Neural Networks for Implicit Representations of 3D Scenes

Luiz Schirmer; Guilherme Schardong; Vinícius da Silva and Hélio Lopes
Pontifícia Universidade Católica do Rio de Janeiro - PUC-Rio

Tiago Novello; Daniel Yukimura; Thales Magalhaes; Hallison Paz and Luiz Velho
Instituto de Matemática Pura e Aplicada - IMPA

***Abstract*—This survey presents methods that use neural networks for implicit representations of 3D geometry — *neural implicit functions*. We explore the different aspects of neural implicit functions for shape modeling and synthesis. We aim to provide a theoretical analysis of 3D shape reconstruction using deep neural networks and introduce a discussion between researchers interested in this research field.**

I. Introduction

Representing shapes as level sets of neural networks has recently been useful for different shape analysis and reconstruction tasks. Computer graphics and 3D computer vision presented multiple approaches to representing 3D geometry for rendering and reconstruction, providing trade-offs across fidelity, efficiency, and compression capabilities. This survey presents methods to model geometry and appearance with Neural Networks that provide extrinsic descriptions, including: i) Spatial Characteristic Functions; ii) Signed Distance Functions (SDFs); and iii) Volumetric Fields. We explore seminal works considering the state-of-the-art high-quality shape representation, interpolation, and completion from partial noisy 3D input data and also image collections. Considering these techniques, implicit data representation can be computed by implicit regularization, periodic activation functions, and loss functions among others explicitly defined over the neural level sets. We also present methods to enhance the performance of these deep neural networks, where it can be possible to be used in real-time applications.

II. Topic Survey

We start giving an overview of the main concepts involved in this topic followed by a classification of the state of the art and recent research work in this area.

A. Main Concepts

The main concepts related to Neural Implicit Representations correspond to the shape and appearance models as well as to the usage of neural networks in these descriptions.

1) Neural Parametric Surfaces: Explicit geometric representations describe a surface intrisically, i.e., by a parametric function $g : U \subset \mathbb{R}^2 \to \mathbb{R}^3$. In that context, a surface can be represented by an atlas, where each chart is a local parameterization [1].

In order to model this kind of represntation by a Neural Network, Groueix et al. [2] and Deprelle et al. [3] suggested modeling charts as Multi-layer perceptrons (MLPs) and Williams et al. [4] focused on individual surface reconstructions. Some works have considered global surface parametrizations [5]–[7]. Global parametrizations produce consistent coverings, although they introduce high distortions [1]. However, parametric representations over implicit neural representations have some disadvantages: it isn't straightforward to produce perfectly overlapping charts.

2) Neural Implicit Surfaces: Implicit geometric representations describe a surface extrinsically, i.e., by a function of the ambient space $f : \mathbb{R}^3 \to \mathbb{R}$ that classifies the points of an embedded object. Differently from explicit geometric representations, such as vertex and triangle lists, implicit descriptions, such as *signed distance functions* (SDFs) may be used to represent geometry, where the object is the zero-level set of a function defined in the space. More formally, in the particular case of SDFs, let $f : \mathbb{R}^3 \to \mathbb{R}$ be a smooth function, f is a signed distance function from $f^{-1}(0)$ if it satisfies the *Eikonal equation* $\|\nabla f\| = 1$. The zero-level set $f^{-1}(0)$ of f is a *hypersurface* in $\mathbb{R}^3$ defined by all points $\mathbf{X} \subset \mathbb{R}^3$ satisfying $f(\mathbf{X}) = 0$. Such representation allows for simple collision tests, continuous (vs. discrete) representation, and differentiability.

B. Model Classification

Recent work in the area of Neural Implicit Representations of 3D Scenes can be classified according to the type of implicit description employed and also by the way it is implemented using a Neural Network.

In that context, we divide the methods in four categories:

- 1st Generation Models
- 2nd Generation Models
- 3rd Generation Models
- 4th Generation Models

It is noteworthy to point out that the evolution of the area followed the developments of these models in chronological order.

1) 1st Generation Models: The First Generation Models correspond to *global* functions of the ambient space and employ as implicit model either the *characteristic function* or the *signed distance function*. They use a fully connected Multi Layer Perceptron (MLP) network architecture. The model is learned by fitting the input data to the model. The Loss function is based either on the $L1$ or $L2$ norm.

The seminal papers of this category appeared in 2019. They are: Occupancy Networks˜ [8], LIF [9], Deep SDF [10], and Deep Level Sets [11].

2) 2nd Generation Models: The Second Generation Models correspond to a set of *local* functions that combined together gives a representation of a function over the whole space. These models are based either on a shape algebra, such as in Constructive Solid Geometry (CSG), or Convolutional Operators.

The seminal papers of this category appeared in 2019 / 2020. They are: LDIF (Genova et al, 2019), BSP-Net [12], CvxNet [13] and Convolutional Occupancy Networks [14].

3) 3rd Generation Models: The Third Generation Models correspond to true signed distance functions (SDF) that are given by the *Eikonal* equation. The model exploits in the Loss function the condition that the gradient of the function has to be constant and with norm equal to one everywhere, i.e., $||\nabla f|| = 1$.

The seminal papers of this category appeared in 2020. They are: IGR [1] and SIREN [15].

4) 4th Generation Models: The Fourth Generation Models correspond to continuous volumetric functions that encode *light fields*. They represent geometry as a density over space and also encode direction-dependent radiance information.

The seminal papers of this category appeared in 2020 /2021. They are: NeRF [16], MNSR [17], among others.

III. First Generation Models

A. Occupancy Networks

Many of prior learning based 3D reconstruction approaches can only represent very coarse 3D geometry or are limited to a restricted domain. Occupancy networks implicitly represent the 3D surface as the continuous decision boundary of a deep neural network classifier. In contrast to previous approaches, that representation encodes a description of the 3D output at infinite resolution without excessive memory footprint.

The main idea is to reason occupancy not only at fixed discrete 3D locations (as in voxel representations) but at every possible 3D point $p \in \mathbb{R}^3$. This is done by defining an occupancy function $o : \mathbb{R}^3 \to \{0, 1\}$ that this network is equivalent to a neural network for binary classification, except that we are interested in the decision boundary which implicitly represents the object's surface. See Figure 1.

When using such a network for 3D reconstruction of an object based on observations of that object (e.g., image, point cloud, etc.), it must be conditioned on the input. Fortunately, a simple functional equivalence can be used for this: a function that takes an observation $x \in X$ as input and has a function from $p \in \mathbb{R}^3$ to $\mathbb{R}$ as output can be equivalently described by a function that takes a pair $(p, x) \in \mathbb{R}^3 \times X$ as input and outputs a real number. The latter representation can be simply parameterized by a neural network f_θ that takes a pair (p, x) as input and outputs a real number which represents the probability of occupancy: $f_\theta : \mathbb{R}^3 \times X \to \{0, 1\}$ (2). This network is called the Occupancy Network.

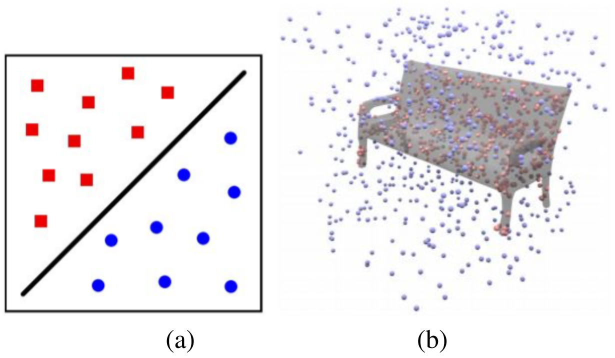

Fig. 1. Occupancy Network as decision Boundary (a) and Classification of Points in Space (bb).

B. Deep SDF

Differently from previous works, [10] proposed a continuous approach for learning 3D shapes. Instead of modelling the shapes explicitly, i.e. by triangle meshes, they proposed using an implicit representation with Signed Distance Functions. The target shape is the zero level-set of this function ($\forall x$ such that $F(x) = 0$, where $F(.)$ is the signed distance function). See Figure 2.

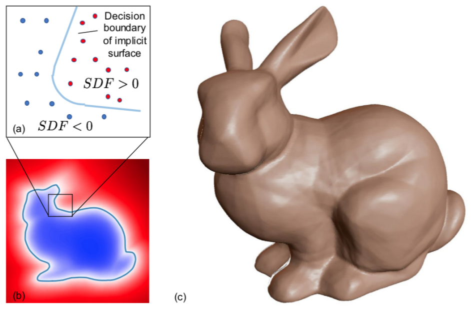

Fig. 2. Deep SDF Implicit Function.

This modelling choice enabled them to make use of the Universal Approximation Theorem for multi-layer perceptrons. Thus, they proposed a MLP model to approximate the SDF representation of a target shape by using points sampled from the domain as input, and their SDF values as targets. A clear advantage of this approach is the hability to learn a continuous representation of the function, being limited only by the model capacity.

Another aspect of DeepSDF is the ability to learn not only a single shape, but a family shapes. By adding a latent vector as an encoder of the target shape, the model is able to map and learn a latent representation of the shape itself. Thus, one may change a latent code while maintaining the point sample as a sort of ”index” to the desired shape. This also allows for shape interpolation by performing a linear interpolation between distinct latent vectors. See Figure 3. The authors propose an encoder-less approach to learn this latent representation, being the first work to do so in the graphics learning community, to the best of our knowledge.

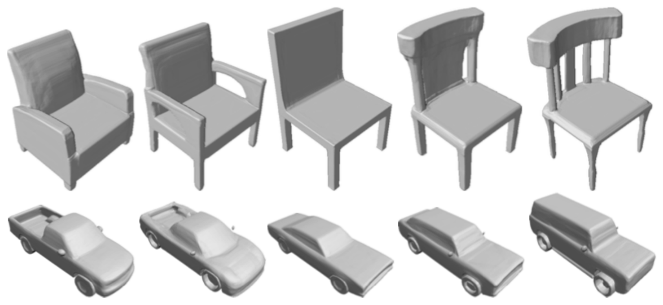
Fig. 3. Interpolation of Shape Families.

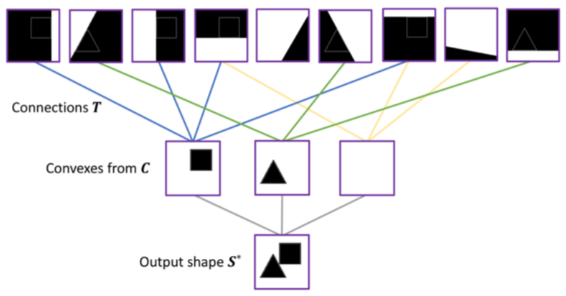

Fig. 4. Neural BSP Tree.

IV. Second Generation Models

A. BSP-Net / CvxNet

Let $\{m_i, g_i\}$ be a given *data-set*. m_i are elements in the *input set* $\mathcal{X}$ (images, point clouds, or voxel grids). $g_i : \mathbb{R}^3 \to \mathbb{R}$ are implicit functions representing objects $\mathcal{O}_i$. Specifically, the object surface $\partial\mathcal{O}_i$ is $\{p \,|\, g_i(p) = 0\}$, its interior is $\{p \,|\, g_i(p) < 0\}$ and $\{p \,|\, g_i(p) > 0\}$ is the exterior. Suppose that $f(\cdot, m_i) = g_i(\cdot)$ for some unknown function $f : \mathbb{R}^3 \times \mathcal{X} \to \mathbb{R}$. The *deep implicit problem* is the task of constructing a function f_θ that approximates f. Once we find a "good" function f_θ, we estimate the object associated with an unknown input $m \in \mathcal{X}$ using $f_\theta(\cdot, m)$. This section presents two networks that solve this problem using *constructive solid geometry* (CSG): CvxNet [13] and BSP-Net [12].

In CSG, *boolean operators* (union, intersection, ...) are used to combine simpler objects to create complex ones. CSG *objects* can be represented by *trees*. The *leaves* correspond to primitives (half-space, cubes, balls, ...) and the *nodes* represent operations. CvxNet and BSP-Net are networks that encode simple CSG trees. The leaves (primitives) are half-spaces, an intermediate layer considers the intersections of the half-spaces to form convex shapes and, finally, the root of the tree is obtained through the union operator that groups the convexes into a single shape.

Given an input $m \in \mathcal{X}$, BSP-Net learns a (CSG) implicit function $f_\theta(\cdot, m) : \mathbb{R}^3 \to \mathbb{R}$. This network consists of three steps. First, an encoder receives the input m and returns a feature code that is the input of an MLP layer. This layer produces the parameters that define a fixed number of plane equations $ax + by + cz + d = 0$. These implicit functions are evaluated on n points (in homogeneous coordinates). The second step considers a binary matrix that forces a collection of half-planes to form a fixed number of convex polytopes. Finally, the last layer groups these convex parts producing the desired implicit function f_θ. Given a new input $m' \in \mathcal{X}$, observe that the zero-level set of $f_\theta(\cdot, m')$ is a surface with *sharp* details since it is the boundary of the union of convex polytopes. Note also that the considered binary matrix learns edges between the half-planes and the convex polytopes, therefore, BSP-Net "learns" the CSG tree. See Figure 4.

The CSG tree is fixed in the CvxNet. However, this network represents a finite family of smooth convex shapes. Specifically, given an input $m \in \mathcal{X}$ an encoder estimates a feature vector λ representing a family of smooth convex shapes. A decoder takes λ and returns a collection of parameter tuples. Each tuple consists of a vector storing the half-space coefficients used to create the corresponding smooth convex. Combining the implicit functions related to these convex shapes gives rise to the desired implicit function f_θ.

Some examples can be seen in Figure 5.

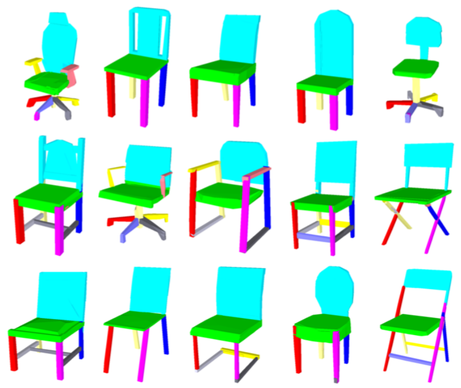
Fig. 5. Correspondence between Decomposition of Structurally Similar Models.

B. Convolution Occupancy Networks

Implicit approaches show exciting results; however, most of the works are limited to analyze simple geometries of single objects. The main related works do not scale to large scenes with higher quality and details. The main factor that may exploit the limitations of these methods is their simple fully-connected network architecture which does not allow for integrating local information in the observations or incorporating inductive biases such as translation equivariance. Peng et al. [18] propose the Convolutional Occupancy Networks, which the author says is a more flexible representation for a detailed reconstruction of objects and 3D scenes.

The main idea is simple but clever: they combine convolutional encoders with implicit occupancy decoders. The method exploits convolutional operations to obtain translation equivariance and the local similarity of 3D structures. They query convolutional features at 3D locations using linear interpolation. In contrast with traditional occupancy networks, this method depends on both input x and the 3D locations. The neural network first processes the input x to obtain a feature encoding for every point or voxel. They use a 3D CNN for voxelized inputs and a shallow PointNet with local pooling for 3D point clouds to construct planar and volumetric feature representations to encapsulate local neighborhoods.

The proposed representation can reconstruct geometry from noisy point clouds and low-resolution voxels. The method scale to large indoor scenes and generalizes from synthetic to real data when compared with Occupancy Networks as we can see in figure 6.

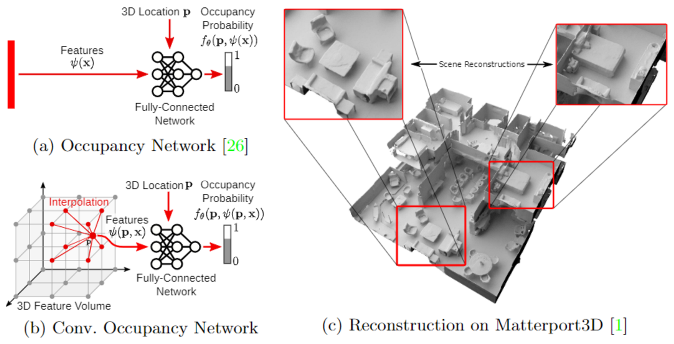
3D Location $\mathbf{p}$
Occupancy Probability
$f_\theta(\mathbf{p}, \psi(\mathbf{x}))$
Features $\psi(\mathbf{x})$
1
0
Fully-Connected Network
(a) Occupancy Network [26]
Interpolation
3D Location $\mathbf{p}$
Occupancy Probability
$f_\theta(\mathbf{p}, \psi(\mathbf{p}, \mathbf{x}))$
Features $\psi(\mathbf{p}, \mathbf{x})$
Fully-Connected Network
3D Feature Volume
(b) Conv. Occupancy Network
Scene Reconstructions
(c) Reconstruction on Matterport3D [1]

Fig. 6. Comparision between Convolutional Occupancy Networks and the traditional one from Peng et al. that shows a reconstruction of a two-floor building from a noisy point cloud on the Matterport3D dataset.

V. Third Generation Models

A. Siren

As we have seen in so far in this survey, implicitly defined, continuous signal representations by neural networks have emerged as a powerful paradigm, offering many possible benefits over conventional representations. However, the network architectures presented in the previous sections for such implicit neural representations are incapable of modeling signals with fine detail, and fail to represent a signal's spatial and temporal derivatives, despite the fact that these are essential to many physical signals defined implicitly as the solution to partial differential equations.

Sitzmann et al. [15] proposes to leverage periodic activation functions for implicit neural representations and demonstrate that these networks are ideally suited for representing complex natural signals and their derivatives. SIRENs can be used for high quality reconstruction of objects and 3D scenes as we can see in Figures 7, 8 and 9.

SIREN [15] — *sinusoidal representation networks* — can be used in order to approximate a sampled implicit function $f : \mathbb{R}^3 \to \mathbb{R}$. SIREN has important properties that are suitable for reconstructing signal, where it has a simple architecture and uses the sine as a periodic activation:

$$f_\theta(p) = W_n \circ f_{n-1} \circ f_{n-2} \circ \cdots \circ f_0(p) + b_n, \qquad (1)$$

$$f_i(p_i) = \sin(W_i \cdot p_i + b_i), \qquad (2)$$

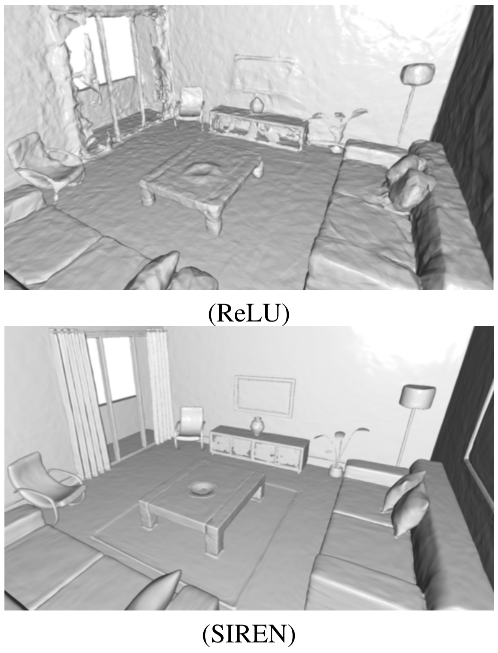

Fig. 7. A SIREN network used for shape representation. The signed distance function is fitted from a point cloud. Compared to a ReLU implicit representation it higher quality for complex 3D scenes.

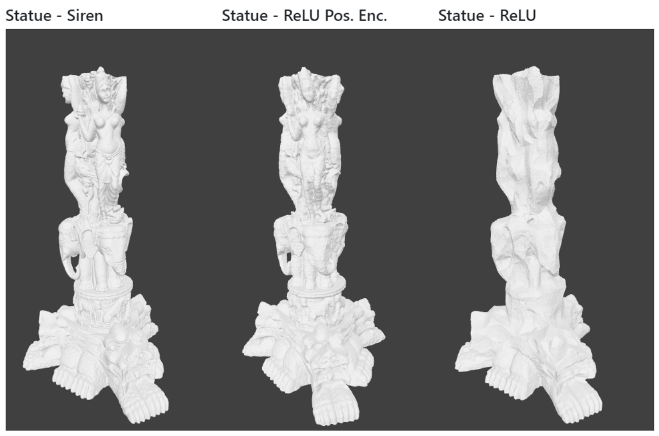

Fig. 8. The SIREN network recovers an SDF from a pointcloud and surface normals by solving the Eikonal equation, a first-order boundary value problem. SIREN can recover a 3D shape given only its pointcloud and surface normals.

where the function $f_i : \mathbb{R}^{N_i} \to \mathbb{R}^{N_{i+1}}$ is the ith layer of the network. This map is obtained by applying the sine to each coordinate of the affine map given by the linear transformation $W_i : \mathbb{R}^{N_i} \to \mathbb{R}^{N_{i+1}}$ translated by $b_i \in \mathbb{R}^{N_{i+1}}$. The linear operators W_i can be represented as matrices and b_i as vectors, therefore, the union of their coefficients correspond to the coefficients θ of the SIREN function f_θ. In other words, f_θ is parameterized by θ.

1) SIREN is smooth: The SIREN function f_θ is smooth since its partial derivatives (of all orders) exist and are continuous. Indeed, each function f_i has all the partial derivatives because, by definition, it is an affine map with the sine applied to each coordinate. Then, the chain rule implies the

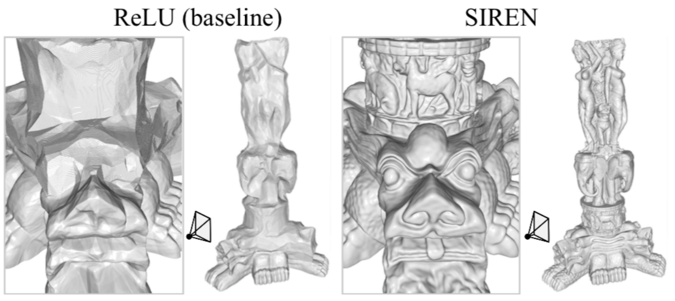

Fig. 9. SIREN significantly improves fine details of objects.

smoothness of f_θ.

We can derive the gradient of f_θ using the chain rule:

$$\begin{aligned}\nabla f_\theta(p) &= \mathbf{J}\Big(W_n \circ f_{n-1} \circ \cdots \circ f_0(p)\Big) + \mathbf{J}(b_n)\\ &= \mathbf{J}W_n(p_n) \circ \mathbf{J}f_{n-1}(p_{n-1}) \circ \cdots \circ \mathbf{J}f_1(p_1) \circ \mathbf{J}f_0(p),\end{aligned}$$

where $\mathbf{J}$ is the *Jacobian* operator and $p_i = f_{i-1} \circ \cdots \circ f_0(p)$. Using the facts that $\mathbf{J}(b_n) = 0$ because b_n is constant, and that $\mathbf{J}(W_n)(q) = W_n$ because W_n is linear, we obtain:

$$\nabla f_\theta(p) = W_n \circ \mathbf{J}f_{n-1}(p_{n-1}) \circ \cdots \circ \mathbf{J}f_1(p_1) \circ Jf_0(p),$$

where $\mathbf{J}f_i(p_i) = W_i \odot \cos\big[a_i\big|\cdots\big|a_i\big]$. The operator $\odot$ is the *Hadamard* product, and the matrix $\big[a_i\big|\cdots\big|a_i\big]$ has N_i copies of the vector $a_i = W_i(p_i) + b_i \in \mathbb{R}^{N_{i+1}}$.

2) SIREN SDF functional: Let $\{p_i, N_i\}$ be a sample of points $\{p_i\}$ and their normals $\{N_i\}$ on a compact surface S embedded in the cube $\mathcal{Q} = [-1,1]^3$, Sitzmann et al. [15] proposed to fit the zero-level set of a SIREN function f_θ to $\{p_i, N_i\}$ forcing f_θ to be a *signed distance function* (SDF). More precisely, they required $f_\theta(p_i) = 0$, $\nabla f_\theta(p_i) = N_i$, and $|\nabla f_\theta(p)| = 1$. The first two equations force the SIREN function f_θ to be zero at the sampled points $\{p_i\}$ and the gradient of f_θ to be aligned to the sampled normals $\{N_i\}$. The equation $|\nabla f_\theta(p)| = 1$ is the *Eikonal* constraint, this is the differential equation for which the solution is a signed distance function. In particular, when we require $f_\theta(p_i) = 0$, it turns out that such restrictions are a sample of the initial condition that we would like to be $f_\theta(p) = 0$ for every $p \in S$.

The *loss function* used in the training [15] of the SIREN function f_θ is very similar to Equation 3:

$$\int_{\mathcal{Q}} |1 - \|\nabla f_\theta(p)\|| \, dp + \int_{S} |f_\theta| + (1 - \nabla f_\theta \cdot N) dS + \int_{\mathcal{Q}/S} e^{\alpha|f_\theta(p)|} dp \quad (3)$$

The term $e^{\alpha|f_\theta(p)|}$ in Equation 3, with $\alpha < -1$, penalizes the points outside the surface S. The function f_θ is supervised using the sample of points $\{p_i, N_i\}$. To train the coefficients θ, the authors used a *minibatch* containing an equal number of points on and off the surface S. The *on-surface* points were uniformly sampled on the point cloud $\{p_i, N_i\}$, and the *off-surface* points were uniformly sampled on the cube $\mathcal{Q}$.

B. IGR - Implicit Geometric Regularization

Implicit representations can be computed using implicit shape representations or loss functions explicitly defined over the neural level sets [19], [20]. The authors of IGR offer a new paradigm for computing high fidelity implicit neural representations directly from raw data in this technique. They observe that a relatively simple loss function, similar to the loss function in SIREN, encourages the neural network to vanish on the input point cloud and to have a unit norm gradient. The authors propose a technique called implicit geometric regularization (IGR) [1]. This method drives the optimization methods to reach a plausible interpretation for the learning and favors smooth and natural zero-level set surfaces. An example of how this method perform shape analysis can be seen in Figure 10.

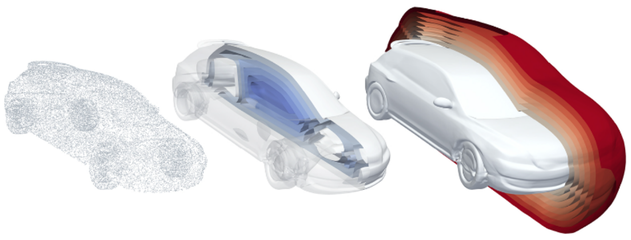

Fig. 10. The level sets of an MLP trained with the IGR method on an input point cloud; positive level sets are in red; negative are in blue; the zero level set, representing the approximated surface, is in white.

VI. Fourth Generation Models

A. NeRF

Neural Radiance Fields (NeRF) [16] generates a novel view from a set of surrounding images and camera poses (Fig.**??**) by representing a scene as a volumetric object and parameterizing it using a neural network. For each 3D point in space and observation direction, the neural network outputs an RGB color and a volume density value. Notice that volume rendering is naturally differentiable and, by using this approach, they were able to optimize the density values as a function of the location and predict the RGB color as a function of both location and viewing direction.

Ray marching through a volume can be a highly costly procedure. To address this issue, the authors propose to optimize two networks simultaneously, so that the output densities of a coarse version could be used to produce more informed sampling of points along the rays for a more refined version. This way, it's possible to do a hierarchical sampling of the scene. This technique demonstrated the capability of representing reflections and specularities when the observation view direction changes, as well as estimating depth information for consistent occlusion tests. See Figure 11.

B. Multiview neural surface reconstruction by disentangling geometry and appearance

In this work, the authors [17] introduce a neural network architecture that simultaneously learns the unknown geometry,

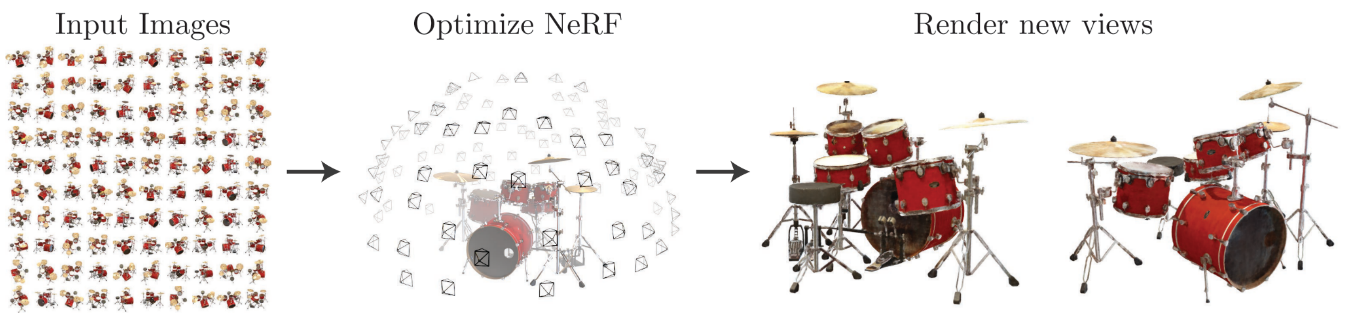

Fig. 11. NeRF rendered views from different images from different angles.

camera parameters, and a neural renderer [21] that approximates the light reflected from a surface towards the camera. The geometry is represented as a zero level-set of a neural network, similarly to the IGR approach. They also derived the rendering from the rendering equation, capable of (implicitly) modeling a broad set of lighting conditions and materials [16]. They trained the network on real-world 2D images of objects with different material properties, lighting conditions, and noisy camera initialization. Figure 12 presents an example of this technique.

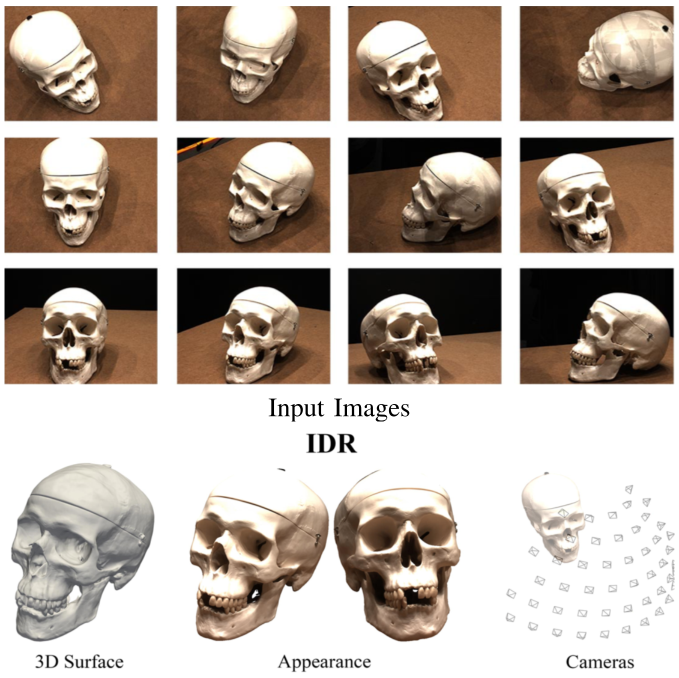

Fig. 12. An end-to-end learning of geometry, appearance and cameras from images.

VII. Network Optimization

Although the expressive results of neural networks in representing 3D surfaces and implicit functions, we fall in performance problems when we consider real-time applications for these methods. In other words, it is unpractical to use these methods in real-time applications due to their computational complexity. In most cases, they fail when considering runtime performance or do not achieve the precision needed. We propose a strategy to improve performance of the SIREN based model considering a trade-off between runtime performance and accuracy. For each layer of SIREN, we use matrix factorizations to not only reduce dimensionality, but also to improve performance. We assume that each layer is over parameterized and its weights can be represented by a matrix or tensor with a lower rank. Actually, the idea of improving the performance of neural networks with matrix factorization is not new. The Faster-RCNN proposes the use of Singular Value Decomposition (SVD) for the fully connected layers.

A fully connected layer essentially does matrix multiplication of its input x by a matrix A, and then adds a bias b:

$$Ax + b \tag{4}$$

We can decompose the matrix A, truncating it, keeping only the first r singular values as in Equation 5:

$$(U_{n\times r} S_{r\times r} V^T_{m\times r})x + b = U_{n\times r}(S_{r\times r} V^T_{m\times r} x) + b \tag{5}$$

Instead of having one fully connected layer, now we have 2 but with smaller weight matrices, where the first one is defined by $S_{r\times r} V^T_{m\times r}$ and the second by the matrix $U_{n\times r}$. The overall number of parameters drops from $n \times m$ to $r(n+m)$.

We know that estimating an optimal rank to approximate the original matrix can be sometimes difficult. We can try different values and check the accuracy, playing with heuristics that consider the trade-off between performance and accuracy. However, the rank selection should be automated. Considering this we can use a technique to estimate an optimal rank approximation for our layers considering the Variational Bayesian Matrix Factorization [22]. The VBMF is a probabilistic algorithm that approximates a matrix $V_{n\times m}$ as the sum of a lower ranking matrices $B_{n\times h} A^T_{h\times m}$ and gaussian noise. After A and B are found, h is an upper bound on the rank. The VB approximation has been successfully applied to matrix factorization, offering automatic dimensionality selection for principal component analysis [23]. Generally, finding the VB solution is a non-convex problem, and most methods rely on a local search algorithm derived through a standard procedure for the VB approximation. Nakajima et.al [22]. presents a global analytical solution for the VBMF, where the global solution is a reweighted SVD of the observed matrix, and each weight can be obtained by solving a quartic equation with its coefficients being functions of the observed singular value.

We test this approach to automatically set an optimal rank to each layer of a Siren Neural Network. With this, we reduce the number of operations for this model by 10 times from the original. With this approach, it was possible to use the model in real-time applications, hlsl shader programs and RTX architecture. Figures 13 and 14 show different results of neural networks with 2 levels of decomposition.

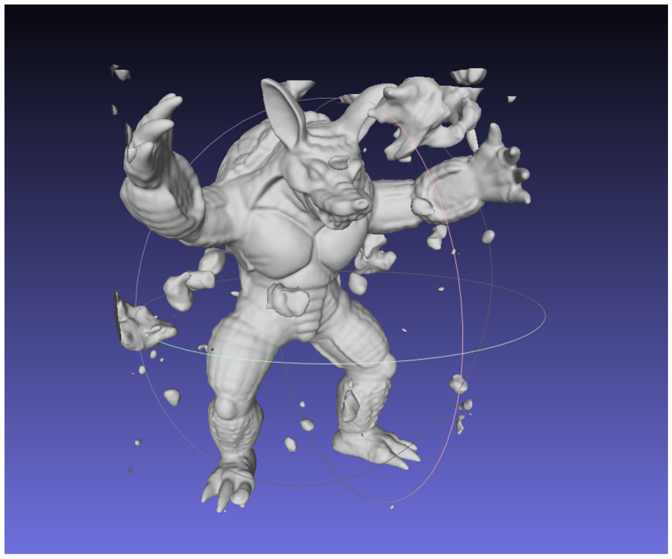

Fig. 13. Result based on a decomposed SIREN with 1/8 of the original parameters size.

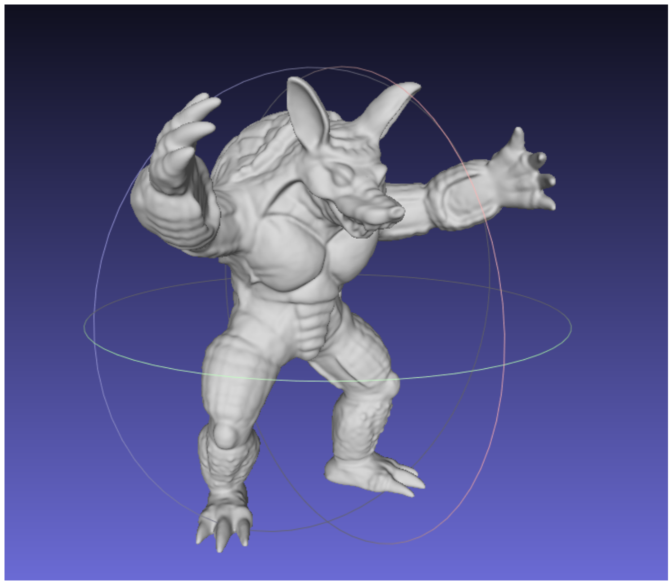

Fig. 14. Result based on a decomposed SIREN with 1/4 of the original parameters size.

VIII. Conclusion

We have presented in this survey the state-of-the-art methods for learning high-fidelity neural implicit representations of 3D shapes. These methods can use different loss functions and activation functions. We described their different architectures and their limitations for real-time applications. At the end, we have proposed strategies to improve the performance and reduce computational complexity.

Due to Neural Implicit Representation's novelty, simplicity and impressive results, the possibilities for future works are fairly vast. There are numerous avenues for exploring its properties, such as sampling approaches to speed-up training convergence, using the approximate SDF values during training to mitigate artifacts in the function domain.

More information about this research area can be found in the companion Web Portal of this survey: "Deep Implicits" (https://lvelho.impa.br/deep-implicits/).

References

[1] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," *arXiv preprint arXiv:2002.10099*, 2020.

[2] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mâché approach to learning 3d surface generation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 216–224.

[3] T. Deprelle, T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "Learning elementary structures for 3d shape generation and matching," *arXiv preprint arXiv:1908.04725*, 2019.

[4] F. Williams, T. Schneider, C. Silva, D. Zorin, J. Bruna, and D. Panozzo, "Deep geometric prior for surface reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 130–10 139.

[5] A. Sinha, J. Bai, and K. Ramani, "Deep learning 3d shape surfaces using geometry images," in *European Conference on Computer Vision*. Springer, 2016, pp. 223–240.

[6] A. Sinha, A. Unmesh, Q. Huang, and K. Ramani, "Surfnet: Generating 3d shape surfaces using deep residual networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 6040–6049.

[7] H. Maron, M. Galun, N. Aigerman, M. Trope, N. Dym, E. Yumer, V. G. Kim, and Y. Lipman, "Convolutional neural networks on surfaces via seamless toric covers." *ACM Trans. Graph.*, vol. 36, no. 4, pp. 71–1, 2017.

[8] L. M. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," *CoRR*, vol. abs/1812.03828, 2018. [Online]. Available: http://arxiv.org/abs/1812.03828

[9] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," 2019.

[10] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 165–174.

[11] M. Michalkiewicz, "Implicit surface representations as layers in neural networks," in *International Conference on Computer Vision (ICCV)*. IEEE, 2019.

[12] Z. Chen, A. Tagliasacchi, and H. Zhang, "Bsp-net: Generating compact meshes via binary space partitioning," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[13] B. Deng, K. Genova, S. Yazdani, S. Bouaziz, G. Hinton, and A. Tagliasacchi, "Cvxnet: Learnable convex decomposition," June 2020.

[14] S. Peng, M. Niemeyer, L. M. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," *CoRR*, vol. abs/2003.04618, 2020. [Online]. Available: https://arxiv.org/abs/2003.04618

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

[16] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *European conference on computer vision*. Springer, 2020, pp. 405–421.

[17] L. Yariv, Y. Kasten, D. Moran, M. Galun, M. Atzmon, R. Basri, and Y. Lipman, "Multiview neural surface reconstruction by disentangling geometry and appearance," *arXiv preprint arXiv:2003.09852*, 2020.

[18] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 2020, pp. 523–540.

[19] M. Atzmon, N. Haim, L. Yariv, O. Israelov, H. Maron, and Y. Lipman, "Controlling neural level sets," *arXiv preprint arXiv:1905.11911*, 2019.

[20] M. Atzmon and Y. Lipman, "Sal: Sign agnostic learning of shapes from raw data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2565–2574.

[21] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 303–312.

[22] S. Nakajima, M. Sugiyama, S. D. Babacan, and R. Tomioka, "Global analytic solution of fully-observed variational bayesian matrix factorization," *Journal of Machine Learning Research*, vol. 14, no. Jan, pp. 1–37, 2013.

[23] S. Nakajima, R. Tomioka, M. Sugiyama, and S. D. Babacan, "Perfect dimensionality recovery by variational bayesian pca." in *NIPS*, 2012, pp. 980–988.