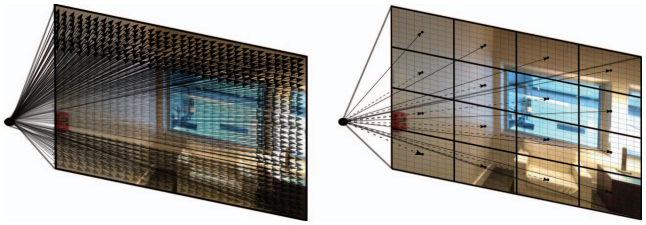
Ray-Patch: An Efficient Querying for Light Field Transformers

Tomás Berriel Martins Javier Civera
I3A, University of Zaragoza, Spain
{tberriel,jcivera}@unizar.es

Abstract

In this paper we propose the Ray-Patch querying, a novel model to efficiently query transformers to decode implicit representations into target views. Our Ray-Patch decoding reduces the computational footprint and increases inference speed up to one order of magnitude compared to previous models, without losing global attention, and hence maintaining specific task metrics. The key idea of our novel querying is to split the target image into a set of patches, then querying the transformer for each patch to extract a set of feature vectors, which are finally decoded into the target image using convolutional layers. Our experimental results quantify the effectiveness of our method, specifically the notable boost in rendering speed for the same task metrics.



Pixel querying Ray-Patch querying (ours)

Figure 1: Light Field Networks sample a ray per pixel to render the target image (left). Our Ray-Patch (right) groups pixels in $k \times k$ patches and samples a ray per patch, reducing the querying cost by a factor of k^2 without loosing accuracy.

1. Introduction

Autonomous agents rely typically on explicit representations of the environment for localization and navigation [4, 32, 23, 3]. However, such approaches lack topological or semantic information, struggle to generalize to changes to novel viewpoints, and do not scale properly to tasks that require reasoning about 3D geometry and affordances.

Implicit representations are better suited to reasoning and hence relevant, as they capture in a continuous space the main high-level features of the scene. Many approaches focus on 3D geometry without topological restrictions using learned occupancy or signed distance functions [7, 24, 18, 17, 25, 9]. Nevertheless, the recent success of neural fields [19] to encode the tridimensional geometry and lighting of a scene has revolutionized the field [33]. They have demonstrated promise in a wide array of tasks such as scene segmentation [41, 13, 5], depth estimation [11], SLAM [31, 42, 1], scene editing [10, 12, 14, 2, 30, 22], and many more [33].

The main limitation of neural rendering is its high computational cost. This is mainly due to 1) the exhaustive querying of the model that is required to recover each pixel of a specific viewpoint, and 2) the need to fit the NeRF model for each scene. Several approaches reduced the 3D querying cost using depth [38, 15, 26, 8], geometry [34, 6, 40, 37], or changing the discretization [16, 39, 20]; and avoided per-scene optimization using latent vectors [34, 16, 6, 40, 37, 11, 14]. Among them, the extensions of Light Field Networks (LFNs) [29] with transformers (Light Field Transformers or LFTs) [28, 27, 11] have shown potential to alleviate both limitations. However, despite significant advances in both qualitative performance and efficiency, all these approaches are still far from being scalable to real scenarios and from real-time performance on low-budget hardware.

In this work we propose Ray-Patch, a novel querying strategy that reduces the computation and memory load of LFTs up to one order of magnitude. Instead of the typical per-pixel querying, we group all pixels in a set of square patches, as shown in Fig.1, and compute a set of feature vectors, which are then grouped and decoded into the target viewpoint. Specifically, it adds to current transformer decoder approaches a convolutional neural networks to reduce the cost of the decoder processing. This results in a drastic reduction in the number of queries, which impacts quadratically in both training and inference cost. In practice, it also allows to train more complex configurations in less time improving both rendering quality and speed.

2. Preliminaries: NeRFs and LFTs

Given a sparse set of multiple views of a scene, a NeRF [19] encodes an implicit continuous volumetric representation of it on the weigths of a MultiLayer Perceptron (MLP). After being optimized for a given scene, this model can then be used to render photorealistic novel views from arbitrary viewpoints. The rendering process involves projecting pixels into rays, sampling 3D positions along each ray, evaluating the MLP network to predict the color and occupancy of the sampled 3D points, and using a rendering equation to aggregate predictions along a ray to estimate the initial pixel value.

Light Field Networks (LFNs) [29] are a variation of NeRFs which directly rely on evaluating the 3D rays into the MLP, removing the aggregation process. To render a novel view they require a single evaluation per pixel rather than the multiple samples required by NeRFs. Light Field Transformers (LFTs) [28, 27, 11] are an extension of LFNs which use a transformer architectures, to encode a latent representation of a novel scene on a latent vector rather than on the model's weights. Therefore, they are able to decode different points of view of novel-scenes without per-scene optimization.

2.1. Transformers

Vanilla transformers [35] are encoder-decoder neural models that incorporate attention mechanisms in their architecture. The encoder performs self-attention on a set of tokens to extract common features. Then the decoder uses cross-attention between the extracted features and a set of queries to compute an output per query. The key component of transformers is the Multi-Head Attention (MHA) layer. In each head h, MHA operates in parallel a scaled dot product attention

$$\text{Attention}_h(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (1)$$

over a set of the three inputs: keys (K), values (V), and queries (Q), projected into a reduced dimension, d_k. To perform self-attention $Q = K = V$ are the tokens to encode. Instead for cross-attention $K = V$ are the extracted features, while Q is the queries to decode.

Computational complexity. The scaled dot product has $\mathcal{O}(n_q n_{kv} d_k)$ complexity, being n_q and n_{kv} the number of queries and keys/values respectively. For self-attention, $n_q = n_{kv}$ hence the complexity is $\mathcal{O}(n_q^2 d_k)$.

2.2. Scene Representation Transformer

The Scene Representation Transformer (SRT) [28] is an LFT, which parametrizes rays with its 3D coordinates and their origin position. Given a set of N input views $\{I_n\}$[1], and their relative camera poses $\{P_n\}$ with camera Instrinsic parameters $\{K_n\}$, the encoder $\mathcal{E}$ generates a set-latent scene representation (SLSR)

$$\mathcal{Z} = \mathcal{E}\left(\{I_n, P_n\}\right), \quad (2)$$

To decode a view of the scene, the light-field based decoder is queried once per-pixel to recover its RGB value. Each query refers to the ray direction and camera center for a given pixel.

The encoder is made of two parts. First, a convolutional network extracts features from the scene images. Then a set self-attention blocks computes common features between the multiple views of the scene to generate a SLSR. The decoder is a two-blocks cross-attention module. It performs attention between the ray queries and the SLSR to generate the RGB pixel values.

Attention cost. With a convolutional encoder which halves the resolution (divides by four the number of queries) three times, $n_q = n_{kv} = N\frac{h \times w}{64}$ for the encoder self-attention block. Therefore the complexity is

$$\mathcal{O}\left(\left(\frac{Nhw}{64}\right)^2 d_k\right). \quad (3)$$

Instead, for the decoder cross-attention block to decode an image, $n_q = h \times w$ and $n_{kv} = N\frac{h \times w}{64}$, therefore the complexity is

$$\mathcal{O}\left(\frac{N\left(hw\right)^2}{64} d_k\right). \quad (4)$$

As a consequence, SRT is limited due to the quadratic scaling of the attention mechanisms cost with respect to the number of input images N and number of pixels (quartic with respect to resolution).

3. Method: Ray-Patch Querying

We propose the Ray-Patch querying to attenuate the quartic complexity of Light Field Transformers with respect to image resolution. Instead of using a ray to query the cross-attention decoder and generate a pixel value, we use a ray to compute a feature vector of a square patch of pixels. Then a convolutional decoder unifies the different patches' feature vectors and recovers the full image. Our approach reduces the number of queries to $\frac{hw}{k^2}$ and the cross-attention cost by the same factor.

[1] We abuse notation here for simplicity, $\{\circ_n\} \equiv \{\circ_1, \ldots, \circ_N\}$

Parametrization. To decode a target view $I_t \in \mathbb{R}^{h \times w \times c}$ of the scene, the view is split into $\frac{hw}{k^2}$ square patches of size $[k, k]$, being the split image now defined as $\{I_{tp} \in \mathbb{R}^{\frac{h}{k} \times \frac{w}{k} \times 3}\}$. Each patch p is parametrized by the location of the camera $\mathbf{o}_t$, and the ray $\mathbf{r_{tp}}$ that passes both by the camera position and the center of the patch. Given the camera intrinsic K_t and extrinsic parameters ${}^{W}T^{C_t} = [R_t|\mathbf{o}_t] \in SE(3)$, the ray $\mathbf{r_{tp}}$ is first computed as the unprojection of the center of patch p in the 2D camera plane, and then translated to the world reference W.

Using Fourier positional encoding [19], the parametrization of each patch is mapped to a higher frequency, to generate a set of queries for the decoder.

$$\{\mathcal{Q}_{tp}\} = \{\gamma(\mathbf{o}_t) \oplus \gamma(\mathbf{r}_{tp})\} \tag{5}$$

Decoder. The decoder $\mathcal{D}$ is a composition

$$\mathcal{D} = (\mathcal{D}_{\text{CNN}} \circ \mathcal{D}_A) \tag{6}$$

of an attention decoder $\mathcal{D}_A$, followed by a convolutional decoder block $\mathcal{D}_{\text{CNN}}$. The attention decoder performs cross-attention between the queries $\{\mathcal{Q}_{tp}\}$ and the SLSR $\mathcal{Z}$, to compute a set of feature vectors

$$\{Z_{tp}\} = \mathcal{D}_A(\{\mathcal{Q}_{tp}\}, \mathcal{Z}) \tag{7}$$

with dimension f. These vectors ensemble a feature map $Z_t \in \mathbb{R}^{\frac{h}{k} \times \frac{w}{k} \times f}$, which is decoded by the convolutional decoder into the target image

$$\hat{I}_t = D_{\text{CNN}}(Z_t). \tag{8}$$

We use a vanilla convolutional decoder $\mathcal{D}_{CNN}$ based on GIRAFFE's decoder [22]. It is a combination of upsampling blocks with convolutions and preliminary outputs.

Integration. The simplicity of the Ray-Patch querying allows to easily integrate it in different LFTs like SRT, OSRT, or DeFiNe. Changing the number of channels of the output of their attention decoders to f, and adding D_{CNN}, they can be used as $\mathcal{D}_A$ to decode the final image as

$$\hat{I}_t = D_{cnn}(D_A(\{\mathcal{Q}_{tp}\}, \mathcal{Z})). \tag{9}$$

The optimization process does not change. The model parameters θ are optimized on a collection of images from different scenes minimizing the Mean Squared Error (MSE) of the generated novel-views for RGB images

$$\mathcal{L}_{rgb} = \frac{1}{hw} \sum_{ij} \left(\hat{I}_t - I_t\right)^2. \tag{10}$$

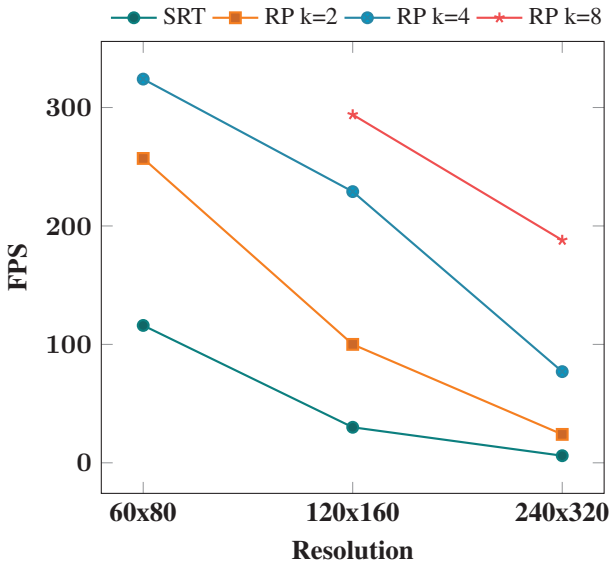


Figure 2: **Single image rendering speed scaling.** The use of the Ray-Patch decoder increase rendering speed at high resolutions up to real-time for both SRT (left).

Attention cost. The proposed Ray-Patch decoder reduces the complexity of the decoders to

$$\mathcal{O}\left(\frac{N(hw)^2}{64k^2} d_k\right), \tag{11}$$

for models with the basic Transformer, like SRT and OSRT, and to

$$\mathcal{O}\left(\frac{hw}{k^2} n_l d_k\right), \tag{12}$$

for PerceiverIO based models, like DeFiNe.

4. Experimental Results

We evaluate Ray-Patch integrating it with SRT for novel view synthesis on the MulstiShapeNet-Easy (MSN-Easy) dataset [30]. The dataset has 70K training scenes and 10K test scenes, with resolution 240×320. Due to the high cost of training SRT, we work at 120×160 and 60×80. Each scene has 3 views sampled at $120°$ steps on a circle around the center of the scene, with extrinsics and intrinsics camera annotations. In each scene there are between 2 and 4 objects of 3 different classes: chair, table, or cabinet.

Given an input image, the model encodes a representation of the scene, and its goal is decoding the other two viewpoints. Due to limited resources, we evaluated on MSN-Easy rather than on MSN-Hard [28, 27], to benefit from its faster convergence (300k steps vs 3M) for more experiments.

	MSN-Easy					
	60×80			120×160		
	SRT	RP-SRT		SRT	RP-SRT	
		k = 2	k = 4		k = 4	k = 8
↑ PSNR	30.98	**31.16**	30.92	**32.842**	32.818	32.306
↑ SSIM	0.903	**0.906**	0.901	0.934	**0.935**	0.929
↓ LPIPS	0.173	**0.163**	0.175	**0.250**	0.254	0.274
↓ Training time	5.6 days	1.7 day	**0.7 days**	6.4 days	1.7 days	**1 day**
↓ Giga FLOPs	48.2	15.8	**7.3**	192.1	28.5	**19.7**
↑ Rendering speed	117 fps	288 fps	**341 fps**	30 fps	275 fps	**305 fps**

Table 1: **Quantitative results on MSN-Easy.** Evaluation of new scene novel view synthesis and computational performance on a simple dataset. While SRT's performance is surpassed only by the configuration with patch size $k = 2$, Ray-Patch increases $\times 3$ and $\times 10$ the rendering speed with minimum impact.

Following Sajjadi et al. [28], rendered views are benchmarked with PSNR, SSIM, and LPIPS. Computational aspects are evaluated measuring image rendering speed, like Sajjadi et al. [28], Float Point Operations (FLOPs) to encode and render an image, and training time. We assume the use of float-32 data, and evaluate time performance on a GPU NVIDIA Tesla V100.

4.1. Computational performance

While our Ray-Patch decoder still has quadratic scaling with n_q, the attenuation performed by the patch size to the number of queries is reflected in a notable boost in the rendering speed, as seen in Fig. 2 and Tab. 1. Furthermore, when increasing the resolution the patch can also be increased, keeping an appropriate rendering speed at higher resolutions.

Comparing rendering speeds for different patches and resolutions in Fig. 2, it can be observed how the improvement tends to saturate for big patch sizes. As a consequence of reducing the number of queries, its impact on the decoder's scaled-dot product complexity will be out-weighted by n_{kv}. For $n_q << n_{kv}$. n_{kv} will set a minimum cost and increasing the patch size over this limit will not be reflected on the rendering speed. Finally, the decrease in n_q implies a reduced vRAM memory peak in the decoder attention, requiring less gpus to train a similar configuration.

4.2. Novel view synthesis

We evaluate two different patch sizes for each resolution: 2 and 4 for 60×80; and 4 and 8 for 120×160.

The experiment metrics shows that all configurations achieve rendering quality on par with base SRT on all metrics. Furthermore, our approach improves rendering speed $\times 10$ for the highest resolution, and reduces training time almost $\times 4$. This is thanks to scaling the attenuation factor *k* together with resolution, compensating for the increasing number of queries, Nevertheless, the size of the patch impacts on the result with smaller patches having better rendering quality, see Tab. 1. For smaller patches, each feature vector is decoded into less pixels than for a bigger patch, and more information is recovered from the same amount of data. Excessively increasing the patch reduces the quality of reconstructed views.

5. Limitations

Our proposed decoder reduces the complexity problem of decoding images with Transformers. Despite that, we cannot decode single pixels and performance may depend on choosing an appropriate patch size. As a simple heuristic to choose the patch, we propose to keep $n_q \sim n_{kv}$, as it has been shown that 1) rendering speed saturates for bigger patches, and 2) too much compression reduces decoding performance. Nevertheless, hyper-parameter tuning may be needed to find the best patch size for each model. Also note that we have only evaluated square patches. Nevertheless this method can also be used with rectangular patches to obtain an intermediate number of queries.

6. Conclusion

In this paper we propose Ray-Patch, which reduces significantly the cost associated to visual transformer decoders. We validate experimentally our approach and its benefits by integrating it into a State-of-the-art LFT model. The models with our Ray-Patch decoder match or even outperform the baseline models in photometric metrics, while at the same time reducing the computation and increasing inference speed one order of magnitude respectively. In addition, this is achieved with a minimum modification to the implementation of the baseline. Reducing the computational footprint of LFTs is essential for its deployment in constrained platforms such as mobile devices or robots, in the same line than works such as [21, 36] did in other architectures and tasks.

References

[1] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6290–6301, 2022.

[2] Miguel Angel Bautista, Pengsheng Guo, Samira Abnar, Walter Talbott, Alexander Toshev, Zhuoyuan Chen, Laurent Dinh, Shuangfei Zhai, Hanlin Goh, Daniel Ulbricht, et al. Gaudi: A neural architect for immersive 3d scene generation. *arXiv preprint arXiv:2207.13751*, 2022.

[3] Michel Breyer, Jen Jen Chung, Lionel Ott, Roland Siegwart, and Juan Nieto. Volumetric grasping network: Real-time 6 dof grasp detection in clutter. In *Conference on Robot Learning*, pages 1602–1611. PMLR, 2021.

[4] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[5] Anh-Quan Cao and Raoul de Charette. Monoscene: Monocular 3d semantic scene completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3991–4001, 2022.

[6] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14124–14133, 2021.

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

[8] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022.

[9] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[10] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021.

[11] Vitor Guizilini, Igor Vasiljevic, Jiading Fang, Rares Ambrus, Greg Shakhnarovich, Matthew Walter, and Adrien Gaidon. Depth field networks for generalizable multi-view scene representation. In *European Conference on Computer Vision (ECCV)*, 2022.

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

[13] Abhijit Kundu, Kyle Genova, Xiaoqi Yin, Alireza Fathi, Caroline Pantofaru, Leonidas J Guibas, Andrea Tagliasacchi, Frank Dellaert, and Thomas Funkhouser. Panoptic neural fields: A semantic object-aware neural scene representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12871–12881, 2022.

[14] Verica Lazova, Vladimir Guzov, Kyle Olszewski, Sergey Tulyakov, and Gerard Pons-Moll. Control-nerf: Editable feature volumes for scene rendering and manipulation. *arXiv preprint arXiv:2204.10850*, 2022.

[15] Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Efficient neural radiance fields for interactive free-viewpoint video. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022.

[16] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020.

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

[18] Mateusz Michalkiewicz, Jhony K Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4743–4752, 2019.

[19] 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 *European conference on computer vision*, pages 405–421. Springer, 2020.

[20] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.

[21] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022.

[22] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021.

[23] Helen Oleynikova, Zachary Taylor, Marius Fehr, Roland Siegwart, and Juan Nieto. Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1366–1373. IEEE, 2017.

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

[25] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *European Conference on Computer Vision*, pages 523–540. Springer, 2020.

[26] Barbara Roessle, Jonathan T Barron, Ben Mildenhall, Pratul P Srinivasan, and Matthias Nießner. Dense depth priors for neural radiance fields from sparse input views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12892–12901, 2022.

[27] Mehdi SM Sajjadi, Daniel Duckworth, Aravindh Mahendran, Sjoerd van Steenkiste, Filip Pavetić, Mario Lučić, Leonidas J Guibas, Klaus Greff, and Thomas Kipf. Object scene representation transformer. *arXiv preprint arXiv:2206.06922*, 2022.

[28] Mehdi SM Sajjadi, Henning Meyer, Etienne Pot, Urs Bergmann, Klaus Greff, Noha Radwan, Suhani Vora, Mario Lučić, Daniel Duckworth, Alexey Dosovitskiy, et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6229–6238, 2022.

[29] Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *Advances in Neural Information Processing Systems*, 34:19313–19325, 2021.

[30] Karl Stelzner, Kristian Kersting, and Adam R Kosiorek. Decomposing 3d scenes into objects via unsupervised volume segmentation. *arXiv preprint arXiv:2104.01148*, 2021.

[31] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6229–6238, 2021.

[32] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

[33] Ayush Tewari, Justus Thies, Ben Mildenhall, Pratul Srinivasan, Edgar Tretschk, W Yifan, Christoph Lassner, Vincent Sitzmann, Ricardo Martin-Brualla, Stephen Lombardi, et al. Advances in neural rendering. In *Computer Graphics Forum*, volume 41, pages 703–735. Wiley Online Library, 2022.

[34] Alex Trevithick and Bo Yang. Grf: Learning a general radiance field for 3d representation and rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15182–15192, October 2021.

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

[36] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2023.

[37] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul P Srinivasan, Howard Zhou, Jonathan T Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. Ibrnet: Learning multi-view image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2021.

[38] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5610–5619, 2021.

[39] Bangbang Yang, Yinda Zhang, Yinghao Xu, Yijin Li, Han Zhou, Hujun Bao, Guofeng Zhang, and Zhaopeng Cui. Learning object-compositional neural radiance field for editable scene rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13779–13788, 2021.

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

[41] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J Davison. In-place scene labelling and understanding with implicit scene representation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15838–15847, 2021.

[42] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12786–12796, 2022.