ReLU Fields: The Little Non-linearity That Could

ANIMESH KARNEWAR, University College London, UK
TOBIAS RITSCHEL, University College London, UK
OLIVER WANG, Adobe Research, USA
NILOY J. MITRA, Adobe Research, USA and University College London, UK

In many recent works, multi-layer perceptions (MLPs) have been shown to be suitable for modeling complex spatially-varying functions including images and 3D scenes. Although the MLPs are able to represent complex scenes with unprecedented quality and memory footprint, this expressive power of the MLPs, however, comes at the cost of long training and inference times. On the other hand, bilinear/trilinear interpolation on regular grid-based representations can give fast training and inference times, but cannot match the quality of MLPs without requiring significant additional memory.

Hence, in this work, we investigate what is the smallest change to grid-based representations that allows for retaining the high fidelity result of MLPs while enabling fast reconstruction and rendering times. We introduce a surprisingly simple change that achieves this task – simply allowing a fixed non-linearity (ReLU) on interpolated grid values. When combined with coarse-to-fine optimization, we show that such an approach becomes competitive with the state-of-the-art. We report results on radiance fields, and occupancy fields, and compare against multiple existing alternatives.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2022 Association for Computing Machinery.
0730-0301/2022/7-ART13 $15.00
https://doi.org/10.1145/3528233.3530707

---

1 Introduction

Coordinate-based Multi-layer Perceptrons (MLPs) have been shown to be capable of representing complex signals with high fidelity and a low memory footprint. Exemplar applications include NeRF [Mildenhall et al. 2020], which encodes lighting-baked volumetric radiance-density field into a single MLP using posed images; LIFF [Chen et al. 2020], which encodes 2D image signal into a single MLP using multi-resolution pixel data. Alternatively, a 3D shape can be encoded as an occupancy field [Chen and Zhang 2019; Mescheder et al. 2019], a signed distance field [Park et al. 2019], or directly as a mesh in the form of a surface map [Morreale et al. 2021].

A significant drawback of such approaches is that MLPs are both slow to train and slow to evaluate, especially for applications that require multiple evaluations per signal-sample (e.g., multiple per-pixel evaluations during volume tracing in NeRFs). On the other hand, traditional data structures like n-dimensional grids are fast to optimize and evaluate, but require a significant amount of memory to represent high frequency content (see Figure 4). As a result, there has been an explosion of interest in hybrid representations that combine fast-to-evaluate data structures with coordinate-based MLPs, e.g., by encoding latent features in regular [Sun et al. 2021] and adaptive [Aliev et al. 2020; Liu et al. 2020; Martel et al. 2021;]
In this paper, we revisit regular grid-based models and look for the minimum change needed to make such grids perform on par with “neural” representations. As the key takeaway message, we find that simply using a Rectified Linear Unit (ReLU) non-linearity on top of interpolated grid values, without any additional learned parameters, optimized in a progressive manner already does a surprisingly good job, with minimal added complexity. For example, in Figure 1 we show results in the context of representing volumes (left) and surfaces (right) and on regularly sampled grid vertices for reconstruction, respectively. As additional benefits, these grid based 3D-models are amenable to generative modeling, and to local manipulation.

In summary, we present the following contributions: (i) we propose a minimal extension to grid-based signal representations, which we refer to as ReLU Fields; (ii) we show that this representation is simple, does not require any neural networks, is directly differentiable (and hence easy to optimize), and is fast to optimize and evaluate (i.e. render); and (iii) we empirically validate our claims by showing applications where ReLU Fields plug in naturally: first, image-based 3D scene reconstruction; and second, implicit modeling of 3D geometries.

2 Related Work

Discrete sample based representations Computer vision and graphics have long experimented with different representations for working with visual data. While working with, images are ubiquitously represented as 2D grids of pixels, while due to the memory requirements, 3D models are often represented (and stored) in a sparse format, e.g., as meshes, or as point clouds. In the context of images, since as early as the sixties [Billingsley 1966], different ideas have been proposed to make pixels more expressive. One popular option is to store a fixed number (e.g., one) of zero-crossing for explicit edge boundary information [Bala et al. 2003; Laine and Karras 2010; Ramanarayanan et al. 2004; Tumblin and Choudhury 2004], by using curves [Parilov and Zorin 2008], or augmenting pixels/voxels with more than one color [Agus et al. 2010; Pavić and Kobbelt 2010]. Another idea is to deform the underlying pixel grid by explicitly storing discontinuity information along general curves [Tarini and Cignoni 2005]. Loviscach [2005] optimized MLP maps, such that the thresholded values match a reference. Similar ideas were also being explored for textures and shadow maps [Sen 2004; Sen et al. 2003], addressing specific challenges in sampling. ReLU Field grid implicitly stores discontinuity information by varying grid values such that when interpolated and passed through a ReLU it represents a zero crossing per grid cell.

In the 2D domain, the regular pixel grid format of images has proven to be amenable to machine learning algorithms because CNNs are able to naturally input and output regularly sampled 2D signals as pixel grids. As a result, these architectures can be easily extended to 3D to operate on voxel grids, and therefore can be trained for many learning-based tasks, e.g., using differentiable volume rendering as supervision [Henzler et al. 2019; Nguyen-Phuoc et al. 2019; Sitzmann et al. 2019; Tulsiani et al. 2017]. However, such methods are inefficient with respect to memory and are hence typically restricted to low spatial resolution.

Learned neural representations Recently, coordinate-based MLPs representing continuous signals have been shown to be able to dramatically increase the representation quality of 3D objects [Groueix et al. 2018; Morrane et al. 2021] or reconstruction quality of 3D scenes [Mescheder et al. 2019; Mildenhall et al. 2020]. However, such methods incur a high computational cost, as the MLP has to be evaluated, often multiple times, for each output signal location (e.g., pixel) when performing differentiable volume rendering [Chan et al. 2021b; Mildenhall et al. 2020; Niemeyer and Geiger 2021; Schwarz et al. 2020]. In addition, this representation is not well suited for post-training manipulations as the weights of the MLP have a global effect on the structure of the scene. To fix the slow execution, sometimes grid-like representations are fit post-hoc to a trained Neural Radiance Fields (NeRF) model [Garbin et al. 2021; Hedman et al. 2021; Reiser et al. 2021; Yu et al. 2021b], however such methods are unable to reconstruct scenes from scratch.

As a result, there has been an interest in hybrid methods that store learned features in spatial data structures, and accompany this with
ReLU Fields: The Little Non-linearity That Could

3 Method

It’s Just a Little ReLU

We look for a representation of \( n \)-valued signals on an \( m \)-dimensional coordinate domain \( \mathbb{R}^m \). For simplicity, we explain the method for \( m = 3 \). Our representation is strikingly simple. We consider a regular \((m = 3)\)-dimensional \((r \times r \times r)\)-grid \( G \) composed of \( r \) voxels along each side. Each voxel has a certain size defined by its diagonal norm in the \((m = 3)\)-dimensional space and holds an \( n \)-dimensional vector at each of its \( 2^{m-2} = 8 \) vertices. Importantly, even though they have matching number of dimensions, these values do not have a direct physical interpretation (e.g., color, density, or occupancy), which always have some explicitly-defined range, e.g., \([0, 1] \) or \([0, +\infty)\). Rather, we store unbounded values on the grid; and thus for technical correctness, we call these grids “feature”-grids instead of signal-grids. The features at grid vertices are then interpolated using \((m = 3)\)-linear interpolation, and followed by a single non-linearity: the ReLU, i.e., function \( \text{ReLU}(x) = \max(0, x) \) which maps negative input values to 0 and all other values to themselves. Note that this approach does not have any MLP or other neural-network that interprets the features, instead they are simply clipped before rendering. Intuitively, during optimization, these feature-values at the vertices can go up or down such that the ReLU clipping plane best aligns with the c1-discontinuities within the ground-truth signal. Figure 3 illustrates this concept.

As a didactic example, we fit an image into a 2D ReLU Field grid similar to [Sitzmann et al. 2020], where grid values are stored as floats in the \((-\infty, +\infty)\) range. For any query position, we interpolate the grid values before passing through the ReLU function (see Algorithm 1). Since the image-signal values are expected to be in the \([0, 1]\) range, we apply a hard-upper-clip on the interpolated values just after applying the ReLU. We can see in Fig. 2 that ReLU Field allows us to represent sharp edges at a higher fidelity than bilinear interpolation (without the ReLU) at the same resolution grid size. One limitation of this representation is that it can only well represent signals that have sparse c1-discontinuities, such as this flat-shaded images and as we show later, 3D volumetric density. However, other types of signals, such as natural images, do not

Algorithm 1 Fetching a 2D ReLU Field.

1. procedure \textsc{ReLUField2D}(G, x)
2. \hspace{1em} \( x_g := \text{Floor}(x) \)
3. \hspace{1em} \( x_f := \text{Frac}(x) \)
4. \hspace{1em} \( y_{00} := \text{Fetch}(G, x_g + (0,0)) \)
5. \hspace{1em} \( y_{01} := \text{Fetch}(G, x_g + (0,1)) \)
6. \hspace{1em} \( y_{10} := \text{Fetch}(G, x_g + (1,0)) \)
7. \hspace{1em} \( y_{11} := \text{Fetch}(G, x_g + (1,1)) \)
8. \hspace{1em} \( y := \text{BiLinear}(y_{00}, y_{01}, y_{10}, y_{11}, x_f) \)
9. return \text{ReLU}(y)
10. end procedure
benefit from using a ReLU Fields representation (see supplementary material).

4 Applications

We now demonstrate two different applications of ReLU Fields; NeRF-like 3D scene-reconstruction (Sec. 4.1), and 3D object reconstruction via occupancy fields (Sec. 4.2).

4.1 Radiance Fields

In this application, we discuss how ReLU Field can be used in place of the coordinate-based MLP in NeRF [Mildenhall et al. 2020]. Input to this approach are a set of images \( I = \{I_1, \ldots, I_N\} \) and corresponding camera poses \( C = \{C_1, \ldots, C_N\} \), where each camera pose consists of \( C = (R, T, H, W, F) \); \( R \) is the rotation-matrix \( (R \in \mathbb{R}^{3x3}) \), \( T \) is the translation-vector \( (T \in \mathbb{R}^{3}) \), \( H, W \) are the scalars representing the height and width respectively, and \( F \) denotes the focal length. We assume that the respective poses for the images are known either through hardware calibration or by using structure-from-motion [Schonberger and Frahm 2016].

We denote the rendering operation to convert the 3D scene representation \( S \) and the camera pose \( C \) into an image \( I = R(S, C) \). Thus, given the input set of images \( I \) and their corresponding camera poses \( C \), the problem is to recover the underlying 3D scene representation \( S \) such that when rendered from any \( C_i \in C \), \( S \) produces rendered image \( I_i \) as close as possible to the input image \( I_i \), and produces spatio-temporally consistent \( I_j \) for poses \( C_j \notin C \).

Scene representation We model the underlying 3D scene representation \( S \), which is to be recovered, by a ReLU Field. The vertices of the grid store, first, raw pre-relu density values in \((-\infty, \infty)\) that model geometry, and, second, the second-degree Spherical Harmonics (SH) coefficients [Wizadwongsa et al. 2021; Yu et al. 2021b] that model view-dependent appearance. The relu is only applied to pre-relu density, not to appearance.

We directly optimize values at the vertices to minimize the photometric loss between the rendered images \( \hat{I} \) and the input images \( I \). The optimized grid \( G^* \), corresponding to the recovered 3D scene \( S \), is obtained as:

\[
G^* = \arg \min_{G} \sum_{i=1}^{N} \| I_i - R(G, C_i) \|^2_2. \tag{1}
\]

Implementation details Similar to NeRF, we use the EA (emission-absorption) raymarching model [Henzler et al. 2019; Max 1995; Mildenhall et al. 2020] for realizing the rendering function \( R \). The grid is scaled to a single global AABB (Axis-Aligned-Bounding-Box) that is encompassed by the camera frustums of all the available poses \( C \), and is initialized with uniform random values. We optimize the vertex values using Adam [Kingma and Ba 2014] with a learning rate of 0.03, and all other default values, for all examples shown.

We perform the optimization progressively in a coarse-to-fine manner similar to Karras et al. [2018]. Initially, the feature grid is optimized at a resolution where each dimension is reduced by a factor of 2\(^4\). After a fixed number of iterations at each stage \( N \), the grid resolution is doubled and the features on the feature-grid \( G \) are tri-linearly upsampled to initialize the next stage. This proceeds until the final target resolution is reached.

Evaluation We perform experiments on the eight synthetic Blender scenes used by NeRF [Mildenhall et al. 2020], viz. CHAIR, DRUMS, FICUS, HOTDOG, LEGO, MATERIALS, MIC, and SHIP and compare our method to prior works, baselines, and ablations. We also show an extension of ReLU Fields to one of their real world captured scenes, named FLOWERS.

First, we compare to the mlp-based baseline NeRF [Mildenhall et al. 2020]. For the purpose of these experiments though, we use the public nerf-pytorch version [ner 2021] for comparable training-time comparisons since all our implementations are in PyTorch. For disambiguation, we refer to this PyTorch version as NeRF-PT and the original one as NeRF-TF and report scores for both. Second, we compare to two versions of traditional grids where vertices store scalar density and second-degree SH approximations of the appearance, namely Grid (i.e., \(128^3\) grid) and GridL (i.e., \(256^3\) grid). Finally, we compare to our approach at the same two resolutions, ReLUField and ReLUFieldL. The above four methods are optimized with the same progressive growing setting with \( N = 2000 \), and all the same hyperparameters except the grid resolution. We report PSNR and LPIPS [Zhang et al. 2018] computed on a held-out test-set of Image-Pose pairs different from the training-set \((I, C)\). All training times were recorded on 32GB-V100 GPU while the inference times were computed on RTX 2070 Super. Our method is implemented entirely in PyTorch and does not make use of any custom GPU kernels.

Table 1 summarizes results from these experiments. We can see that traditional physically-based grid baselines Grid and GridL perform the worst, while our method has comparable performance to NeRF-PT and is much faster to reconstruct and render. This retains the utility of grid-based models for real-time applications without compromising on quality. Figure 4 demonstrates qualitative results from these experiments.

Ablations We ablate the components described in 4.1, and add and remove results in Table 1 in the last two columns. RFLong is a normal ReLU Field optimized for a much longer time (comparable to NeRF-PT’s training time). We see minor improvement over the default settings, however we can see that the optimization time plays less of a role than the resolution of the grid itself (ReLUFieldL outperforms ReLUFieldL). ReLUFieldL is trained without progressive growing for the same number of total steps. We see that it yields a much lower reconstruction quality, indicating that progressive growing is critical for the grid to converge to a good reconstruction.

Real scene extension Similar to the real-captured-360 scenes from the NeRF, we also show an extension of ReLU Fields to modeling real scenes. In this example, we model the background using a “MultiSphereGrid” representation, as proposed by Attal et al. [2020]. Please note that the background grid is modeled as a regular bilinear grid without any ReLU. For simplicity, we use an Equi-rectangular projection (ERP) instead of Omni-directional stereo (ODS) for mapping the Image-plane to the set of background spherical shells. Fig. 5 shows qualitative results for this extension after one hour of optimization. Here, we can see that the grid does a good job of representing the complex details in the flower, while the background is modeled reasonably well by the shells.
Fig. 4. Qualitative comparison between NeRF-PT, GridL and ReLUFieldL. Grid-based versions converge much faster, and we can see significant sharpness improvements of ReLUFieldL over GridL, for example in the leaves of the plant. See also supplementary video.
we do not store any SH coefficients on the grid, and simply model

volumetric occupancy as a probability from $[0, 1]$, if the point lies inside the mesh, and 0 otherwise, rather than rendering an image and applying the loss on the rendered image. Finally, since the ground truth occupancy values are binary, we use a binary cross entropy (BCE) loss. Thus, we obtain the optimized grid $G^*$ as,

$$G^* := \arg \min_G \sum_{x \in B} \text{BCE}(O(x), \text{ReLUField3D}(\tanh(G), x))$$ (2)

where, $O$ is the ground truth occupancy, $x$ denote sample locations inside an axis-aligned bounding box $B$, BCE denotes the binary cross entropy loss, and $G$ represents the ReLU Field grid. Note that we use the tanh to limit the grid values in $(-1, 1)$, although other bounding functions, or tensor-normalizations can be used.

**Implementation details** We initialize the grid with uniform random values. The supervision signal comes from sampling random points inside and around the tight AABB of the GT high resolution mesh, and generating the occupancy values for those points by doing an inside-outside test on the fly during training. For rendering, we directly show the depth rendering of the obtained occupancy values. We define the grid-extent and the voxel size by obtaining the AABB directly show the depth rendering of the obtained occupancy values.

Table 2. Evaluation results on modeling 3D geometries as occupancy fields. Metric used is Volumetric-IoU [Mescheder et al. 2019]. The baseline MLP is our implementation of OccupancyNetworks [Mescheder et al. 2019].

<table>
<thead>
<tr>
<th>Scene</th>
<th>NeRF-TF</th>
<th>NeRF-PT</th>
<th>Grid</th>
<th>GridL</th>
<th>RelUField</th>
<th>RelUFieldL</th>
<th>RFLong</th>
<th>RFNoPro</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR</td>
<td>LPIPS</td>
<td>PSNR</td>
<td>LPIPS</td>
<td>PSNR</td>
<td>LPIPS</td>
<td>PSNR</td>
<td>LPIPS</td>
</tr>
<tr>
<td>CHAIR</td>
<td>33.00</td>
<td>0.04</td>
<td>33.75</td>
<td>0.03</td>
<td>25.53</td>
<td>0.12</td>
<td>27.08</td>
<td>0.11</td>
</tr>
<tr>
<td>DRUMS</td>
<td>25.01</td>
<td>0.09</td>
<td>23.82</td>
<td>0.12</td>
<td>19.85</td>
<td>0.20</td>
<td>20.70</td>
<td>0.17</td>
</tr>
<tr>
<td>FICUS</td>
<td>30.13</td>
<td>0.04</td>
<td>28.96</td>
<td>0.04</td>
<td>22.10</td>
<td>0.13</td>
<td>23.61</td>
<td>0.11</td>
</tr>
<tr>
<td>HOTDOG</td>
<td>36.18</td>
<td>0.12</td>
<td>33.52</td>
<td>0.06</td>
<td>28.53</td>
<td>0.12</td>
<td>29.83</td>
<td>0.10</td>
</tr>
<tr>
<td>LEGO</td>
<td>32.54</td>
<td>0.05</td>
<td>28.36</td>
<td>0.08</td>
<td>23.76</td>
<td>0.17</td>
<td>23.97</td>
<td>0.15</td>
</tr>
<tr>
<td>MATERIALS</td>
<td>29.62</td>
<td>0.06</td>
<td>29.23</td>
<td>0.04</td>
<td>21.87</td>
<td>0.18</td>
<td>22.74</td>
<td>0.13</td>
</tr>
<tr>
<td>MIC</td>
<td>32.91</td>
<td>0.02</td>
<td>33.08</td>
<td>0.02</td>
<td>25.87</td>
<td>0.08</td>
<td>25.91</td>
<td>0.08</td>
</tr>
<tr>
<td>SHIP</td>
<td>28.65</td>
<td>0.20</td>
<td>29.22</td>
<td>0.14</td>
<td>23.86</td>
<td>0.25</td>
<td>22.54</td>
<td>0.24</td>
</tr>
</tbody>
</table>

Average volumetric-IoU 0.935 0.903 0.949

Fig. 5. Qualitative results for the real-captured scene extension of ReLU Fields on Flowers. We decompose the scene into a series of spherical-background shells and a foreground ReLU Field layer, which are alpha-composited together to give final novel view renderings. The top-left visualization shows the composite of the background spherical shells un-projected onto a 2D image-plane.

4.2 Occupancy Fields

Another application of coordinate-based MLPs is as a representation of (watertight) 3D geometry. Here, we fit a high resolution ground-truth mesh, as a 3D occupancy field [Mescheder et al. 2019] into a ReLU Field. One might want to do this in order to, for example, take advantage of the volumetric-grid structure to learn priors over geometry, something that is harder to do with meshes or coordinate-based MLPs directly.

**Occupancy representation** The core ReLU Field representation used for this application only differs from the radiance fields setup (see Sec. 4.1) as follows: First, since we are only interested in geometry, we do not store any SH coefficients on the grid, and simply model volumetric occupancy as a probability from $[0, 1]$. Second, as supervision, we use ground truth point-wise occupancy values in 3D (i.e., 1, if the point lies inside the mesh, and 0 otherwise), rather than rendering an image and applying the loss on the rendered image. Finally, since the ground truth occupancy values are binary, we use a BCE loss. Thus, we obtain the optimized grid $G^*$ as,

$$G^* := \arg \min_G \sum_{x \in B} \text{BCE}(O(x), \text{ReLUField3D}(\tanh(G), x))$$ (2)

where, $O$ is the ground truth occupancy, $x$ denote sample locations inside and around the tight AABB of the GT high resolution mesh, and generating the occupancy values for those points by doing an inside-outside test on the fly during training. For rendering, we directly show the depth rendering of the obtained occupancy values. We define the grid-extent and the voxel size by obtaining the AABB ensuring a tight fit around the GT mesh.
Evaluation. Figure 6 shows the qualitative results of the different representations used for this task. We can see that a ReLUField in this case yields higher quality reconstructions than a standard Grid, or a coordinate-based MLP. Quantitative scores, Volumetric-IoU as used in [Mescheder et al. 2019], for the ThaiStatue, Lucy, Bimba, Grog, Lion, Ramses, and Dragon models are summarized in Table 2. ReLUField and Grid require 15 mins, while MLP requires 1.5 hours for training.

5 Discussion
5.1 Limitations
Our approach has some limitations. First, the resulting representations are large. A ReLU Field of size $128^3$ used for radiance fields (i.e., with SH coefficients) takes 260Mb, and the large version at $256^3$ takes 2.0 Gb of storage. We believe that combining ReLU Field with a sparse data structure would see significant gains in performance and reduction in the memory footprint. However, in this work we emphasize the simplicity of our approach and show that the single non-linearity alone is responsible for a surprising degree of quality improvement.

ReLU Field also cannot model more than one “crease” (i.e., discontinuity) per grid cell. While learned features allow for more complex signals to be represented, they do so at the expense of high compute costs. The purpose of this work is to refocus attention on what is actually required for high fidelity scene reconstruction. We believe that the task definition and data are responsible for the high quality results we are seeing now, and show that traditional approaches can yield good results with minor modifications, and neural networks may not be required. However, this is just one data-point in the space of possible representations, for a given specific task we expect that the optimal representation may be a combination of learned features, neural networks, and discrete signal representations.

5.2 Conclusion
In summary, we presented ReLU Field, an almost embarrassingly simple approach for representing signals; storing unbounded data on N-dimensional grid, and applying a single ReLU after linear interpolation. This change can be incorporated at virtually no computational cost or complexity on top of existing grid-based methods, and strictly improve their representational capability. Our approach contains only values at grid vertices which can be directly optimized via gradient descent; does not rely on any learned parameters, special initialization, or neural networks; and performs comparably with state-of-the-art approaches in only a fraction of the time.

Acknowledgments
The authors would like to thank the reviewers for their valuable suggestions. The research was partially supported by the European Union’s Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 956585, gifts from Adobe, and the UCL AI Centre.