf{z}_p\}_{p=1}^{V \times P} = F_E \circ F_C(\{\mathbf{I}_i, \mathbf{p}_i\}) \quad (1)$$

$$C(\mathbf{r}) = F_D(\mathbf{r} | \{\mathbf{z}_p\}) \quad (2)$$

Here, the shared CNN backbone (F_C) extracts P patch-level features from each posed input image. These are aggregated into a set of flat patch embeddings and fed as input tokens to the transformer encoder (F_E). The encoder transforms input tokens into a set-latent scene representation $\{\mathbf{z}_p\}$ via self-attention. To render a novel viewpoint, the decoder F_D queries for each ray \mathbf{r} pertaining to the target pixels and yields an RGB color by attending to the scene representation $\{\mathbf{z}_p\}$.

3.2. Geometry-biased Transformer (GBT) Layer

The core reasoning module in a transformer is a multi-head attention layer that aggregates information from the right context for each query. In our work, we propose to extend this module by incorporating geometric reasoning.

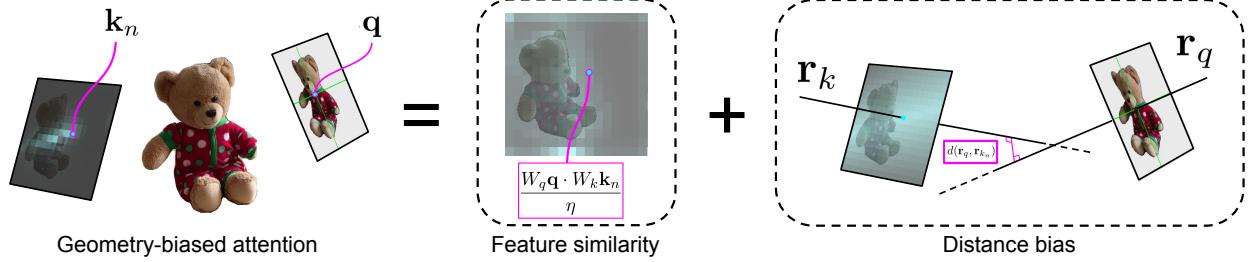


Figure 3. **An illustration of attention within GBT layer.** Given the query \mathbf{q} , key $\{\mathbf{k}_n\}$, value $\{\mathbf{v}_n\}$ tokens, a typical transformer layer computes:

$$\mathbf{q}' = T(\mathbf{q}, \{(\mathbf{k}_n, \mathbf{v}_n)\}) \quad (3)$$

which consists of a multi-head attention module, followed by normalization and linear projection. During the context aggregating step, each multi-head attention layer aggregates token values based on query-key similarity weights:

$$w_n = \text{softmax}_n \left(\frac{W_q \mathbf{q} \cdot W_k \mathbf{k}_n}{\eta} \right) \quad (4)$$

Incorporating ray distance as geometric bias. In our use case, each query and context token pertains to some ray. For instance, all tokens passed to the encoder are patch embeddings that have associated patch rays (Fig. 2b). Likewise, we query the decoder using target pixel rays (Fig. 2c).

In such a scenario, we propose to bias the transformer’s attention by encouraging similarity between rays that are closer to each other in 3D space. Specifically, the GBT layer couples the query and key tokens with the associated rays $(\mathbf{q}, \mathbf{r}_q), \{(\mathbf{k}_n, \mathbf{r}_{k_n})\}$ and performs the token transformation:

$$\mathbf{q}' = GBT((\mathbf{q}, \mathbf{r}_q), \{(\mathbf{k}_n, \mathbf{r}_{k_n}, \mathbf{v}_n)\}) \quad (5)$$

The attention layer is modified to account for the distance between $\mathbf{r}_q = (\mathbf{d}_q, \mathbf{m}_q)$ and $\mathbf{r}_{k_n} = (\mathbf{d}_{k_n}, \mathbf{m}_{k_n})$:

$$w_n = \text{softmax} \left(\frac{W_q \mathbf{q} \cdot W_k \mathbf{k}_n}{\eta} - \gamma^2 d(\mathbf{r}_q, \mathbf{r}_{k_n}) \right) \quad (6)$$

where,

$$d(\mathbf{r}_q, \mathbf{r}_{k_n}) = \begin{cases} \frac{|\mathbf{d}_q \cdot \mathbf{m}_{k_n} + \mathbf{d}_{k_n} \cdot \mathbf{m}_q|}{\|\mathbf{d}_q \times \mathbf{d}_{k_n}\|_2}, & \mathbf{d}_q \times \mathbf{d}_{k_n} \neq 0 \\ \frac{\|\mathbf{d}_q \times (\mathbf{m}_q - \mathbf{m}_{k_n}/s)\|}{\|\mathbf{d}_q\|_2^2}, & \mathbf{d}_{k_n} = s\mathbf{d}_q, s \neq 0 \end{cases} \quad (7)$$

and γ is a learnable parameter controlling the relative importance of geometric bias. This formulation explicitly accounts for both appearance (feature similarity between \mathbf{q}

and \mathbf{k}_n), and geometry (distance between \mathbf{r}_q and \mathbf{r}_{k_n}). This attention mechanism is illustrated in Fig. 3. In practice, the distance bias results in faster convergence to the right context during training. While one can fix γ to some constant hyperparameter, we found improved results by learning γ .

3.3. Learning Novel View Synthesis with GBTs

Given multiview images $\{\mathbf{I}_i \in \mathbb{R}^{H \times W \times 3}\}_{i=1}^V$ with paired camera poses $\{\mathbf{p}_i \in \mathbb{R}^{3 \times 4}\}_{i=1}^V$, we wish to render a target viewpoint described by the camera pose $\mathbf{p}_q \in \mathbb{R}^{3 \times 4}$. Our network, as illustrated in Fig. 2, first processes the posed multiview images using a CNN F_C to extract patch-level latent features. We then use GBT encoder F_E to extract a scene encoding, and GBT decoder F_D to yield pixel colors given target ray queries.

a) Camera-fused patch embedding (F_C). We process each context image \mathbf{I}_i through a ResNet18 backbone to obtain patch-level image feature grid. Subsequently, each patch feature is concatenated with the corresponding ray embedding (Fig. 2a) as follows:

$$[\mathbf{f}_c]_i^k = \mathbf{W} \left([F_C(\mathbf{I}_i)]^k \oplus h((\mathbf{d}_i^k, \mathbf{m}_i^k)) \right) \quad (8)$$

where $h(\cdot)$ denotes harmonic embedding [13], $(\mathbf{d}_i^k, \mathbf{m}_i^k)$ denotes the Plücker coordinates for k^{th} patch ray in the i^{th} input image, and \oplus denotes concatenation. We define each patch ray as the ray passing through the center of the receptive field of the corresponding cell in the feature grid. The concatenated features are projected using a linear layer \mathbf{W} .

While SRT fuses input images with per-pixel rays before the CNN, we fuse the CNN output feature grid with per-patch rays (observe different inputs to F_C in Eq. 1 and Eq. 8). This late fusion enables us to leverage transfer learning using pretrained image backbones. Furthermore, since the patch ray embeddings implicitly capture the positional information for each patch, we do not require 2D positional encoding or camera ID embedding after the CNN (unlike SRT), thus simplifying the architecture significantly.

Table 1. **Evaluation of novel view synthesis.** Given $V = 3$ input views, we evaluate the reconstruction quality (PSNR \uparrow and LPIPS \downarrow) of each method on the CO3Dv2 [18] dataset. GBT denotes our proposed approach, and GBT-nb is an ablation. See Sec. 4.2.

10 training cat.	Apple		Ball		Bench		Cake		Donut		Hydrant		Plant		Suitcase		Teddybear		Vase		Mean	
	PSNR	LPIPS																				
pixelNeRF [32]	20.87	0.29	20.17	0.30	18.69	0.34	19.20	0.34	20.79	0.29	20.43	0.26	20.68	0.30	22.19	0.32	19.80	0.34	20.82	0.28	20.37	0.31
NerFormer [18]	20.91	0.31	17.50	0.35	16.06	0.52	18.08	0.46	21.19	0.33	19.33	0.31	19.31	0.50	20.31	0.46	16.95	0.47	18.04	0.39	18.77	0.41
ViewFormer [11]	21.70	0.24	19.34	0.30	17.08	0.30	18.04	0.32	19.59	0.28	18.59	0.21	18.34	0.31	21.61	0.26	16.60	0.31	21.52	0.21	19.24	0.27
GBT-nb	22.83	0.28	20.59	0.32	19.22	0.34	20.56	0.34	21.87	0.31	21.32	0.24	21.52	0.30	23.30	0.29	19.82	0.34	22.65	0.27	21.37	0.30
GBT	25.08	0.23	22.96	0.26	19.93	0.31	21.51	0.30	23.05	0.27	22.76	0.22	21.88	0.27	24.15	0.27	20.89	0.30	23.36	0.25	22.56	0.27

Table 2. **Evaluation of variable context views setting.** We report PSNR (\uparrow) and LPIPS (\downarrow) averaged over 10 categories for each V .

10 training cat.	PSNR \uparrow			LPIPS \downarrow		
	$V = 2$	$V = 3$	$V = 6$	$V = 2$	$V = 3$	$V = 6$
pixelNeRF [32]	18.47	20.37	22.25	0.36	0.31	0.26
NerFormer [18]	17.88	18.77	20.01	0.43	0.41	0.38
ViewFormer [11]	18.62	19.24	20.12	0.28	0.27	0.26
GBT-nb	20.91	21.37	21.49	0.31	0.30	0.30
GBT	21.47	22.56	23.09	0.29	0.27	0.27

b) Geometry-biased scene encoding (F_E). Given local patch features, we employ GBT encoder layers to augment them with the global scene context through self-attention. Specifically, we compute $\mathbf{f}_e = F_E(\mathbf{f}_c, \{(\mathbf{d}_i^k, \mathbf{m}_i^k)\})$ where F_E contains a stack of GBT encoder layers as depicted in Fig. 2b. The query, key, and value tokens for the encoder layers are derived from the patch features $[\mathbf{f}_c]_i^k$ and their corresponding patch rays $(\mathbf{d}_i^k, \mathbf{m}_i^k)$. For each transformer encoder layer, we learn a separate γ parameter.

Finally, the encoder outputs a global scene encoding $\{[\mathbf{f}_e]_i^k\}$ that characterizes the appearance and the geometry of the object as observed from the multiple input views. Note, this extension of the set-latent representation [20] incorporates both appearance and geometric priors.

c) Geometry-biased ray decoding (F_D). To render a novel viewpoint given camera pose \mathbf{p}_q , we construct an $H \times W$ grid of query rays $\mathbf{r}_q = (\mathbf{d}_q, \mathbf{m}_q)$, with one ray per query pixel. We then employ a stack of GBT decoder layers F_D that decodes each query ray independently by aggregating meaningful context via cross-attention (Fig. 2c). Specifically, the query tokens for the multihead attention pertain to the query ray embeddings $h(\mathbf{r}_q)$, while the keys and values comprise of the global scene encoding tokens $\{[\mathbf{f}_e]_i^k\}$ along with the patch rays. The transformed query embeddings are processed by an MLP to predict the pixel color. Similar to F_E , we learn a separate parameter γ for each GBT decoder layer in F_D .

Architectural details. We use a ResNet18 (ImageNet initialized) up to the first 3 blocks as F_C . The images are re-

Table 3. **Evaluation of novel-view synthesis on unseen categories.** Given $V = 3$ input views, we evaluate the reconstruction quality (PSNR \uparrow and LPIPS \downarrow) on unseen categories.

5 heldout cat.	Backpack		Book		Chair		Mouse		Remote		Mean	
	PSNR	LPIPS										
pixelNeRF [32]	22.87	0.31	18.86	0.34	20.30	0.32	23.39	0.27	23.74	0.23	21.83	0.30
ViewFormer [11]	20.84	0.31	16.84	0.32	15.94	0.31	21.55	0.26	20.42	0.22	19.12	0.28
GBT-nb	23.55	0.33	19.38	0.35	20.50	0.32	23.72	0.27	24.00	0.22	22.23	0.30
GBT	24.08	0.30	20.36	0.32	21.46	0.28	24.91	0.23	24.63	0.21	23.09	0.27

sized to $H \times W = 256 \times 256$ and F_C outputs a 16×16 feature grid. We use 8 GBT encoder layers and 4 GBT decoder layers, wherein each transformer contains 12 heads for multi-head attention with *gelu* activation. For the harmonic embeddings h , we use 15 frequencies $\{2^{-6}\pi, \dots, 2^8\pi\}$. Since we do not have access to a consistent world coordinate frame across scenes, we choose an arbitrary input view as identity [20, 32]. All other cameras are represented relative to the identity view. See Appendix C for more details.

Training and Inference. During training, we encode $V = 3$ posed input views and query the decoder for $Q = 7168$ randomly sampled rays for a given target pose \mathbf{p}_q . The pixel color is supervised using an L2 reconstruction loss. The model is trained with Adam optimizer with 10^{-5} learning rate until loss convergence. At inference, we encode the context views once and decode a batch of $H \times W$ rays for each query view in a single forward pass. This results in a fast rendering time. See Appendix D for more details.

4. Experiments

4.1. Setup and Training Data

Dataset. We experiment on the Common Objects in 3D (CO3Dv2) dataset [18] that contains multi-view images along with camera pose annotations. This is a challenging dataset containing real-world object captures from 51 MS-COCO categories. Following [18], we train our network on 10 categories (see Table 1). Further, we evaluate our method on 5 additional heldout categories (see Table 3) to demonstrate generalization to unseen categories (see Appendix D for details on training and testing splits).

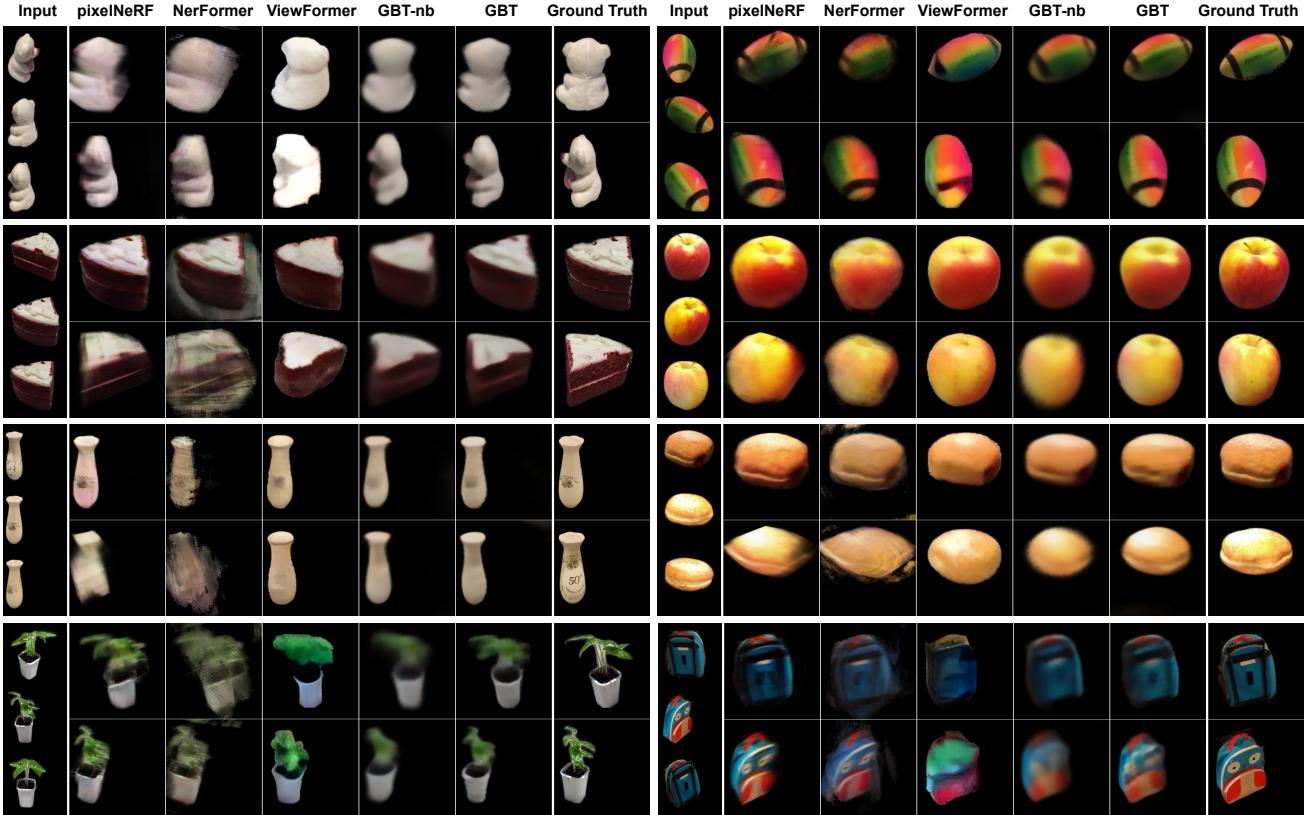


Figure 4. **Qualitative results on heldout objects from training categories.** For each object, we consider $V = 3$ input views and compare the reconstruction quality of each method on 2 other query views. Best viewed in color.

Baselines. We benchmark GBT against three state-of-the-art methods:

- *pixelNeRF* [32] which is a representative of projection-guided methods for generalizable view synthesis. Similar to our setting, we train a single category-agnostic pixelNeRF model on 10 categories from the CO3Dv2 dataset.
- *NerFormer* [18] which uses attention-based mechanisms to aggregate projected features along a query ray. We utilize (category-specific) models provided by the authors.¹
- *ViewFormer* [11] which uses a two-stage ‘geometry-free’ architecture to first encode the input images into a compact representation, and then uses a transformer model for view synthesis. For evaluation, we use the co3d-10cat model provided by the authors.

Additionally, we compare against another variant of our approach, where we replace the geometry-biased transformer layers with regular transformer layers (equivalently, set $\gamma = 0$ during training and inference). We refer to this as GBT-nb (no bias) in further discussion. GBT-nb is an extension of SRT [20], where we use Plücker coordinates

¹ While we evaluated per-category models, the NerFormer authors conveyed this performance is similar to a cross-category model.

Table 4. **Ablative analysis.** We train a separate category-specific model from scratch under each setting. The models are evaluated on the held out objects under consistent settings.

Method	Hydrant		Teddybear	
	PSNR (\uparrow)	LPIPS (\downarrow)	PSNR (\uparrow)	LPIPS (\downarrow)
SRT*	19.63	0.23	19.48	0.32
GBT-nb	21.30	0.20	19.32	0.31
GBT-fb	23.93	0.17	20.99	0.28
GBT	24.22	0.17	21.45	0.26

representation of rays and perform a late camera-fusion in the feature extractor.

Evaluation Metrics. To evaluate reconstruction quality, we measure the peak signal-to-noise ratio (PSNR) and perceptual similarity metric (LPIPS). For each category, we select 10 scenes from the dev set for evaluation. We randomly sample V context views and 32 query views for each scene and report the average metrics computed over these query views. We set appropriate seeds such that the context and query views are consistent across all methods.

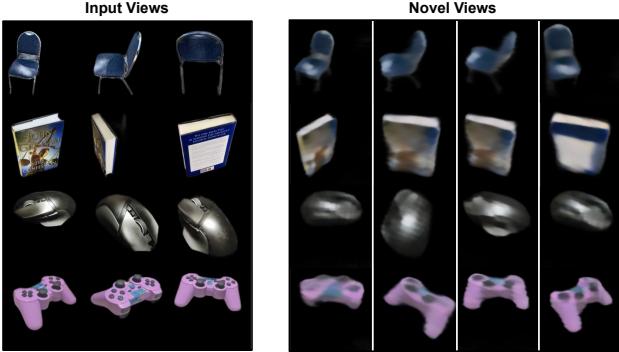


Figure 5. **Qualitative results on heldout categories.** On each row we visualize the rendered views obtained from GBT (right) given $V = 3$ input views (left). Note that the model has never seen these categories of objects during training.

4.2. Results

Novel view synthesis for unseen objects. Table 1 demonstrates the efficacy of our method in synthesizing novel views for previously unseen objects. GBT consistently outperforms other methods in all categories in terms of PSNR. With the exception of a few categories, we also achieve superior LPIPS compared to other baselines.

For categories such as bench, hydrant, etc. we attribute ViewFormer’s higher perceptual quality to their use of a 2D-only prediction model, which comes at the cost of multi-view consistent results. For instance, in Fig 4, ViewFormer’s prediction for the donut is plausibly similar to some donut, however, lacks consistency with the corresponding ground truth query view. Also, in cases where the query view is not visible in any of the input views (ball, top-right), pixelNeRF and NerFormer - which rely solely on projection-based features from input images - suffer from poor results, while our method is capable of hallucinating these unseen regions.

Table 2 analyses the performance of all methods with variable number of context views. While GBT is only trained with a fixed $V = 3$ input views, it is capable of generalizing across different input view settings. We observe a higher performance gain under fewer context views (2-3). However, as the number of input views increases, pixelNeRF becomes more competitive.

Generalization to unseen categories. To investigate whether our model learns generic 3D priors and can infer global context from given multi-view images, we test its ability to generalize to previously unseen categories. In Table 3 we benchmark our method by evaluating over 5 held out categories. We empirically find that GBT demonstrates better generalizability compared to baselines, and also observe this in the qualitative predictions in Figure 5.

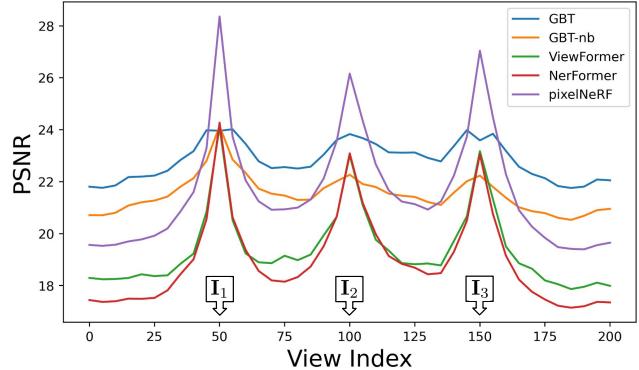


Figure 6. **Effect of viewpoint distance in prediction accuracy.** Given 200 frames, we set the 50th, 100th, 150th frame as the input views, and evaluate the performance of novel view synthesis over all other views. While the prior methods show accurate results close to the input views, our approach (GBT) consistently outperforms them in other views.

4.3. Analysis

Effect of Viewpoint Distance in Prediction Accuracy. In Fig 6, we analyze view synthesis accuracy as a function of distance from context views. In particular, we use 80 randomly sampled sequences from across categories with 200 frames each, and set the 50th, 100th, 150th views as context, and evaluate the average novel view synthesis accuracy across indices. We find that all approaches peak around the observed frames, but our set-latent representation based methods (GBT, GBT-nb) perform significantly better for query views dissimilar from the context views. This corroborates our intuition that a global set-latent representation is essential for reasoning in the sparse-view setup.

Ablative analysis. We investigate the importance of the design choices made in GBT, by ablating individual components and analysing performance. First, we analyze the effect of learnable geometric bias by fixing $\gamma = 1$ (GBT-fb) during the training process. Next, we remove the geometric bias component (GBT-nb); equivalently $\gamma = 0$. Finally, we replace Plücker coordinates for ray representation with $\mathbf{r} = (\mathbf{o}, \mathbf{d})$. We term this trimmed version of GBT as SRT* (variant of SRT with late camera fusion).

For each ablation (see Table 4), we train a category-specific model from scratch and evaluate results on held-out objects. From Table 4, we see that learnable γ yields some benefit over fixed $\gamma = 1$. However, removing geometry altogether results in a considerable drop in performance. Also, the choice of Plücker coordinates as ray representations improves the predictions in general.

Robustness to camera noise. As the use of the geometric bias requires known camera calibration, we study the effect

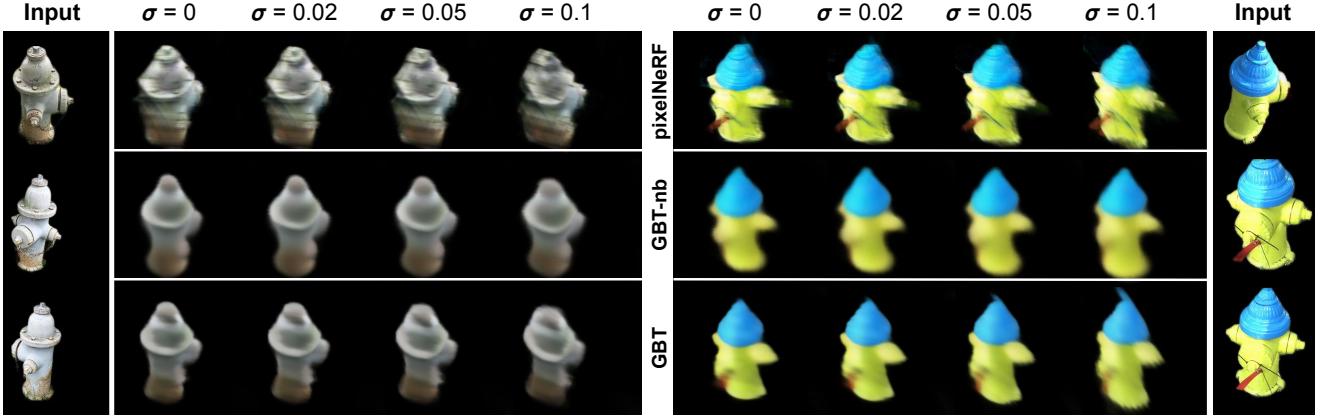


Figure 7. **Effect of camera noise.** Given the 3 input views with noisy camera poses (increasing left to right), we visualize the predictions for a common query view across three methods (rows).

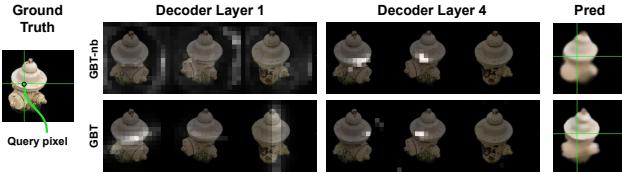


Figure 8. **Attention visualization.** For the query pixel marked in green, we visualize the attention over the input patches for the 1st and the 4th decoder layer. We compare the attention maps of GBT-nb (top) and GBT (bottom), wherein GBT is observed to yield sharper results. See Sec. 4.3.

Table 5. **Evaluation of noisy cameras.** All models are trained on 10 categories and evaluated on the Hydrant category.

	$\sigma = 0$	$\sigma = 0.02$	$\sigma = 0.05$	$\sigma = 0.1$				
	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS
pixelNeRF	20.43	0.26	20.06	0.26	19.20	0.27	18.09	0.29
GBT-nb	21.32	0.24	21.26	0.24	20.85	0.24	19.88	0.25
GBT	22.76	0.22	22.40	0.22	21.43	0.23	19.84	0.25

of noisy cameras on novel view synthesis. Following [12, 20], we synthetically perturb input camera poses to various degrees and analyze the effect of noise during inference (for models trained without any camera noise during training).

We report the results in Table 5, and see that performance degrades across all methods with camera noise. Although GBT-nb degrades more gracefully, the performance of GBT is better until a large amount of noise is added (about 10cm camera motion for a camera unit distance away from an object, and 9 degree rotation). Fig. 7 demonstrates these observations visually.

Visualizing attention. In Fig 8 we visualize attention heatmaps for a particular query ray highlighted in green. In absence of geometric bias (GBT-nb), we observe a diffused attention map over the relevant context, which yields blurrier results. On adding geometric bias (GBT), we observe more concentrated attention toward the geometrically valid regions, resulting in more accurate details.

5. Discussion

Our work introduced a simple but effective mechanism for adding geometric inductive biases in set-latent representation based networks. In particular, we demonstrated that for the task of novel view synthesis given few input views, this allows Transformer-based networks to better leverage geometric associations while preserving their ability to reason about global structure. While our approach led to substantial improvements over prior works, there are several unaddressed challenges. First, unlike projection-based methods, the set-latent representation methods (including ours) struggle to predict precise details and it remains an open question how one can augment such methods to overcome this. Moreover, the use of geometric information in our approach presumes access to (approximate) camera viewpoints for inference, and this may limit its applicability to in-the-wild settings. While our work focused on the task of view synthesis, we believe that the geometry-biasing mechanisms proposed would be relevant for other tasks where a moving camera is observing a common scene (*e.g.* video segmentation, detection).

Acknowledgements. We thank Zhizhuo Zhou, Jason Zhang, Yufei Ye, Ambareesh Revanur, Yehonathan Litman, and Anish Madan for helpful discussions and feedback. We also thank David Novotny and Jonáš Kulhánek for sharing outputs of their work and helpful correspondence. This project was supported in part by a Verisk AI Faculty Award.

References

- [1] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022. 2
- [2] Mark Boss, Andreas Engelhardt, Abhishek Kar, Yuanzhen Li, Deqing Sun, Jonathan T. Barron, Hendrik Lensch, and Varun Jampani. SAMURAI: Shape and material from unconstrained real-world arbitrary image collections. In *NeurIPS*, 2022. 2
- [3] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. TensoRF: Tensorial Radiance Fields. In *ECCV*, 2022. 2
- [4] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. MVSNeRF: Fast Generalizable Radiance Field Reconstruction from Multi-View Stereo. In *ICCV*, 2021. 2
- [5] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised NeRF: Fewer Views and Faster Training for Free. In *CVPR*, 2022. 2
- [6] Patrick Esser, Robin Rombach, and Björn Ommer. Tampering Transformers for High-Resolution Image Synthesis. In *CVPR*, 2021. 2
- [7] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance Fields Without Neural Networks. In *CVPR*, 2022. 2
- [8] Shubham Goel, Georgia Gkioxari, and Jitendra Malik. Differentiable Stereopsis: Meshes from Multiple Views using Differentiable Rendering. In *CVPR*, 2022. 2
- [9] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis. In *ICCV*, 2021. 2
- [10] Yan-Bin Jia. Plücker Coordinates for Lines in the Space. *Problem Solver Techniques for Applied Computer Science, Com-S-477/577 Course Handout*, 2020. 3
- [11] Jonáš Kulhánek, Erik Derner, Torsten Sattler, and Robert Babuška. ViewFormer: NeRF-free Neural Rendering from Few Images Using Transformers. In *ECCV*, 2022. 2, 3, 5, 6, 12
- [12] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. BARF: Bundle-Adjusting Neural Radiance Fields. In *ICCV*, 2021. 8
- [13] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*, 2020. 1, 2, 4, 11
- [14] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.*, 2022. 2
- [15] Phong Nguyen-Ha, Lam Huynh, Esa Rahtu, and Janne Heikkila. Sequential View Synthesis with Transformer. In *ACCV*, 2020. 2
- [16] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf: Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, 2022. 2
- [17] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: Unifying Neural Implicit Surfaces and Radiance Fields for Multi-View Reconstruction. In *ICCV*, 2021. 2
- [18] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common Objects in 3D: Large-Scale Learning and Evaluation of Real-Life 3D Category Reconstruction. In *ICCV*, 2021. 2, 3, 5, 6, 12, 13
- [19] Robin Rombach, Patrick Esser, and Björn Ommer. Geometry-Free View Synthesis: Transformers and No 3D Priors. In *ICCV*, 2021. 2
- [20] Mehdi S. M. Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lucic, Daniel Duckworth, Alexey Dosovitskiy, Jakob Uszkoreit, Thomas Funkhouser, and Andrea Tagliasacchi. Scene Representation Transformer: Geometry-Free Novel View Synthesis Through Set-Latent Scene Representations. In *CVPR*, 2022. 2, 3, 5, 6, 8
- [21] Vincent Sitzmann, Semon Rezhikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light Field Networks: Neural Scene Representations with Single-Evaluation Rendering. In *NeurIPS*, 2021. 3
- [22] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations. In *NeurIPS*, 2019. 2, 3
- [23] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Single-View to Multi-View: Reconstructing Unseen Views with a Convolutional Network. *CoRR abs/1511.06702*, 1(2):2, 2015. 2
- [24] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *ICCV*, 2021. 2
- [25] Aaron Van Den Oord, Oriol Vinyals, et al. Neural Discrete Representation Learning. In *NeurIPS*, 2017. 2
- [26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NeurIPS*, 2017. 2
- [27] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction. In *NeurIPS*, 2021. 2
- [28] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *CVPR*, 2021. 2
- [29] Jiajun Wu, Chengkai Zhang, Xiuming Zhang, Zhoutong Zhang, William T Freeman, and Joshua B Tenenbaum. Learning Shape Priors for Single-View 3D Completion and Reconstruction. In *ECCV*, 2018. 3
- [30] Jimei Yang, Scott E Reed, Ming-Hsuan Yang, and Honglak Lee. Weakly-Supervised Disentangling with Recurrent Transformations for 3D View Synthesis. In *NeurIPS*, 2015. 2
- [31] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume Rendering of Neural Implicit Surfaces. In *NeurIPS*, 2021. 2

- [32] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. PixelNeRF: Neural Radiance Fields from One or Few Images. In *CVPR*, 2021. [2](#), [3](#), [5](#), [6](#), [12](#)
- [33] Jason Zhang, Gengshan Yang, Shubham Tulsiani, and Deva Ramanan. Ners: Neural reflectance surfaces for sparse-view 3d reconstruction in the wild. *NeurIPS*, 2021. [2](#)
- [34] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View Synthesis by Appearance Flow. In *ECCV*, 2016. [2](#)

Appendix A. Additional Random Results

We provide additional results on randomly selected objects across each category, and, provide 360-degree rendering for each figure in the main text. See the project page for video visualizations, and Sec. E for attention map visualizations on more examples across each category.

We observe that while ViewFormer produces plausible images, these are not 3d consistent due to the stochastic nature of the rendering pipeline. While pixelNeRF and NerFormer produce accurate results in the vicinity of the observed context views, the results are inaccurate and implausible under larger camera deviations. Our baseline, GBT-nb produces consistent but blurry results. Finally, GBT improves over GBT-nb by furnishing finer details while preserving consistency across all viewpoints, although there is clear room for improvement in the level of details modeled.

Appendix B. Classwise metrics for Table 2

In Table 2, we present averaged results for $V = 2, 3, 6$ over 10 categories. The per-category metrics are presented in Table 6 (for $V = 2$) and Table 7 (for $V = 6$). Note, the per-category results for $V = 3$ setting is presented in the paper (in Table 1).

Appendix C. Architectural Details

We will make our implementation publicly available for reproducibility. We also describe the implementation details of GBT here. Overall, GBT consists of 3 components - the CNN backbone F_C , GBT Encoder F_E and the GBT Decoder F_D . The input to the model is a set of V posed images $\{\mathbf{I}_i, \mathbf{p}_i\}_{i=1}^V$, and $H \times W$ ray queries $\{\mathbf{r}_j\}_{j=1}^{H \times W}$ generated using the target camera pose \mathbf{p}_q . The model outputs RGB colors for each query ray, which are then reshaped to generate an image of size $H \times W \times 3$.

We use PyTorch for model development. In the discussion below, tensor shapes are annotated in monospace font. We omit the batch dimension for simplicity. Across all models, the image size used is $H = W = 256$.

C.1. GBT

a) Camera-fused patch embedding (F_C). We use a ResNet18 backbone (upto Res3 block) shared across input images to extract patch level features. Concretely, given the V input images $\{\mathbf{I}_i\}_{i=1}^V$ of shape $(V, 3, 256, 256)$, the CNN outputs a feature grid $(V, 256, 16, 16)$.

Each of the 16×16 cells in the feature grid corresponds to a receptive field in the input image. We associate each receptive field with a ray that passes through its center (called as ‘input patch ray’ in the paper). Each input patch ray is represented in the Plücker coordinates $(\mathbf{d}_i^k, \mathbf{m}_i^k) \in \mathbb{R}^6$ - a tensor of shape $(V, 16, 16, 6)$, where the notation

implies i^{th} image’s k^{th} patch. We extract harmonic embeddings [13] over the Plücker coordinates, $h((\mathbf{d}_i^k, \mathbf{m}_i^k))$, using 15 frequencies $f = -6, \dots, 8$. Specifically, we get $h(x) = [\sin(2^f \pi x), \cos(2^f \pi x)]$ for each coordinate. This results in a $6 \times 2 \times 15 = 180$ -d feature representation, yielding a ray embedding tensor of shape $(V, 16, 16, 180)$.

The CNN features $\{[F_C(\mathbf{I}_i)]^k\}$ and the ray embeddings $h((\mathbf{d}_i^k, \mathbf{m}_i^k))$ are concatenated along the channel dimension $\{[F_C(\mathbf{I}_i)]^k \oplus h((\mathbf{d}_i^k, \mathbf{m}_i^k))\}$ that results in a tensor of shape $(V, 16, 16, 436)$. Finally, these features are projected to a 768 dimensional feature space using a linear layer \mathbf{W} (*i.e.* camera fusion). The output of the first stage is therefore camera-fused patch level features $[\mathbf{f}_c]_i^k$ represented by a tensor of shape $(V, 16, 16, 768)$.

b) Geometry-biased scene encoding. We use GBT encoder to embed the global scene context into the patch features. The GBT encoder consists of 8 geometry-biased transformer encoder layers with GELU activation, 12 MHA heads, and 768-d latent feature size. Each MHA module is biased using ray distances as done in Eq. 6.

We construct the query, key and value tokens using flattened patch embeddings. Each query and key token is associated with the patch ray (Plücker coordinates). Therefore, the input to the GBT encoder is patch-level feature tensor of shape $(V * 16 * 16, 768)$ along with the patch ray tensor of shape $(V * 16 * 16, 6)$. Note, the patch ray tensor is the same across all 8 GBT encoder layers, while the learnable weight γ is different for each layer.

The output of the GBT encoder module $\{[\mathbf{f}_e]_i^k\}$ is a tensor of shape $(V * 16 * 16, 768)$ which is the set-latent representation of the scene. Each output token $[\mathbf{f}_e]_i^k$ summarizes the appearance and the geometry of the scene incorporating both local and global features. These output tokens are used as the memory for the GBT decoder module to decode ray queries as described below.

c) Geometry-biased ray decoding. To render an image, we construct Q ray queries using the query camera pose \mathbf{p}_q and use the GBT decoder to predict the RGB color for each pixel. The GBT decoder contains a stack of 4 geometry-biased transformer decoder layers, followed by a shallow MLP. Similar to encoder, each decoder layer consists of 12 MHA heads biased with ray distances, 768-d latent dimensions and GELU activation. The MLP consists of 2 ReLU activated hidden layers (256-d, 64-d) and a sigmoid activated output (0-1 normalized RGB values).

The decoder’s query tokens consist of harmonic ray embeddings $h((\mathbf{d}_j, \mathbf{m}_j))$ and the Plücker coordinates $(\mathbf{d}_j, \mathbf{m}_j)$ for each query ray. Similar to the encoder, we use 15 frequencies which results in a harmonic ray embedding tensor of shape $(Q, 180)$. These are projected to a 768-d feature space (GBT decoder’s input dimension) via a linear

Table 6. **Evaluation of novel view synthesis.** Given $V = 2$ input views, we evaluate the reconstruction quality (PSNR \uparrow and LPIPS \downarrow) of each method on the CO3Dv2 [18] dataset. GBT denotes our proposed approach, and GBT-nb is an ablation.

10 training cat.	Apple		Ball		Bench		Cake		Donut		Hydrant		Plant		Suitcase		Teddybear		Vase		Mean	
	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS
pixelNeRF [32]	18.21	0.36	17.74	0.35	17.59	0.38	17.22	0.38	18.51	0.35	18.44	0.31	19.39	0.36	20.71	0.37	17.74	0.41	19.17	0.34	18.47	0.36
NerFormer [18]	20.11	0.34	16.63	0.37	15.09	0.55	17.23	0.48	20.07	0.36	18.11	0.35	18.37	0.53	19.69	0.46	15.73	0.51	17.79	0.39	17.88	0.43
ViewFormer [11]	20.53	0.25	18.35	0.31	16.58	0.3	17.66	0.33	18.88	0.29	17.93	0.22	18.04	0.31	21.11	0.26	15.87	0.32	21.23	0.21	18.62	0.28
GBT-nb	22.13	0.3	19.83	0.33	18.69	0.36	20.2	0.35	21.0	0.32	21.16	0.24	21.17	0.31	23.02	0.3	19.52	0.35	22.35	0.28	20.91	0.31
GBT	22.96	0.27	21.45	0.28	19.1	0.33	20.71	0.32	21.78	0.29	21.82	0.23	21.29	0.29	23.41	0.28	19.93	0.32	22.28	0.26	21.47	0.29

Table 7. **Evaluation of novel view synthesis.** Given $V = 6$ input views, we evaluate the reconstruction quality (PSNR \uparrow and LPIPS \downarrow) of each method on the CO3Dv2 [18] dataset. GBT denotes our proposed approach, and GBT-nb is an ablation.

10 training cat.	Apple		Ball		Bench		Cake		Donut		Hydrant		Plant		Suitcase		Teddybear		Vase		Mean	
	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS	PSNR	LPIPS
pixelNeRF [32]	23.07	0.24	22.26	0.25	19.94	0.29	21.18	0.28	23.02	0.24	22.62	0.21	21.86	0.26	23.78	0.27	21.35	0.29	23.38	0.22	22.25	0.26
NerFormer [18]	22.03	0.26	18.16	0.33	17.09	0.5	19.53	0.43	23.1	0.29	21.1	0.27	20.62	0.46	21.48	0.43	18.29	0.44	18.73	0.37	20.01	0.38
ViewFormer [11]	22.66	0.23	20.11	0.29	18.06	0.28	19.05	0.31	20.79	0.27	19.62	0.2	18.94	0.29	22.18	0.25	17.57	0.29	22.2	0.21	20.12	0.26
GBT-nb	22.53	0.28	20.59	0.32	19.5	0.34	20.77	0.34	22.15	0.3	21.24	0.23	21.83	0.3	23.43	0.29	19.85	0.34	23.0	0.26	21.49	0.30
GBT	25.5	0.23	23.35	0.26	20.64	0.3	22.34	0.3	23.55	0.27	23.18	0.21	22.46	0.27	24.65	0.26	21.22	0.3	24.06	0.25	23.10	0.26

layer. The keys and values tokens (*i.e.* memory) pertain to the set-latent representation output by the GBT encoder, *i.e.* a tensor of shape $(V * 16 * 16, 768)$.

The GBT decoder outputs a tensor of shape $(Q, 768)$ that consists of decoded ray features for each target pixel. Finally, the MLP predicts a tensor of shape $(Q, 3)$ containing the RGB colors for each queried pixel. During training, we compute L2 reconstruction loss on $Q = 7168$ predicted pixel colors, and at inference we predict the colors for $Q = 256 \times 256$ rays which is reshaped to yield the image tensor of shape $(3, 256, 256)$.

C.2. Ablations

We also propose 3 ablations of GBT in the paper:

GBT-fb (fixed bias). This variant employs a fixed $\gamma = 1$ weight in all the geometry-biased transformer layers as opposed to learning the weight γ . During training this model requires lesser memory overhead since the gradients for γ are no longer computed. At inference, the compute overhead is similar to GBT.

GBT-nb (no bias). In this variant, we remove the geometry-biased transformer layers in the encoder and decoder, and replace them with regular transformer layers (implemented in PyTorch). During training and inference, this model incurs lesser computational overhead than GBT since the ray distances are no longer computed. However, this comes at the cost of quality, which corroborates the need to account for geometry during attention.

SRT*. This variant is closest to SRT, wherein we no longer use geometric bias, nor Plücker ray representations. Rays are represented using the origin \mathbf{o} and direction \mathbf{d} as $\mathbf{r} = (\mathbf{o}, \mathbf{d})$. While the compute overhead is similar to GBT-nb, this model is the least performing among all the variants which demonstrates the benefits of our design choices.

Appendix D. Experimental Details

D.1. Training & Inference

Training. We perform mixed-precision training with 2×NVIDIA A6000 (48GB) GPUs with a batch size of $B = 6$ scenes. For each scene in a batch, we randomly sample $V = 3$ input views and $Q = 7168$ rays from an arbitrary query viewpoint. The predicted pixel RGB color for each query ray is supervised using an L2 reconstruction loss with respect to the ground truth pixel in the query viewpoint. The training is performed till loss convergence which is about 1.6Mil iterations for GBT and about 2Mil iterations for GBT-nb trained on all 10 categories (about 9-10 days).

Inference. At inference we are provided with V posed input images and a query camera pose \mathbf{p}_q . We generate a batch of $H \times W = 256 \times 256$ query rays that are decoded in a single forward pass. The inference time for a single query image with $V = 3$ input views for GBT is 0.09s (~11 FPS), and for GBT-nb is 0.025s (~40 FPS). Compared to GBT, the prior methods exhibit more runtime - pixelNeRF takes 7.3s (~0.13 FPS), NerFormer takes 2.7s (~0.37 FPS), and ViewFormer takes 0.68s (~1.5 FPS), using default parameters (1×A6000 GPU).

D.2. Dataset Splits

We use the CO3Dv2 dataset [18] that contains multi-view images along with camera pose annotations of 51 object categories. We select 10 categories to train our models - [apple, ball, bench, cake, donut, hydrant, plant, suitcase, teddybear, vase]. Additionally, we choose 5 heldout categories - [backpack, book, chair, mouse, remote], which are used to evaluate the generalization of methods. All images are cropped and resized to 256×256 (the camera parameters are modified accordingly).

CO3Dv2 provides three dataset splits - fewview_train, fewview_dev, and fewview_test. Since the fewview_test ground-truth has been redacted for online evaluation, we use fewview_train for training and fewview_dev for testing. We use all available views in each scene in fewview_train split for training.

For computing metrics on the fewview_dev split, we evaluate the models on 32 randomly selected views for the first 10 scenes in each category. We set random seed such that the input and query viewpoints are consistent across all methods. For the viewpoint distance experiment in Fig. 6, we evaluate the average PSNR over 80 sequences across categories for each of 200 query views, with the 50th, 100th, 150th views being the input views.

Appendix E. Attention Visualization

We plot the attention maps for GBT and GBT-nb in Fig. 9-13. Overall, the incorporation of geometric bias results in more concentrated attention towards the geometrically valid regions. For instance, see the attention maps for GBT and GBT-nb in the two hydrant examples in Fig. 9. We hypothesize that concentrated attention toward the relevant context improves the quality of the rendered images.

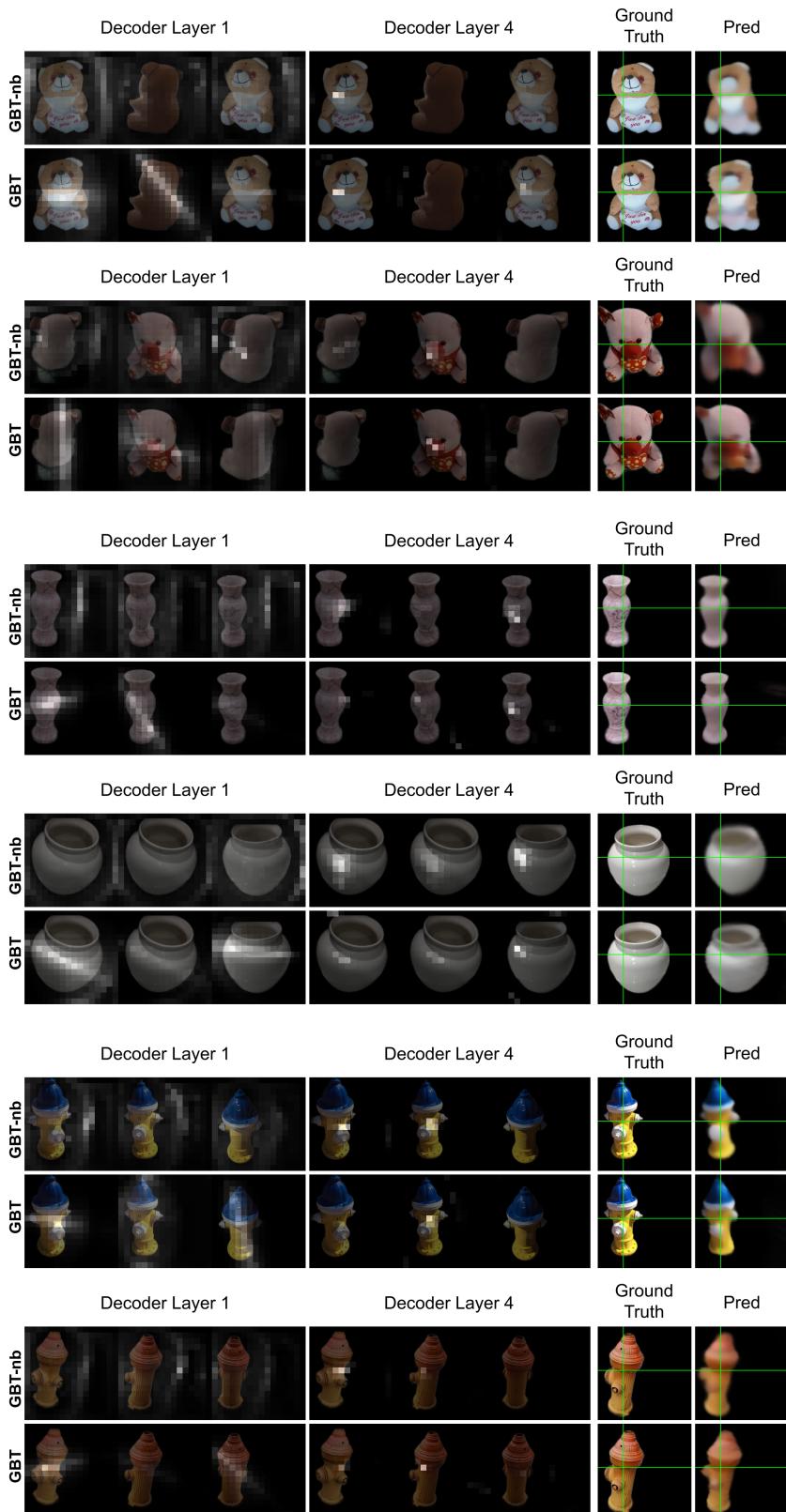


Figure 9. Attention maps for held out objects in teddybear, vase and hydrant categories.

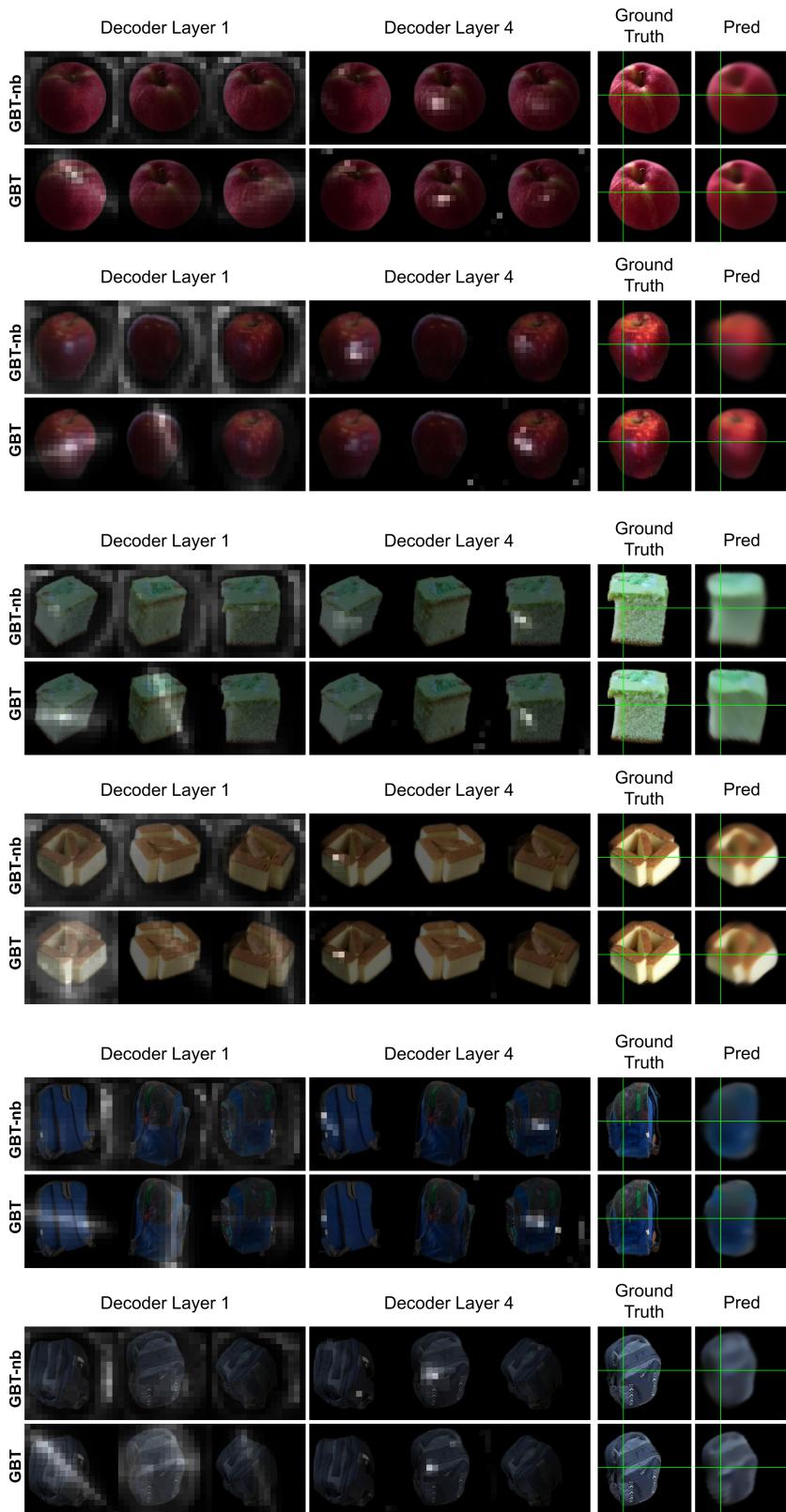


Figure 10. Attention maps for held out objects in apple, cake and backpack categories.

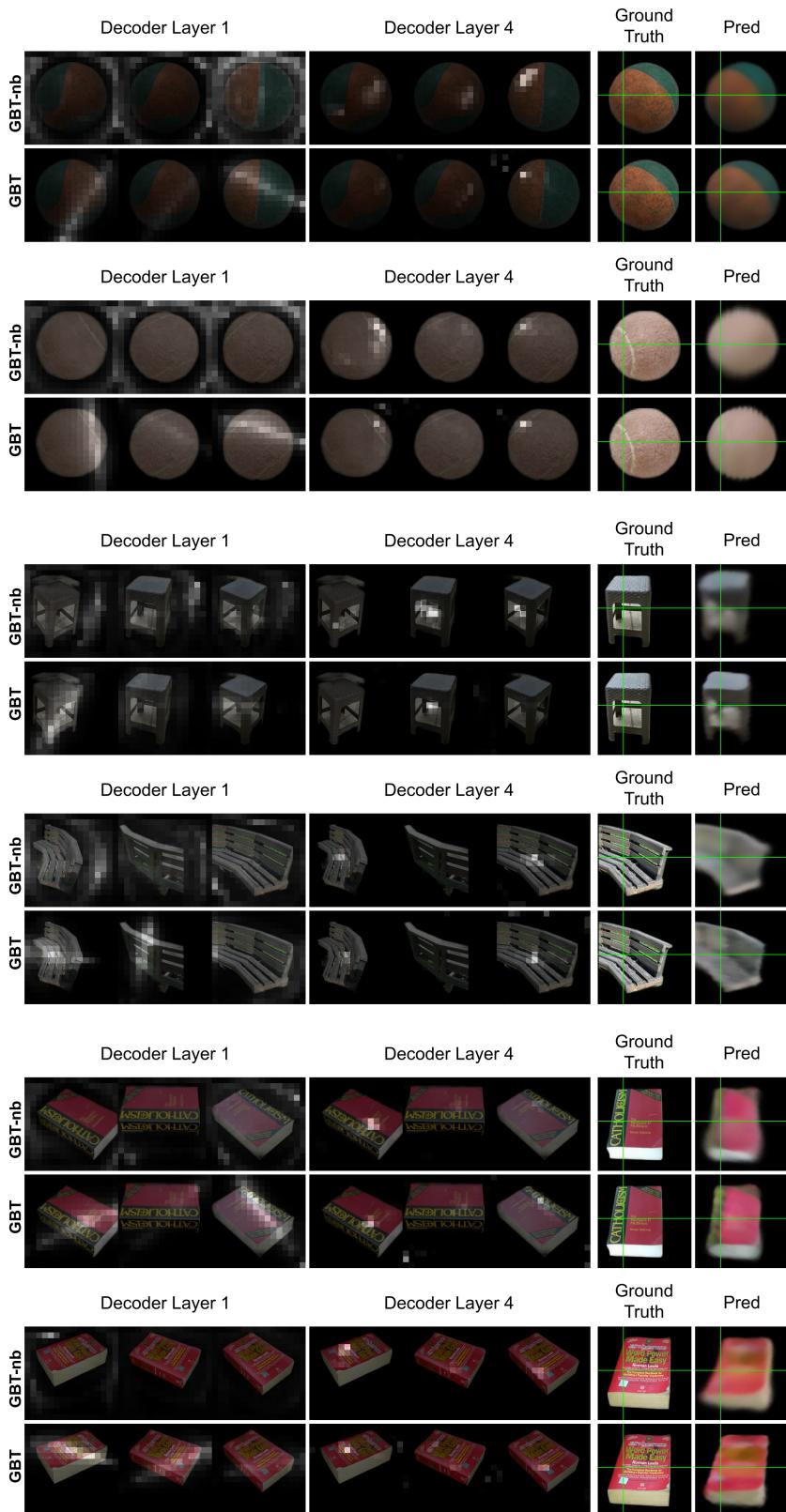


Figure 11. Attention maps for held out objects in ball, bench and book categories.

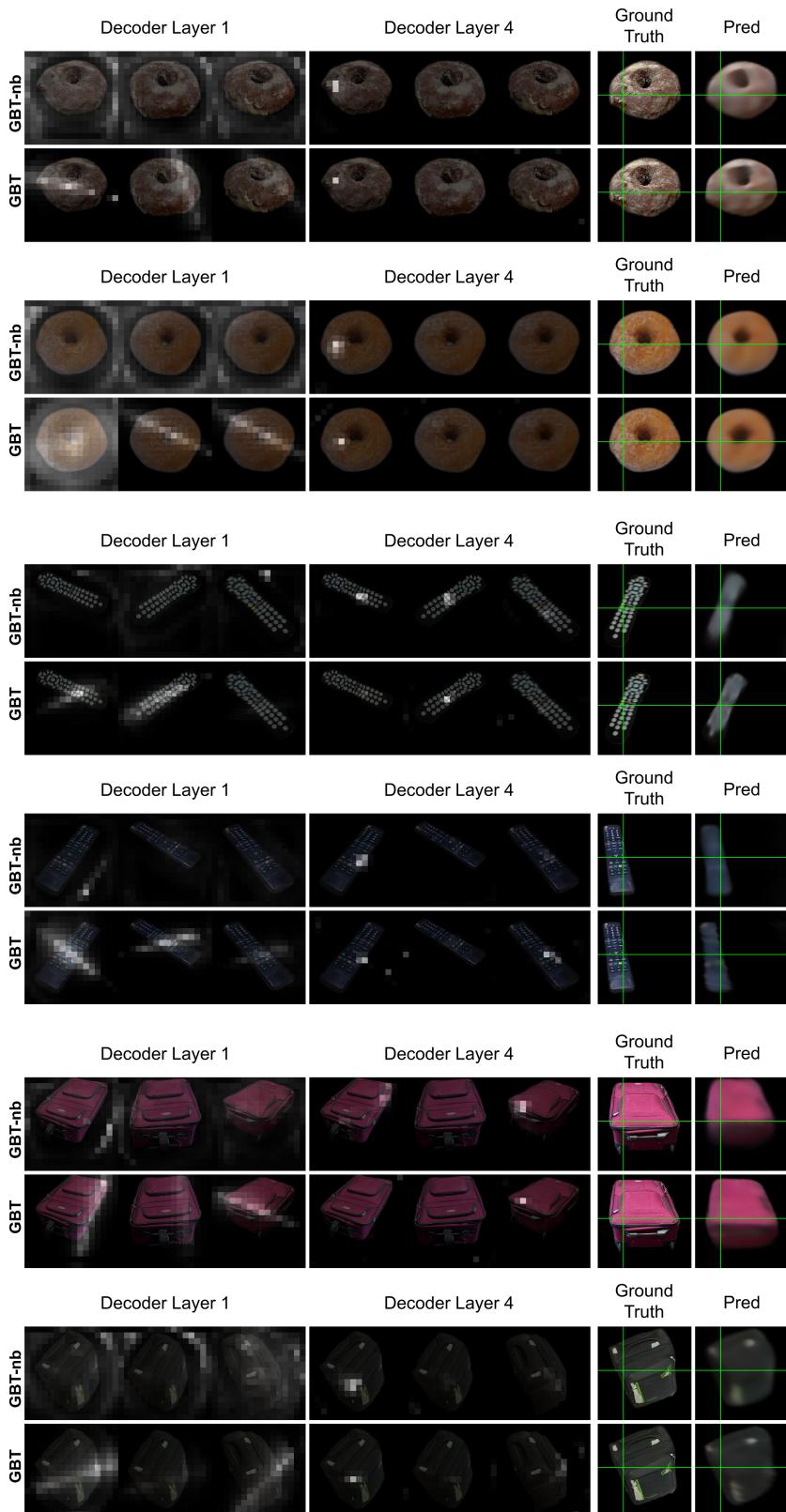


Figure 12. Attention maps for held out objects in donut, remote and suitcase categories.

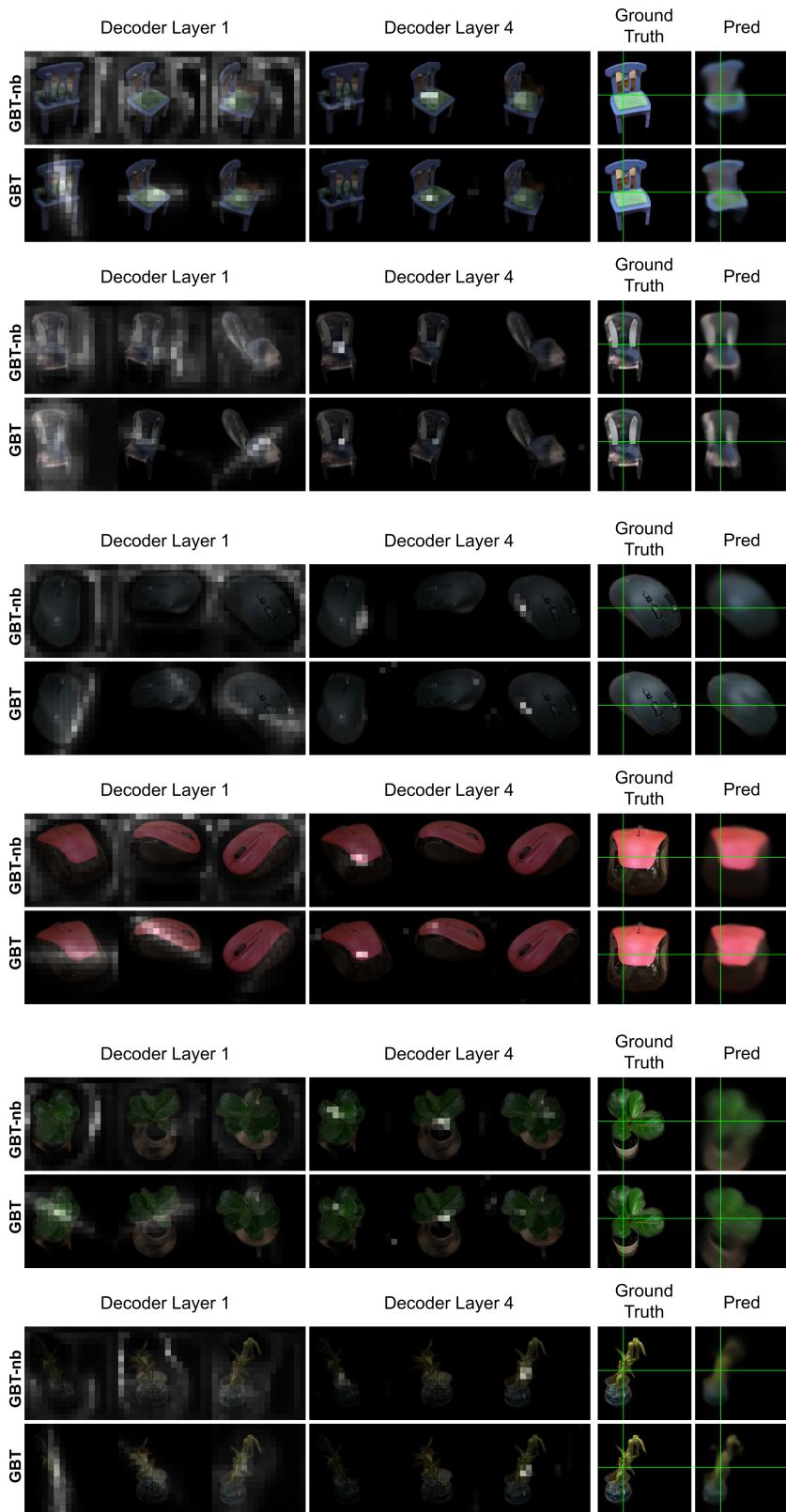


Figure 13. Attention maps for held out objects in chair, mouse and plant categories.