# ACCELERATED IMPLICIT NEURAL REPRESENTATION WITH SPLIT ENCODER AND MULTISCALE PARTITIONING

*K S Ashin Shanly*

IIT Gandhinagar

## ABSTRACT

Implicit Neural Representation (INR) is an emerging signal representation and rendering technique. These representations are continuous, implicit and differentiable. Their primary advantages are memory efficiency with high spatial resolution and the ability to be incorporated into pipelines based on differentiable learning. However, neural scene representations are slow and cannot represent complex scenes. In this work, we aim to speed up the training time for implicit neural representation networks without compromising the quality of the reconstructed signal. We propose an input-split network architecture that flexibly distributes network resources during training based on the intricacy of the input signal at the given locality. The model learns each underlying dimension of the input coordinates separately and partitions the input space in a multiscale fashion. We show results on large-scale images such as the 64 MP Pluto image captured by the New Horizons space probe and on complex Stanford 3D models varying the maximum number of blocks used in the partition. We present a detailed comparison of results indicating training time speedups of up to $\sim 37.8\%$ more than the state-of-the-art approaches for $1$ MP images, $\sim 25\%$ more for $64$ MP large images, and $\sim 23.72\%$ more for 3D signals.

***Index Terms*—** Implicit Neural Representation, Render, Reconstruct, Occupancy Networks, F-score, PSNR, Encoder, Decoder, Signed Distance Function

## 1. INTRODUCTION

Explicit representations, such as point clouds, volumes, and meshes, involve direct memory storage of spatial and temporal data, which limits their scalability according to the Nyquist sampling requirements. Maintaining critical information necessitates careful selection of the sampling rate to prevent it from falling below a specific threshold, which would result in the loss of crucial details. According to the Nyquist-Shannon sampling theorem [1], the sampling frequency should be at least twice the highest frequency of the signal to avoid information loss. The level of detail within these discrete representations is constrained by the resolution of the chosen input grid.

In contrast, implicit representations employ neural networks that take low-dimensional spatiotemporal coordinates as input and generate learned signals, such as RGB values or volume density, to represent 3D objects, images, and videos. Unlike explicit representations, the complexity of these signals is not constrained by grid size but by the capacity of the learning network. Consequently, implicit representations offer memory efficiency combined with high spatial resolution.

However, neural scene representations encounter challenges in terms of speed and efficiency, particularly when it comes to handling complex scenes and training processes. Preserving detailed information while implicitly representing the input signal poses a significant hurdle. In this study, our aim is to accelerate training times for implicit neural representation networks. Building upon the foundation of ACORN [2], we introduce an input-split coordinate encoder architecture that dynamically allocates resources during training, considering the intricacy of the signal at specific localities. Our model achieves a peak signal-to-noise ratio (PSNR) of approximately $42$ dB in less than $40$ minutes while fitting $1$-megapixel images. In contrast, traditional representations fail to adequately fit the same images, even after several hours of computation. SIREN [3] and ACORN [2] serve as baseline methods for comparison, and our results are extensively documented.

Our contributions are the following.

- We propose a split coordinate encoder architecture that partitions the input in a multiscale fashion based on the complexity of the input signal, resulting in notable speedups of up to $\sim 37.8\%$ more than the state-of-the-art approaches for $1$ MP images, $\sim 25\%$ more for $64$ MP large images, and $\sim 23.72\%$ more for 3D signals.
- We provide a comprehensive quantitative analysis that demonstrates the efficacy of our proposed model in representing large-scale images and complex 3D objects using neural representations. These results indicate the model's capability to handle diverse and challenging visual data.

The paper commences with a concise overview of the relevant literature pertaining to the problem at hand. Subsequently, we delve into a detailed exposition of the proposed

architecture and elucidate the factors contributing to its superior performance compared to existing techniques in Section 3. In Section 4, we present the experimental evaluation, accompanied by quantitative and qualitative results. Lastly, the paper concludes by outlining some of the challenges encountered and identifying potential avenues for further research in the final section.

## 2. RELATED WORKS

A vast body of literature exists within the domain of implicit neural representations (INR). In this section, we aim to provide a concise overview of some relevant works.

Voxels are generally the most typically used representation for 3D problems belonging to discriminative [4, 5, 6] and generative [7, 8] categories. Curless and Levoy [9] used weighted sums of truncated signed distance functions in voxel grids to achieve sub-voxel precision. However, they have a significantly high memory demand for large scenes, and the reconstructed shapes are blocky even after using octree-based approaches [10] to enhance the resolution of the grid. Apart from voxels, point clouds are a viable alternative to 3D representation. They are lightweight and resemble the data collected by sensors such as LiDARs, depth cameras etc. Point clouds, however, do not produce watertight surfaces and fail to efficiently describe the topology of the scene or object at hand. It is also necessary to use post-processing steps [11, 12, 13] to generate the final 3D mesh, which makes it inefficient. Meshes have also been widely used for 3D classification or segmentation [14, 15, 16]. N. Wang [17] considered meshes as 3D reconstruction outputs. These techniques, however can only generate topologically simple meshes [17], cannot guarantee closed surfaces [18] and may lead to self-intersecting meshes as output.

Explicit models (point cloud, voxel, and triangle mesh) have discretization errors; conversely, implicit representations can deal with intricate shapes with varying topologies by representing them continuously. To attain a detailed representation of a 2D image or 3D object, higher memory usage and computational effort are required. Implicit-based methods use a deep network that maps 3D coordinates to signed distance [19] or occupancy [20] values at arbitrary query points. Learning a continuous field of signed distance values or occupancy values can attain an infinitely higher resolution without considerable growth in memory footprint. The capability of neural networks to implicitly represent objects [19], shape parts [21], or scenes [22] have been extensively shown in several latest works. Zhumekenov [23] introduced Fourier neural networks, which were designed to replicate the Fourier transform using single-hidden layer networks. Mildenhall [24] encoded the appearance of objects using multi-view 2D images with the help of neural rendering [25]. DeepSDF [19] used a single global latent code and failed to represent fine details satisfactorily. Capturing high-frequency features has been focused on many follow-up works since then. Sitzmann [3] used periodic sinus activation functions. Martel introduced a hybrid implicit-explicit neural network architecture, leveraging the concept of multiscale block-coordinate decomposition, for implicit signal representation in their work [2]. Meanwhile, Ruofan Liang [26] proposed a split multi-layer perceptron (MLP) architecture, leading to accelerated inference and training times for INR networks.

Motivated by the pioneering contributions of these studies, we explore architectures incorporating split coordinate encoders and multiscale decomposition modules. These architectural enhancements are particularly beneficial for applications involving implicit neural representations of signals with large memory footprints.

## 3. METHOD

### 3.1. Overview

The proposed architecture incorporates a block parametrization module, utilizing a quadtree for 2D signals and an octree for 3D signals, to facilitate the decomposition of coordinates into blocks. These block coordinates are subsequently passed through a split-coordinates module, which independently decomposes the input coordinates into each dimension. The resulting decomposed signal is then directed to a split coordinate encoder. This encoder operates by receiving each individual dimension as input through distinct input layer branches, as illustrated in Figure 1. The split coordinate encoder generates a discrete feature grid from the input. The local coordinates within each input block are then determined through linear interpolation and by passing them through a shallow feature decoder.

### 3.2. Multiscale Input Partitioning

Let's delve into the intricate details of our architecture. The initial phase involves partitioning the input space using a tree-based technique. For 2D inputs, we utilize the quadtree, while for 3D inputs, the octree is employed. This partitioning allows us to distinguish two key types of coordinates: global coordinates, which pertain to individual blocks, and local coordinates, which specify the relative positions of input values within each block.

To ensure optimal resource allocation, our approach focuses on directing computational resources toward learning intricate features of the input signal, prioritizing complex regions over flat-textured areas. Conventional neural representations, unfortunately, fail to allocate resources adaptively and require an equal number of forward passes through the network for rendering each component of the input signal.

To address this limitation and achieve efficient resource allocation, we propose an adaptive method that adjusts the scale of each input block. We encode the scale of each block as the last element of its global index. Every input coordinate

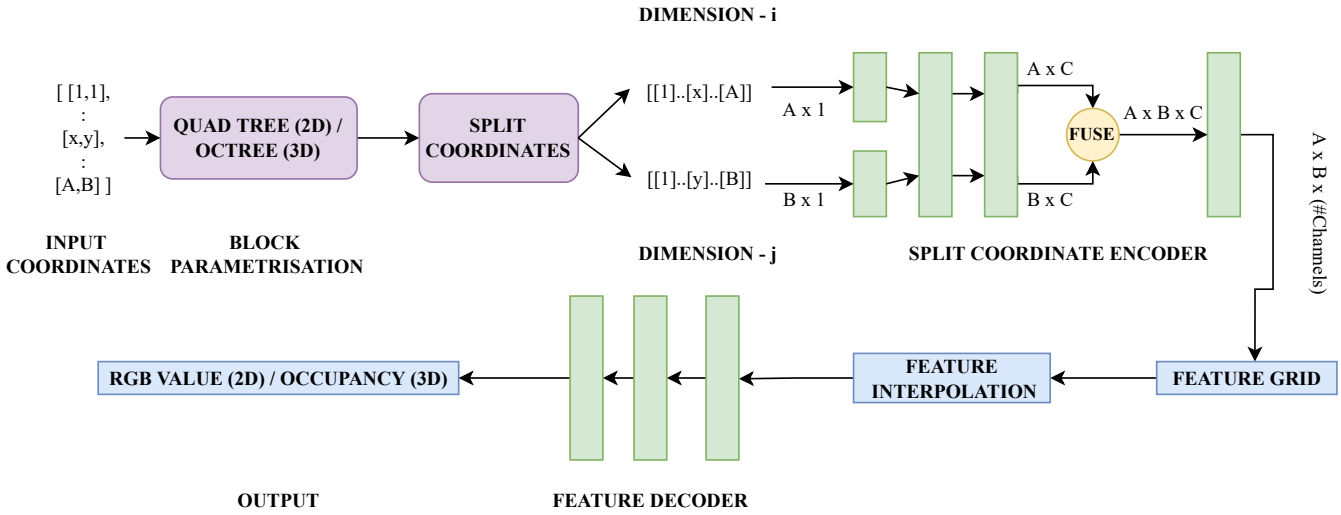
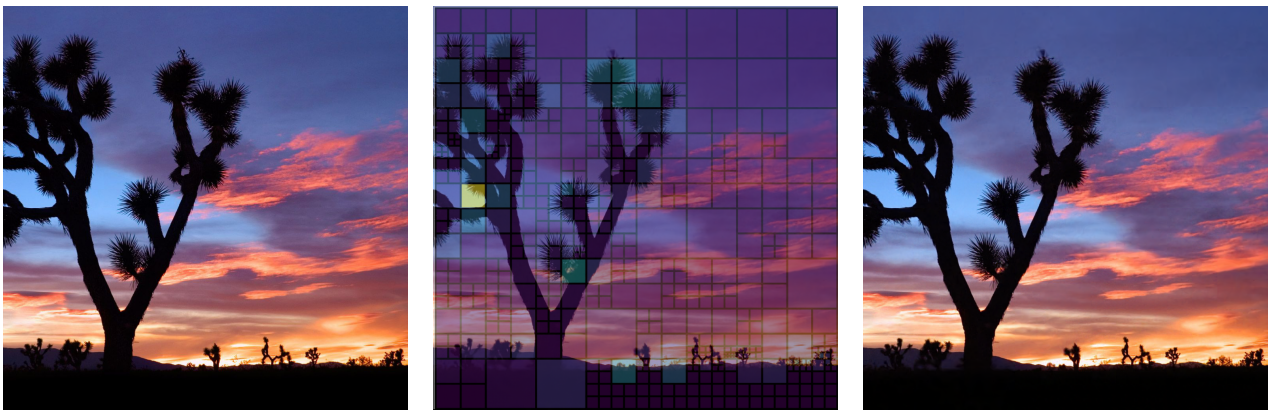



**Fig. 1**. Proposed Network Architecture.



(a) Ground Truth. (b) Tiling. (c) Ours.

**Fig. 2**. 1 MP image fitting. Figure 2(a) is the ground truth image. Figure 2(b) shows the final block decomposition color-coded based on intricacy. More intricate regions of the image are partitioned into more blocks than less intricate regions. Figure 2(c) shows our reconstruction result.

is defined at one scale and associated with one block. The selection of the block that each input value is associated with is an optimization problem. This adjustment is achieved by solving an integer linear problem, akin to the approach utilized in ACORN [2]. We have extensively investigated and elucidated this process in Equation (1).

$$\min_{I_1,...,I_{N_g}} \sum_i w_i^\top I_i, \tag{1}$$

$$\text{w.r.t} \quad I_i^G + I_i^M + I_i^S = 1 \; \forall \; B_i, \text{ and}$$

$$(\sum_i (1/N_s) I_i^M + I_i^E + N_s I_i^S) \leq M_B,$$

where $I_i = (I_i^E, I_i^M, I_i^S)^\top$ and $w_i = (w_i^E, w_i^M, w_i^S)^\top$.

Here, $N_g$ is the total number of blocks in the partition, and $I_i^E, I_i^M, I_i^S$ are the binary decision variables that indicate if a block $B_i$ stays the same, merges with its siblings, or splits to create more blocks respectively. For each block, there is a vector of weights associated, which represents the fitting error due to the decision to merge ($w^M$), split ($w^S$), or stay the same ($w^E$). Here, $N_s$ represents the number of sibling blocks created as a result of any of these three actions. When all the siblings of a particular level are ready for merging, we set a variable $I_i^G$ (defined in Equation (2)) to value 1.

$$I_i^G = (1/N_s)(I_i^M + \sum_{j \in S(i)} I_j^M), \tag{2}$$

where $S(i)$ represents the group of siblings of block $B_i$.

$M_B$ is the maximum number of blocks that is fixed beforehand and is treated as a hyperparameter. After feature bilinear interpolation and decoding, the block error ($B_{err}$) is calculated by taking the mean of the L1 loss of all local coordinates inside the block, which is then used to update the weight vector of the same block. We define $B_{err}$ in Equation (3),

$$B_{err}[i] = \frac{\|y^i - y^i_{GT}\|}{N_l}; \; i = 1, 2, ..., N_l, \tag{3}$$

where $N_l$ is the number of local coordinates in block $B_i$.

The overall loss ($\mathcal{L}$) is computed as the mean of all the block errors in the partition. This loss is back propagated to update the input split encoder and lightweight decoder.

$$\mathcal{L} = \frac{\sum_{i=1}^{N_g} B_{err}[i]}{N_g}, \tag{4}$$

By adopting this approach, our model can intelligently allocate computational resources based on the complexity of each block in the input. Consequently, the model can dedicate more resources to blocks with intricate features, allowing for a deeper understanding and representation learning of these regions.

Overall, the combination of tree-based partitioning, global and local coordinates, and the adaptive resource allocation method constitutes a powerful and efficient architecture that significantly improves the model's ability to handle complex input signals.

### 3.3. Split Coordinate Encoder

The Split Coordinate Encoder proves to be an effective architectural element in our model, providing several advantages in processing input data. Let's delve deeper into the reasons behind its effectiveness. Consider an image with dimensions A×B, where the set of input coordinate points is represented as $I \in R^{N \times D}$. Here, $N$ represents the number of coordinate points, and $D$ signifies the dimension of each point.

In conventional MLP models, each of the $N$ input points is queried individually, resulting in a process that scales with the dimensions A×B. However, the split coordinate encoder module introduces a clever partitioning strategy. In this approach, the first layer of the encoder is divided into $D$ separate input branches, where each branch corresponds to one dimension of the input coordinate. Consequently, each dimension of the input coordinate is individually fed into its corresponding input branch. This decomposition transforms the original input coordinate grid $I \in R^{N \times D}$ into a set of decomposed input grids $I^{(i)} \in R^{S^i \times D^i} | i = 1, 2, 3, ...$, where $i$ denotes the input branch, $S^i$ denotes the number of input points assigned to branch $i$, and $D^i$ signifies the dimension of each decomposed coordinate point inputted to branch $i$.

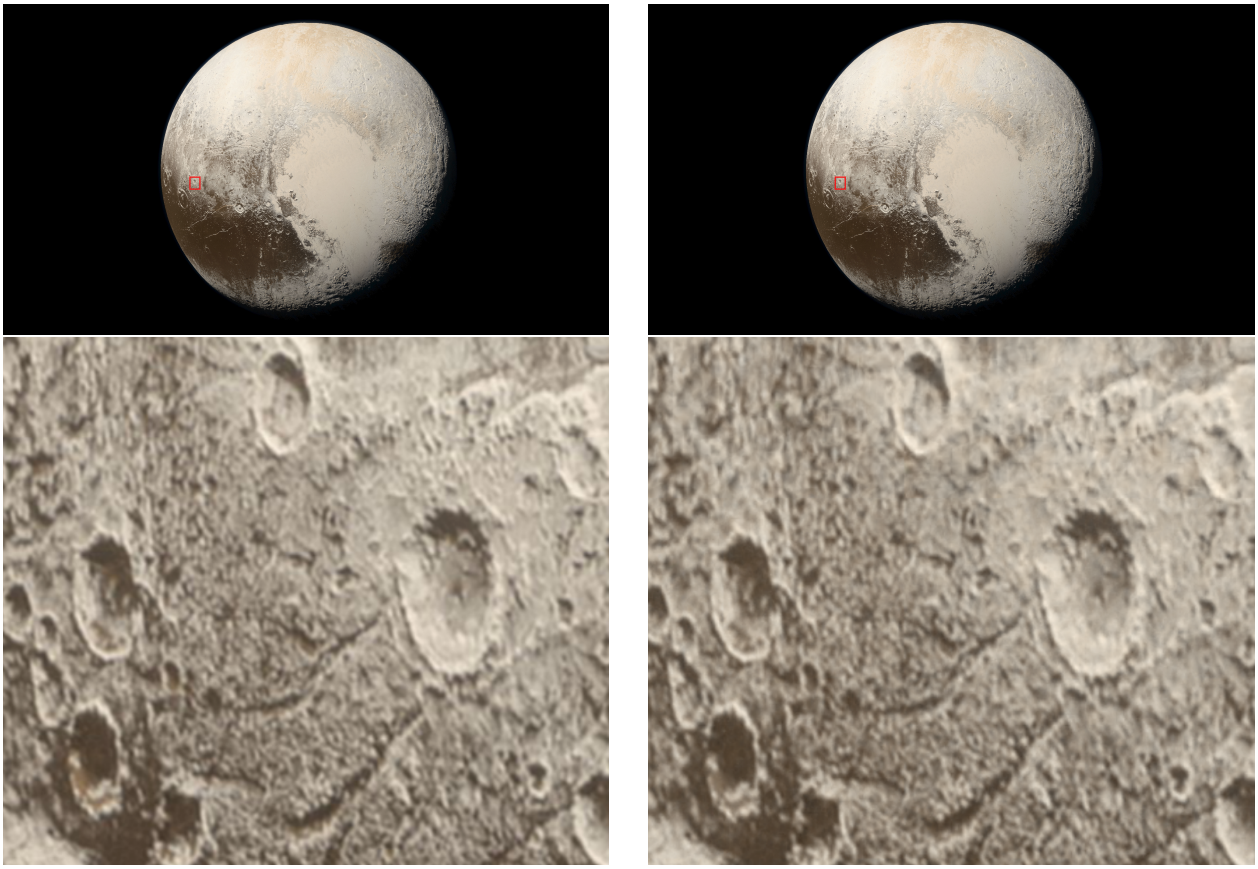
By splitting the initial layer of the encoder, the computational complexity of the process is significantly reduced from A×B to A+B. This reduction in complexity leads to more efficient computations and enables the model to handle higher-dimensional inputs more effectively.

Following the decomposition step, the resulting features from the different branches are fused together using the outer product operation. The fused features are then passed to the final layer of the fully connected network to generate the signal values.

The effectiveness of the split coordinate encoder is supported by the work of Liang et al. [26], who have quantitatively demonstrated the concept of learning different coordinate dimensions individually rather than treating the input signal as a whole. This approach allows the model to gain a deeper understanding of the input's multi-dimensional structure and leads to improved representation learning, which positively impacts the overall performance and efficiency of our model.

## 4. EVALUATION

For the evaluation of the baselines and the proposed model, we employed an NVIDIA RTX 2080 Ti GPU, conducting the assessment over a span of $100,000$ training epochs. In this



(a) Ground Truth. (b) Ours.

**Fig. 3**. 64 MP large scale image fitting.

Ground Truth SIREN [3] ACORN [2] Ours

**Fig. 4**. Comparison against state-of-the-art for 3D volume fitting (Stanford Lucy model).

**Table 1**. Average of fitting results for hundred 1 MP images.

| Method | A $T_{training}$(hrs) | PSNR |
|---|---|---|
| SIREN [3] | 6 h | 39.5 dB |
| ACORN [2] | 45 min | 42.00 dB |
| **Ours** | **28 min** | **42.00 dB** |

experiment, the input split MLP consisted of 4 hidden layers and 256 hidden units, serving as the coordinate encoder, while the feature decoder utilized a lightweight MLP with a single hidden layer. The reported training times represent the average of over ten runs to ensure statistical robustness. To ensure fairness in the comparison, we set the model hyperparameters, including the learning rate (0.001), number of epochs $(100,000)$, and batch size (1), uniformly across all networks.

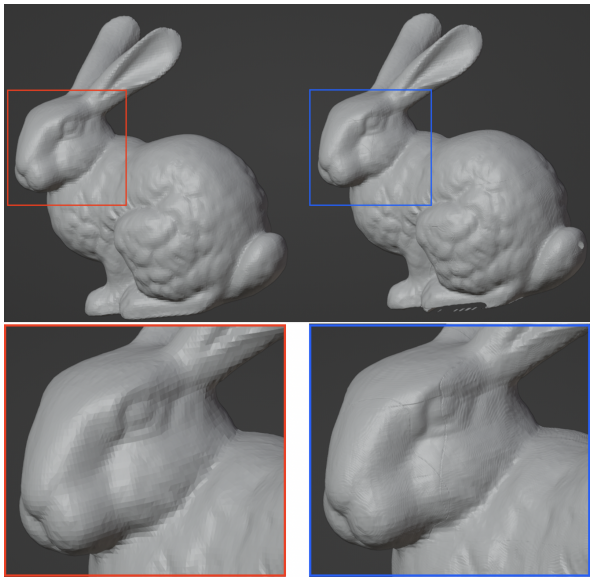
### 4.1. Image Fitting Results

In our research, we establish a maximum of 512 blocks to decompose the input for 2D images. The qualitative reconstruction outcomes are demonstrated using a representative 1MP image, as depicted in 2. Notably, Figure 2(b) showcases the final block decomposition attained for the corresponding image. Our proposed approach intelligently allocates more blocks to areas of the signal with higher complexity while employing relatively fewer blocks for less intricate regions. As a result, this distribution of network resources allows for more focused learning of complicated regions, surpassing the learning capacity of less complex areas. Furthermore, the reconstruction result for a larger-scale image of size 64MP captured by the New Horizons space probe is shown in 3.

The effectiveness of our proposed model is demonstrated through the quantitative experimental results, as presented in both Table 1 and Table 2. Extensive testing on a dataset of one hundred 1 MP images yielded an impressive average PSNR value of 42.00 dB. Remarkably, the average training time surpassed ACORN's [2] performance by approximately $\sim 37.8\%$, as evidenced in 1.

Furthermore, we conducted evaluations on a challenging dataset consisting of one hundred 64 MP high-resolution large-scale images. Our model exhibited exceptional performance in this scenario, achieving an average peak signal-to-noise ratio (PSNR) value of 39.00 dB. This outperformed ACORN [2] by over $\sim 25\%$ in terms of required training time, as indicated in the results.

One of the key advantages of our architecture lies in its improved efficiency, as it demands the same memory footprint as ACORN, which is 2.4 GB. In stark contrast, the same set of images necessitates a substantial memory demand of 30 GB when utilizing SIREN. This substantial reduction in memory requirements highlights the practicality and scalability of our proposed model.



(a) Ground Truth. (b) Ours.

**Fig. 5**. Stanford 3D Bunny model.

**Table 2**. Average of fitting results for hundred 64 MP images.

| Method | A $T_{training}$(hrs) | PSNR |
|---|---|---|
| SIREN [3] | 37.0 h | 33.5 dB |
| ACORN [2] | 4 h 52 min | 38.92 dB |
| **Ours** | **3 h 50 min** | **39.00 dB** |

Figure 8 shows the variation of 64 MP image PSNR while training. It can be seen that ACORN [2] and the proposed model converge to similar reconstruction accuracies. The slight drop in PSNR (3 dB) for image representation might be due to the presence of more high-frequency variations in 2D image signals. This may result in loss of information when the input is split into individual dimensions in the split coordinate encoder. Our model is significantly faster while requiring the same amount of memory taken by ACORN (2.4 GB) as compared to 30 GB memory required by SIREN for the same image. The acceleration in training time required is noted to increase in the case of bigger images.

### 4.2. 3D Fitting Results

In our investigation of 3D objects, we have set the maximum number of octants to 256 as a configuration parameter. To represent these 3D objects effectively, we leveraged occupancy networks [20] and evaluated the reconstruction quality using the F-score metric. The evaluation process encompassed both qualitative and quantitative analyses of complex models obtained from the Stanford 3D Scanning Repository [27], after conducting 1000 epochs of training. The resulting reconstruc-

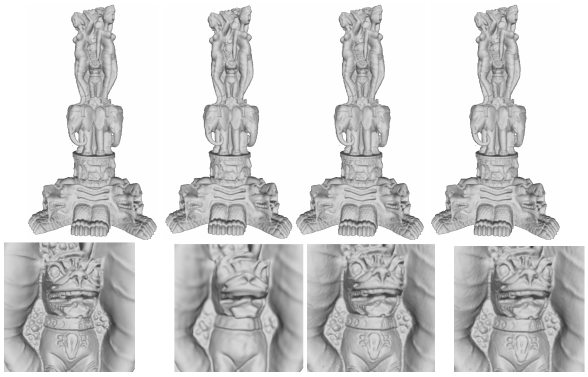



**Fig. 6**. Comparison against state-of-the-art for 3D volume fitting (Stanford Thai statue).

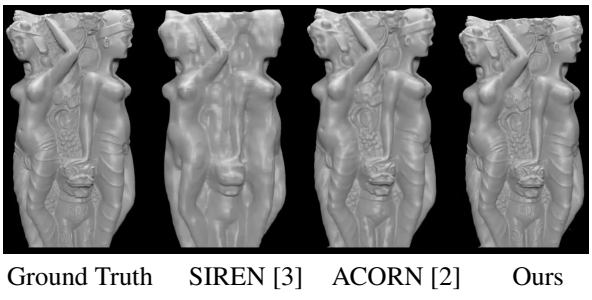



**Fig. 7**. Zoomed in version of Stanford Thai statue results.

**Table 3**. Average of 3D fitting results for hundred Stanford 3D models

| Method | A $T_{training}$(hrs) | F-score |
|---|---|---|
| SIREN [3] | 30.0 h | 0.996 |
| ACORN [2] | 7 h 52 min | **0.999** |
| **Ours** | **6.0 h** | **0.999** |

tion outcomes are showcased in Figures 4, 5, 6, and 7.

It is also noted from Figure 9 that our model converges similar to ACORN with a minor difference in occupancy loss observed but with decreased training time required. The qualitative assessment of our model's performance reveals that it achieves a reconstruction quality that not only surpasses SIREN [3] but is also comparable to ACORN [2]. This achievement is a testament to the robustness and accuracy of our proposed method in capturing intricate 3D structures.

The quantitative results further reinforce the promising nature of the findings. The proposed model demonstrates training speeds that are up to $23.72\%$ faster than ACORN [2] and an impressive over $89\%$ faster than SIREN [3] when applied to 3D object representations. This significant acceleration in training times is a direct consequence of the reduction in model parameters facilitated by the innovative split coordinate encoder and the efficient multiscale input grid partitioning. These architectural enhancements effectively streamline the fitting process, enabling faster convergence and reducing computational overhead.

In summary, our research presents compelling evidence of the capabilities of our model in handling 3D object reconstruction tasks. The qualitative visualizations of Figures 4, 5, 6, and 7 exhibit its prowess in faithfully reproducing intricate shapes and structures. Moreover, the quantitative analyses, as summarized in Table 3, demonstrate its superiority in terms of reconstruction quality and training efficiency when compared to state-of-the-art approaches like SIREN [3] and ACORN [2].

## 5. CHALLENGES AND FUTURE WORKS

The prevalence of high-frequency variations in 2D image signals is a common occurrence. Consequently, the decomposition of input signals into individual dimensions using the split coordinate encoder often leads to the loss of pertinent information. This predicament can be addressed by introducing additional parameters into the network, such as augmenting the feature size of each input-splitting branch or increasing the number of hidden units within the pre-fusion layer.

During the training process, it is frequently observed that fitting losses exhibit spikes. However, the introduction of a differentiable approach to updating the input partitioning can effectively mitigate this issue, consequently facilitating accelerated convergence rates. Currently, our representation of

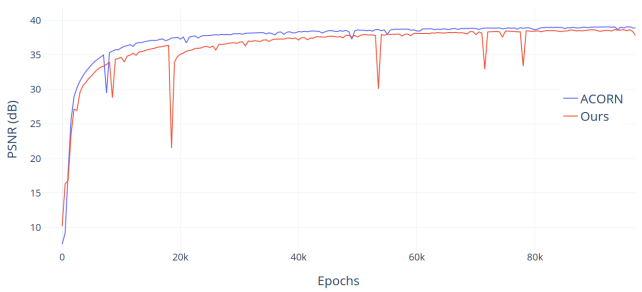



**Fig. 8**. Train image PSNR. The plot shows PSNR as a function of training epochs for the image shown in Fig. 4. Our model converges similar to ACORN, but requires significantly lesser training time.

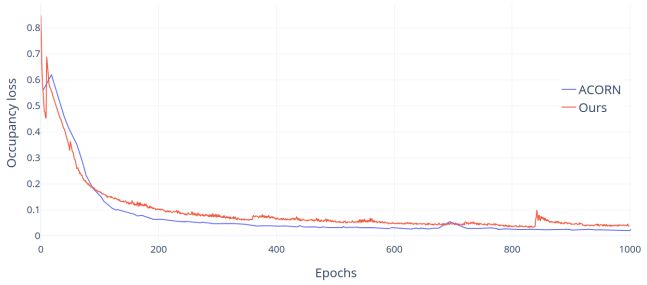



**Fig. 9**. 3D Occupancy loss. The plot shows the variation of 3D occupancy loss while fitting on the Stanford Thai statue model. Our model converges similar to ACORN, but faster.

each input block solely relies on a singular scale. An alternative approach that involves employing lightweight multilayer perceptrons (MLPs) at multiple scales for representing each block may yield superior outcomes in terms of expediting training and rendering durations.

## 6. CONCLUSION

Efficiently representing the underlying signal stands as a pivotal aspect in the amalgamation of computer graphics and neural computing disciplines. The realm of implicit neural representation techniques has witnessed rapid progress, opening up novel avenues in diverse computer graphics and processing domains. In this context, we present a pioneering architectural framework that adeptly assimilates input signal decomposition and multilevel partitioning. Our proposed model showcases commendable memory efficiency, comparable accuracy levels, and expedited training in comparison to contemporary state-of-the-art methodologies.

Empirical validation through rigorous experimentation underscores the efficacy of our approach in expeditiously fitting complex two-dimensional images and three-dimensional shapes, outperforming the current state-of-the-art ACORN method [2]. Our comprehensive assessment reveals notable performance acceleration across diverse experimental setups, thus establishing a significant milestone toward swifter neural approaches in imaging, simulation, and rendering realms.

## 7. REFERENCES

[1] C.E. Shannon, "Communication in the presence of noise," *Proceedings of the IRE*, vol. 37, no. 1, pp. 10–21, jan 1949.

[2] Julien N. P. Martel, David B. Lindell, Connor Z. Lin, Eric R. Chan, Marco Monteiro, and Gordon Wetzstein, "Acorn: Adaptive coordinate networks for neural scene representation," *ACM Trans. Graph. (SIGGRAPH)*, vol. 40, no. 4, 2021.

[3] A. Bergman D. Lindell V. Sitzmann, J. Martel and G. Wetzstein., "Implicit neural representations with periodic activation functions.," *Advances in Neural Information Processing Systems, vol. 33*, 2020.

[4] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition.," *IEEE International Conf. on Intelligent Robots and Systems(IROS)*, 2015.

[5] M. Nießner A. Dai M. Yan C. R. Qi, H. Su and L. Guibas, "Volumetric and multi-view cnns for object classification on 3d data.," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[6] S. Song and J. Xiao, "Deep sliding shapes for amodal 3d object detection in rgb-d images.," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[7] J. Gwak K. Chen C. B. Choy, D. Xu and S. Savarese, "3d-r2n2: A unified approach for single and multi-view 3d ob-ject reconstruction.," *European Conf. on Computer Vision (ECCV)*, 2016.

[8] M. Rodriguez R. Girdhar, D. F. Fouhey and A. Gupta, "Learning a predictable and generative vector representation for objects.," *European Conf. on Computer Vision (ECCV)*, 2016.

[9] B. Curless and M. Levoyk, "A volumetric method for building complex models from range images," *23rd annual conference on Computer graphics and interactive techniques*, 1996.

[10] A. Dosovitskiy M. Tatarchenko and T. Brox., "Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs.," *IEEE International Conf. on Computer Vision (ICCV)*, 2017.

[11] H. Rushmeier C. Silva F. Bernardini, J. Mittleman and G. Taubin, "The ball-pivoting algorithm for surface reconstruction.," *IEEE Trans. on Visualization and ComputerGraphics (VCG)*, 1999.

[12] F. Calakli and G. Taubin., "Ssd: smooth signed distance surface reconstruction.," *Computer Graphics Forum*, 2011.

[13] M. Kazhdan and H. Hoppek, "Screened poisson surface reconstruction," *ACM Transactions on Graphics (ToG)*, 2013.

[14] Y. LeCun A. Szlam M. M. Bronstein, J. Bruna and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data.," *Signal Processing Magazine,*, vol. 34, no. 4, pp. 18–42, 2017.

[15] D. Zou K. Guo and X. Chen., "3d mesh labeling via deep convolutional neural networks.," *SIGGRAPH*, 2015.

[16] Y. Zhang P. Wang, Y. Gan and P. Shui., "3d shape segmentation via shape fully convolutional networks.," *Computers Graphics*, 2017.

[17] Z. Li Y. Fu W. Liu N. Wang, Y. Zhang and Y. G. Jiang., "Pixel2mesh: Generating 3d mesh models from single rgb images.," *European Conf. on Computer Vision (ECCV)*, 2018.

[18] Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan Russell, and Mathieu Aubry, "AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[19] Julian Straub Richard Newcombe Jeong Joon Park, Peter Florence and Steven Lovegrove., "Deepsdf: Learning continuous signed distance functions for shape representation," *CVPR*, 2019.

[20] Michael Niemeyer Sebastian Nowozin Lars Mescheder, Michael Oechsle and Andreas Geiger, "Occupancy networks: Learning 3d reconstruction in function space," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[21] Daniel Vlasic Aaron Sarna William T Freeman Kyle Genova, Forrester Cole and Thomas Funkhouser., "Learning shape templates with structured implicit functions.," *ICCV*, 2019.

[22] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe, "Deep local shapes: Learning local sdf priors for detailed 3d reconstruction," *European Conference on Computer Vision*, 2020.

[23] Abylay Zhumekenov, Malika Uteuliyeva, Olzhas Kabdolov, Rustem Takhanov, Zhenisbek Assylbekov, and Alejandro J. Castro, "Fourier neural networks: A comparative study," *CoRR*, vol. abs/1902.03011, 2019.

[24] 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," *Commun. ACM*, vol. 65, no. 1, pp. 99–106, dec 2021.

[25] Justus Thies Vincent Sitzmann Stephen Lombardi Kalyan Sunkavalli Ricardo Martin-Brualla Tomas Simon Jason Saragih Matthias Nießner et al. Ayush Tewari, Ohad Fried, "State of the art on neural rendering.," *Eurographics*, 2020.

[26] Ruofan Liang, Hongyi Sun, and Nandita Vijaykumar, "Coordx: Accelerating implicit neural representation with a split MLP architecture," *CoRR*, vol. abs/2201.12425, 2022.

[27] Stanford Computer Graphics Laboratory, "Stanford 3d scanning repository.," *http://graphics.stanford.edu/data/3Dscanrep/*, 2014.

[28] William E Lorensen and Harvey E Cline, "Marching cubes: A high resolution 3d surface construction algorithm.," *ACM siggraph computer graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[29] X. Zhang Z. Zhang W. T. Freeman J. Wu, C. Zhang and J. B. Tenenbaum., "Learning shape priors for single-view 3d completion and reconstruction.," *European Conf. on Computer Vision (ECCV)*, 2018.

[30] K. Mo C. R. Qi, H. Su and L. J. Guibas., "Pointnet: Deep learning on point sets for 3d classification and segmentation," *CVPR*, 2017.

[31] H. Su C. R. Qi, L. Yi and L. J. Guibas., "Pointnet++: Deep hierarchical feature learning on point sets in a metric space.," *NIPS*, 2017.

[32] Heli Ben-Hamu, Haggai Maron, Itay Kezurer, Gal Avineri, and Yaron Lipman, "Multi-chart generative surface modeling," *ACM Trans. Graph.*, vol. 37, no. 6, Dec 2018.

[33] Y. Shen Y. Yang, C. Feng and D. Tian., "Foldingnet: Interpretable unsupervised learning on 3d point clouds.," *arXiv preprint arXiv:1712.07262*, 2017.

[34] H. Su H. Fan and L. J. Guibas., "A point set generation network for 3d object reconstruction from a single image," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.