DEEP SEMI-SUPERVISED 3D SHAPE RECONSTRUCTION BY SOLVING A POISSON EQUATION WITH SPECTRAL METHODS

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper we propose a deep learning method for unsupervised 3D implicit shape reconstruction from point clouds. Our goal is to approximate 3D shapes as the iso-surface of a scalar field that is the solution of a Poisson partial differential equation. We propose neural network architecture that learns the distance field in the Fourier domain, and solve the PDE by using spectral differentiation through two novel loss functions. Our experiments show that our architecture can efficiently learn the Fourier coefficients while accurately estimating the target distance field. We train our models without any ground truth mesh, scalar distance field values, or surface normals.

1 INTRODUCTION

In recent years, 3D reconstruction has gained momentum in computer vision and related areas of study. Data-driven 3D reconstruction based on machine learning methods have potentiated the surge of a new variety of generative methods to compute the 3D geometry of shape from different inputs, such as point clouds or 2D images (Achlioptas et al., 2017; Fan et al., 2016; Ranjan et al., 2018; Wang et al., 2018; Saito et al., 2019). While many of these methods have shown significant success to recover full 3D geometries, there are still some challenges. Some of these challenges arise from the choice of representation of the models's output. For instance, generative point cloud networks are lightweight methods to represent 3D geometries. However, point clouds (Fan et al., 2016; Wang et al., 2020; Qi et al., 2016) lack the connectivity structure that triangular meshes have to approximate the surface of a shape. Mesh deformation and mesh prediction approaches (Kanazawa et al., 2018; Wang et al., 2018; Gkioxari et al., 2019; Gupta & Chandraker, 2020) successfully estimate shapes as a piece-wise surface, but struggle to produce smooth manifold meshes. On the other hand, volumetric approaches (Brock et al., 2016; Gadelha et al., 2017; Dai et al., 2017; Ulusoy et al., 2015) are a straightforward way to represent 3D geometries, but they are memory inefficient at high resolutions.

Another important challenge in 3D reconstruction it that most methods rely heavily on full 3D supervision, which is in many cases hard to obtain. Learned-based supervision for 3D reconstruction often comes as ground truth triangular meshes, occupancy voxel grids, or sampled values from a Signed Distance Function (SDF).

A class of 3D reconstruction approaches that has gained popularity in recent years is implicit representation for shape reconstruction (Xu et al., 2019; Saito et al., 2019; Gropp et al., 2020; Mescheder et al., 2019). In an implicit representation, the surface of a shape is estimated as a scalar field over $\mathbb{R}^3$. Continuous spatial coordinates are fed as input features into a neural network which directly produces the values of the implicit functions. Implicit methods are a promising line of research because 1) they are capable of estimating shapes of any topology (number of holes), and 2) they can be sampled at arbitrary resolution to produce high resolution meshes. However, they suffer from the above-mentioned supervision problem. One can argue that in uncontrolled environments, values of the distance to a surface are difficult to collect for training.

In this work, we propose a novel unsupervised deep neural network method to implicitly recover the 3D geometry of a shape from an un-oriented point cloud. Our formulation arises from posing the implicit scalar field as the solution f of a partial differential equation (PDE) with Dirichlet boundary conditions. We let our network learn the Fourier coefficients of the spectral expansion of f, instead

of directly estimating the values of the distance function. We then approach the PDE solution using spectral differentiation Johnson (2011); Canuto et al. (1988) with two novel loss functions. Fig. 2 summarizes our model's main components.

Figure 1: Proposed architecture. Our network takes an input shape in the form of a point cloud, and outputs the spectral coeficients of our PDE-based distance field.

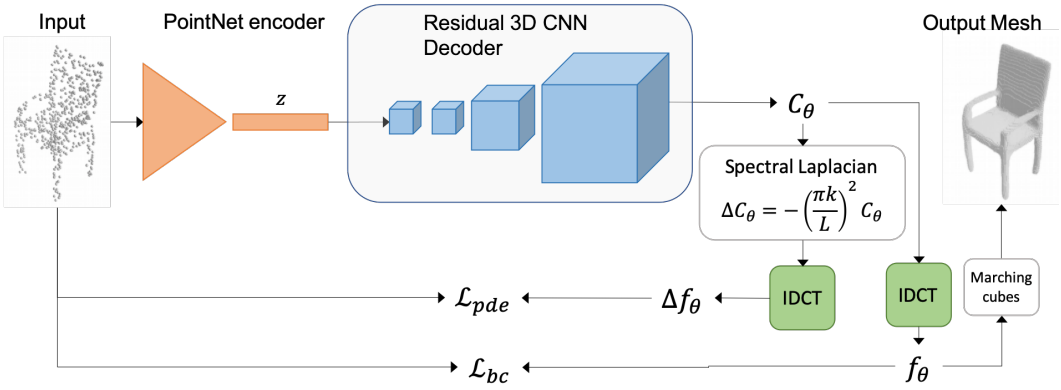


Figure 2: Caption

The contributions of our 3D reconstruction method are three-fold:

- We approach the problem of implicit shape representation by estimating the distance field as the solution of a linear elliptic PDE.
- We learn the distance field on the Fourier domain and solve the PDE by using spectral differentiation.
- Our method effectively reconstructs the surface of 3D shapes without any supervision from ground truth mesh, scalar distance field, or surface normals.

2 Background and Notation

2.1 Deep Implicit 3D Reconstruction

Implicit methods for 3D reconstruction have gained popularity in recent years. They aim to recover the geometry of a target shape Ω as the iso-surface of continuous scalar field $f : \mathbb{R}^3 \rightarrow \mathbb{R}$, such that

$$\partial\Omega = \left\{ \boldsymbol{x} \in \mathbb{R}^3 | f(\boldsymbol{x}) = \tau \right\}. \tag{1}$$

It is usual to set $\tau = 0$ to recover the zero-crossing surface and set f to be the Signed Distance Function (SDF) associated with the surface of Ω.

In general, implicit representations take the form of a deep neural network that approximates the SDF on a subset of the Euclidean space $\mathbb{R}^3$ around the target shape (Genova et al., 2019; Michalkiewicz et al., 2019; Niemeyer et al., 2019; Xu et al., 2019). Other approaches, instead, learn a decision boundary of a binary classifier (Chen & Zhang, 2018; Mescheder et al., 2019; He et al., 2020), indicating whether a point x is inside the surface.

Despite their remarkable reconstruction accuracy, implicit reconstruction methods come with the significant drawback of needing a considerable amount of supervision. They require enough point-distance sample pairs $(\boldsymbol{x}, s) \in \mathbb{R}^3 \times \mathbb{R}$ around the target shape to properly learn the SDF. This amount of supervision is infeasible in many real-world scenarios.

An alternative, to the stated supervision problem comes from the fact that the SDF is known to be a solution a non-linear PDE called the Eikonal equation

$$||\nabla f|| = 1. \tag{2}$$

The scalar field f can be approximated by solving Eq. 2. However, solutions to Eq. 2 are not unique without specifying appropriate Neumann boundary conditions, a.k.a additional supervision in the form of normal vectors at the boundary. This restriction motivates the research for PDE-based alternatives to model scalar distance fields with similar properties, fewer supervision requirements, and potentially easier to solve.

2.2 POISSON EQUATION IN SHAPE ANALYSIS

Poisson equation is one of the simplest and most widely studied second-order linear PDEs. It is defined as

$$\Delta f(\boldsymbol{x}) = \rho(\boldsymbol{x}),\ \forall \boldsymbol{x} \in \Omega \tag{3}$$

where ρ is a known source function, and Δ denotes the Laplace operator. The solution to Eq. 3 can be proven to always be unique and smooth given just Dirichlet boundary conditions, in contrast with the Eikonal equation.

Poisson equation has been used extensively in computer vision, as it arises naturally in many variational problems such as rendering high dynamic range images (Fattal et al., 2002), image matting (Sun et al., 2004), shadow removal (Finlayson et al., 2002), image stitching (Zomet et al., 2006; Agarwala, 2007), image impainting (Elder & Goldberg, 2001; Pérez et al., 2003; Raskar et al., 2004; Agarwala et al., 2004; Jia et al., 2006).

Furthermore, the Poisson equation finds applications in tasks related to shape analysis such as mesh optimization (Nealen et al., 2006), editing and deforming (Sorkine et al., 2004; Yu et al., 2004), and shape representation (Gorelick et al., 2006). It is also the intuition behind one of the most widely used methods for 3D reconstruction, Poisson Surface Reconstruction (PSR) (Kazhdan et al., 2006).

The PSR method reconstructs a shape's surface $\partial\Omega$ using an oriented point cloud $\boldsymbol{X}$. The surface normals at points $\boldsymbol{x} \in \boldsymbol{X}$ are assumed to come from a vector field $\boldsymbol{G} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$. Later, PSR uses $\boldsymbol{G}$ when solving the PDE in Eq. 3 by setting $\rho(\boldsymbol{x}) = \text{div}(\boldsymbol{G}(\boldsymbol{x}))$, where $\text{div}(.)$ is the divergence operator.

Notice that PSR relies on estimations of the surface normals, which means more supervision might not always be available or accurate. Moreover, including surface normals as supervision to solve the PDE, a.k.a Newman boundary conditions, might make the problem over-constrained (Gropp et al., 2020) as the problem need to satisfy to many restriction at once.

2.3 DEEP LEARNING FOR PDE SOLUTIONS

Partial differential equations (PDEs) are among the most ubiquitous tools for modeling physics and engineering problems. Analytic solutions are rarely available for PDEs; thus, computational methods to approximate their solutions are critical for many applications (Beatson et al., 2020).

Given the remarkable success of deep learning, it is not surprising that some authors have started to address the question: Can a neural network approximate the solution of a PDE?. Although solving PDEs with deep learning is an emerging field, there are already some exciting and promising work (Beatson et al., 2020; Berner et al., 2020; Li et al., 2020; Holl et al., 2020; Um et al., 2020; Grohs & Herrmann, 2020). Such approaches come in a wide range of formulations, from solving families of high-dimensional Kolmogorov partial differential equations (Berner et al., 2020), finding discretization invariant solutions to PDEs (Li et al., 2020), to solving them with deep neural architectures that interact with external solvers (Holl et al., 2020; Um et al., 2020).

Despite this progress, some challenges remain. First, many common PDEs include derivatives of a higher order. While most deep learning frameworks allow computation of derivatives with respect to the network's inputs via double back-propagation, this is cumbersome and computationally expensive in practice. Moreover, most neural network architectures use ReLU-based multilayer perceptrons (MLPs). Despite the representation power of ReLU networks, their higher-order derivatives are zero everywhere, making them unsuitable for a wide range of PDEs (Liu et al., 2020; Sitzmann et al., 2020).

3 METHOD

Our goal is to implicitly represent the geometry of 3D shapes as the zero-crossing iso-surface of a scalar field $f : \mathbb{R}^3 \rightarrow \mathbb{R}$. We define define f as the solution of PDE in Eq. 3. We approximate f in the spectral domain by letting a neural network learn its frequency coefficients. Later, we use spectral methods to estimate the target function derivatives. We train our architecture by minimizing two loss functions that enforce f to satisfy the PDE and the boundary condition.

3.1 POISSON-BASED IMPLICIT DISTANCE FUNCTION APPROXIMATION

Consider a bounded domain $\Omega \in \mathbb{R}^3$ with a smooth boundary $\partial\Omega$ given by the surface of a 3D shape. Similar to (Gorelick et al., 2006; Aubert & Aujol, 2014), we approximate the SDF induced by $\partial\Omega$ as the solution of the homogeneous Dirichlet problem for the Poisson equation. To solve the PDE, we use boundary conditions given by f vanishing at all points on $\partial\Omega$, and set the source function to have always negative values:

$$\rho(\boldsymbol{x}) < 0, \ \forall \boldsymbol{x} \in \mathbb{R}^3 \quad \text{and} \quad f(\boldsymbol{x}) = 0, \ \boldsymbol{x} \in \partial\Omega. \tag{4}$$

We need to show that Eq. 3 conditioned with 4 is a suitable alternative to estimate the SDF of given shape. To this purpose, we need f to satisfy two requirements that solutions the Eikonal equation satisfy too. First, f must be zero at all points on $\partial\Omega$. Second, we require f to be positive on all points inside the domain Ω and negative outside, as it happens in the case of an SDF.

Our estimation of the distance field satisfies the first requirement because it is subject to the boundary conditions in 4. The second requirement is essential when recovering the surface as a triangular mesh with an iso-surface extraction algorithm like marching cubes. We need to recall a well-known property of the solution of the Poisson equation to show that f satisfies the second requirement.

Proposition 1 (Minimum principle)**.** *Consider the PDE and the boundary condition in equation 3. If the source function is such that $\rho(\boldsymbol{x}) < 0 \ \forall \boldsymbol{x} \in \mathbb{R}^n$, then the PDE´s solution $f(\boldsymbol{x})$ takes is minimum value somewhere at $\partial\Omega$.*

Proof. Let us assume that f is the solution to Eq. 3 conditioned to 4. Suppose now that f reaches a minimum value $\boldsymbol{x}^*$ given by $M = \min\{f(\boldsymbol{x}) \mid \boldsymbol{x} \in \Omega\}$. This means that $M = f(\boldsymbol{x}^*)$ for some $\boldsymbol{x}^*$ inside Ω. According to the second derivative test, $\boldsymbol{x}^*$ is a minimum point of f if and only if its symmetric Hessian matrix $\boldsymbol{H}(f)(\boldsymbol{x})$ is positive-definite when evaluated at $\boldsymbol{x}^*$.

If $\boldsymbol{H}(f)(\boldsymbol{x}^*)$ is positive definite then it has a Cholesky decomposition $\boldsymbol{H}(f)(\boldsymbol{x}^*) = \boldsymbol{L}\boldsymbol{L}^\intercal$, and thus every element of its main diagonal is of the form $\boldsymbol{H}_{k,k} = \sum_{j=1}^{k} \boldsymbol{L}_{k,j}^2$. This implies that the trace of $\boldsymbol{H}(f)(\boldsymbol{x}^*)$ is also strictly positive and then

$$0 < Tr(\boldsymbol{H}(f)(\boldsymbol{x})) = \Delta f(\boldsymbol{x}) = \rho(\boldsymbol{x}) < 0,$$

which is a contradiction for any point inside Ω. Consequently, f can only reach a minimum value along $\partial\Omega$. □

Note that we refer here to the solution of equation 3 in the weak sense. Therefore we are also interested in what happens outside Ω. Additionally, the opposite sign convention (negative inside / positive outside) is also valid up to a change of sign in $\rho(x)$. The change of sign does not affect the solution because 3 is a linear PDE.

Solutions to the Poisson equation can be guaranteed to be unique using only Dirichlet boundary conditions. In contrast, the Eikonal equation needs a restriction on the gradient of the solution (normal vectors at the surface) to achieve uniqueness. In a learned-based 3D reconstruction setting, requiring the normal vectors at the surface translates into extra supervision at training time. Non-unique solutions can potentially create local minima when optimizing the cost function.

3.2 SPECTRAL METHODS FOR SOLVING POISSON EQUATION

Spectral methods are a class of numerical techniques to solve certain PDEs using the Fourier Transform. They allow us to approximate the solution of Eq. 3 with a trigonometric polynomial that

interpolates f at equally spaced points. Under the assumption of periodicity, one can use the Discrete Fourier Transform (DFT) coefficients to define the interpolating polynomial and expand f through its Inverse Discrete Fourier Transform (IDFT). In the case where f is a function of a single value it takes the form

$$f_n = f\left(\frac{nL}{N}\right) = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \mathcal{F}_k \, e^{\frac{2\pi i}{N} nk}, \tag{5}$$

where, N is the number of samples in the original signal, L is the function's period, and $\mathcal{F}$ is the DFT of f evaluated at the k-th frequency sample. We can approximate f at any continuous point by computing

$$f(x) \approx \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \mathcal{F}_k \, e^{\frac{2\pi i}{L} xk}. \tag{6}$$

Recall that for an absolutely continuous differentiable function, the Laplace operator commutes with the summation in the IDFT. This is possible because the Fourier transform is a linear operation. We can use this result along with Eq. 6 to rewrite the PDE. We then have that

$$\begin{aligned} \Delta f(x) &= \Delta \left(\frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} \mathcal{F}_k \, e^{\frac{2\pi i}{L} xk} \right) = \rho(x) \\ \Delta f(x) &= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} -4\pi^2 k^2 L^{-2} \mathcal{F}_k \, e^{\frac{2\pi i}{L} xk} = \rho(x), \end{aligned} \tag{7}$$

where, we call the factor $-4\pi^2 k^2 L^{-2} \mathcal{F}_k$ the spectral second derivative of f. We assume N to be an odd number, since the above derivation corresponds to the second derivatives only when the Nyquist term vanishes. In the 3-dimensional case, equation 7 holds, correspondingly changing the spectral derivative with the spectral Laplace operator given by $-4\pi^2 ||\boldsymbol{k}||^2 L^{-2} \mathcal{F}_k$, with $\boldsymbol{k} = [k_1, k_2, k_3]$, and $||.||$ the Euclidean norm. See (Johnson, 2011; Canuto et al., 1988) for a more detailed derivation of spectral differentiation.

Let us define f_i as the solution of equation 7 with Dirichlet boundary conditions given by the surface of the i-th element of a 3D collections of shapes $\mathcal{X}$. For each point cloud $\boldsymbol{X}_i \in \mathcal{X}$, we aim to recover its implicit surface by estimating f_i. We approximate the Fourier coefficients of each f_i with a 3D CNN architecture $\mathcal{F}_k(\boldsymbol{z}_i; \theta)$ with parameters θ. $\mathcal{F}_k(\boldsymbol{z}_i; \theta)$ predicts the Fourier Transform coefficients on a regular 3-dimensional grid of length N. We condition $\mathcal{F}_k(\boldsymbol{z}_i; \theta)$ with a latent code z_i generated through a encoder network $\boldsymbol{z}_i = g(\boldsymbol{X}_i; \psi)$ that takes the point cloud X_i as input.

3.3 Source function choice

A straight forward choice for the source function is $\rho(\boldsymbol{x}) = -c^2$, with c a constant. However, this poses a problem: The value of the Laplacian of f at $\boldsymbol{x}$ accounts for how fast the function is decreasing, and that we do not have any guarantees of what happen to f outside Ω. A function that continues decreases could potentially creating discontinuities that could manifest in undesired artifacts in the spectral domain such as Gibbs phenomenon. This happens when the function keep deacresing at a constant rate driven by c. To avoid these artifacts we set

$$\rho(x) = -\frac{1}{|\boldsymbol{X}_i|} \sum_{j=1}^{|\boldsymbol{X}_i|} \exp\left(-\frac{(\boldsymbol{x} - \boldsymbol{x}_j)^\intercal (\boldsymbol{x} - \boldsymbol{x}_j)}{2\sigma^2} \right), \text{ with } \boldsymbol{x} \in \mathbb{R}^3. \tag{8}$$

Note that Eq. 8 is always negative, and its values get closer to zero when $\boldsymbol{x}$ is far from $\partial|\Omega$. Thus, our choice of $\rho(\boldsymbol{x})$ reduces the magnitude of possible discontinuities.

3.4 Implementation details

3.4.1 Learning the coefficients with the Discrete Cosine Transform

Because the DFT is a complex-valued function, one must estimate two values per coefficient in $\mathcal{F}_k$, the real and the imaginary part. Hence, handling complex numbers doubles the memory requirements

to train our model. To alleviate this, we leverage the Discrete Cosine Transform (DCT) as an alternative to learn the coefficients in our spectral method formulation.

The DCT applies only to real-even function. We know that our distance field approximation f is real but most likely not even. However, we can consider an even extension of f, such that:

$$f^e(x) = \begin{cases} f(x) & 0 < x \leq L \\ f(2L - x) & L \leq x \leq 2L. \end{cases}$$

It is easy to check that the Fourier expansion of f^e only includes cosines as basis functions. Additionally, it has been shown that the DCT of the original signal f can be efficiently recovered from an N-point DFT on the odd values of f^e (Makhoul, 1980).

Now, we can exchange the DFT of f with its DCT in Eq. 7 because spectral differentiation is also possible (Ito, 2020) when using the DCT. Our approach to the solution of the PDE becomes

$$\begin{aligned} \Delta \left(\frac{\mathcal{C}_0(z_i;\theta)}{2} + \sum_{k=1}^{N-1} \mathcal{C}_k(z_i;\theta)\ cos\left[\frac{(2x+1)\pi k}{2L}\right] \right) &= \rho(x) \\ \sum_{k=0}^{N-1} -\pi^2 k^2 L^{-2} \mathcal{C}_k(z_i;\theta)\ cos\left[\frac{(2x+1)\pi k}{2L}\right] &= \rho(x), \end{aligned} \tag{9}$$

where $\mathcal{C}_k(z_i;\theta)$ is now the output of our neural network architecture.

3.4.2 Architecture

We model the point cloud encoder $\boldsymbol{z}_i = g(\boldsymbol{X}_i;\psi)$ as a PointNet (Qi et al., 2016) network with *max.* activation functions. The encoder produces a 1024-dimensional latent code $\boldsymbol{z}_i$ from each $\boldsymbol{X}_i$. Later, a 3D CNN decodes z_i into the DCT coefficients, $\mathcal{C}_k$. We model $\mathcal{C}_k$ with a four-block Residual 3D CNN. There is an up-sampling convolution at the end of every block, except for the last convolution, where a linear activation function predicts the DCT values. The network's output is a regular grid of size $64 \times 64 \times 64$ containing the DCT coefficients.

In Fig. 2 we show an overview of the entire pipeline of our proposed method.

3.4.3 Loss functions

Regardless of the dimensions of the $\mathcal{C}_k$ grid, we can estimate f at continuous values of $\boldsymbol{x}$ using Eq. 6. Let us denote by $\Delta\hat{f}(\boldsymbol{x};\boldsymbol{z}_i,\theta)$ and $\hat{f}(\boldsymbol{x};\boldsymbol{z}_i,\theta)$ the predicted Laplacian and the Poisson Distance field after taking the IDCT, respectively. We train our model without any ground-truth distance field or surface normals by defining two loss functions.

Boundary condition loss: Our first loss function intends to enforce the Dirichlet boundary conditions necessary to find a solution of the PDE for each shape $\boldsymbol{X}_i$. We define the boundary condition loss as:

$$\mathcal{L}_{bc} = \frac{1}{|\boldsymbol{X}_i|} \sum_{j=1}^{|\boldsymbol{X}_i|} ||\hat{f}(\boldsymbol{x}_{ij};\boldsymbol{z}_i,\theta) - \tau||_1,\ \forall \boldsymbol{x}_{ij} \in \boldsymbol{X}_i. \tag{10}$$

PDE Loss: We derive the second loss function from Eq. 3. Let $\mathcal{S}_i = \left\{\mathcal{S}_i^{\text{surf}} \cup \mathcal{S}^{\text{rand}}\right\}$ be a set of points in $\mathbb{R}^3$. The set $\mathcal{S}_i^{\text{surf}}$ contains points on the i-th shape's surface, while $\mathcal{S}^{\text{rand}}$ consist of points randomly sampled around each surface point. For each $\boldsymbol{x}_{ij} \in \boldsymbol{X}_i$ we sample points normally distributed with $\mathcal{N}(\boldsymbol{x}_{ij}, 0.05)$. We then define the PDE loss as:

$$\mathcal{L}_{pde} = \frac{1}{|\mathcal{S}_i|} \sum_{j=1}^{|\mathcal{S}_i|} ||\Delta\hat{f}(\boldsymbol{x}_{ij};\boldsymbol{z}_i,\theta) - \rho(\boldsymbol{x}_{ij})||_1,\ \forall \boldsymbol{x}_{ij} \in \mathcal{S}_i \tag{11}$$

The overall loss is a weighted sum of the two losses with weights $\lambda_1 = 1.0$ and $\lambda_2 = 2.0$.

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{pde} + \lambda_2 \mathcal{L}_{bc} \tag{12}$$

Table 1: Poisson distance field results. The table shows the Chamfer distance and the normal consistency for all methods in our study.

Method	CD (↓)			Normal consistency (↑)		
	planes	cars	chairs	planes	cars	chairs
SDF Regress	0.0028	0.0039	0.0092	0.782	0.781	0.783
IGR	0.0060	0.0183	0.0175	0.819	0.756	0.807
OccNET	0.0093	0.0112	0.0302	0.767	0.810	0.765
Ours	0.0846	0.0052	0.0407	0.734	0.784	0.657

4 EXPERIMENTS

Baselines: We test our method against three baselines. We use an occupancy network (OccNet) (Mescheder et al., 2019) to test our model against fully supervised approaches. OccNet assumes ground truth occupancy values on random points around the surfaces of the target shapes. We include in our experiments the Implicit Geometric Regularization approach (IGR) (Gropp et al., 2020) to compare with non-fully supervised methods. Note, however, that the IGR's results largely depend on including the surface's normals into their training pipeline. The surface's normals constitute additional supervision which is not always available. Furthermore, when we omit the surface's normals in IGR, it drops its performance.

Finally, we also compare our proposed method with a baseline of our own by directly estimating a volume with the values of f. Instead of the IDCT, we use a discrete Laplace filter to compute Δf. We use the same loss functions and keep the rest of the pipeline the same. We call this baseline SDF Regress.

Dataset: We use a subset of three classes from ShapeNet v2.0 dataset (Chang et al., 2015) consisting of "planes", "cars", and "chairs". We use 80% of the objects in each category for training, 15% for testing, and 5% for validation.

The meshes in ShapeNet are not guaranteed to be watertight or winding-consistent. Thus, we pre-process them into 2-manifolds using the method defined in (Huang et al., 2018) and remove all the internal meshes with code provided in[1]. Additionally, we center all the meshes at the origin and normalize them to a cube of side 2. Later, we scale the meshes by a factor of 0.85 to guarantee enough padding inside the cube.

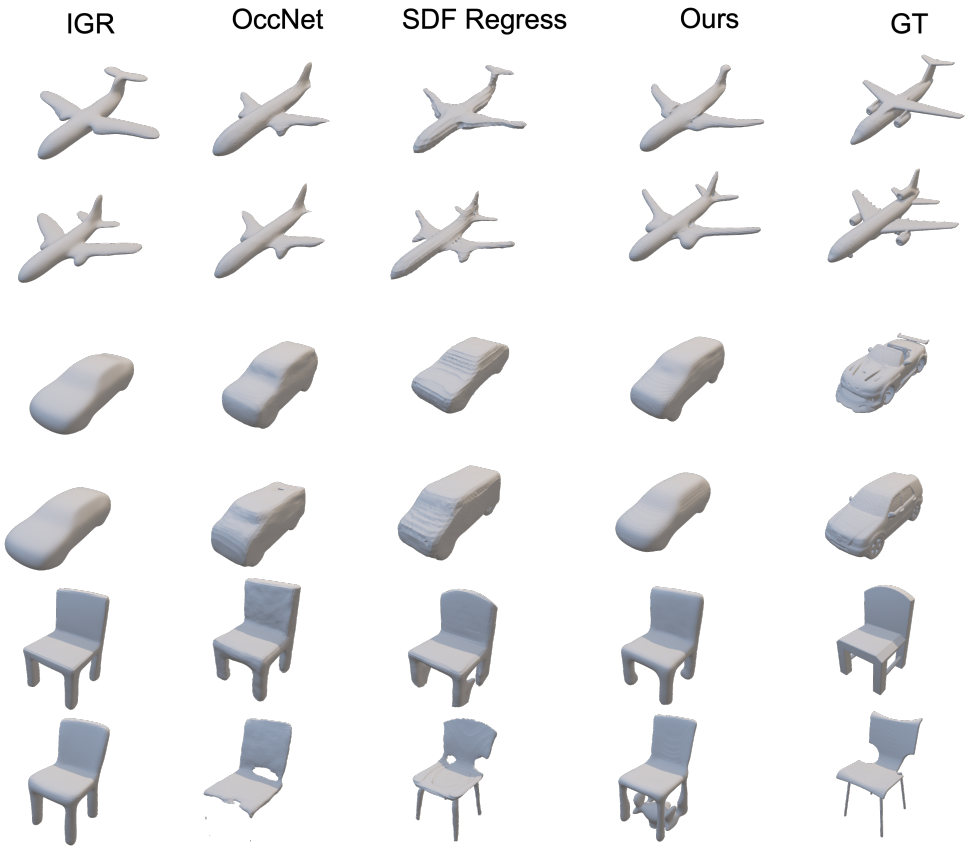
We uniformly sample 4096 surface points on each watertight mesh for all of our experiments. We sample $5\text{e}4$ additional points to be used within the PDE loss. Before feeding the points to our models, we add Gaussian noise with $\mathcal{N}(0, 0.05)$.

Metrics: To evaluate the quality of our 3D reconstructions, we measure the Chamfer-L^2 distance (CD) between our predictions and their ground truth. We use the CD definition from Wang et al. (2018). Additionally, we report the normal consistency score NCS. The NCS is defined as the mean absolute dot product of the surface normals in one mesh and the normals at the corresponding nearest neighbors in the other. We randomly sample $1\text{e}4$ points with its normals on both meshes' surface to estimate the CD and the normal consistency.

Training: Regardless of the category, we train our models for 150.000 steps with a batch size of 22. We train with ADAM optimizer with a base learning rate of $1\text{e}{-5}$. This training strategy ensures at least 35 epochs per category. Because our model is compact, we train each model on a single NVIDIA 2080Ti GPU. The total training time per model is approximately 48 hours. Further speed improvement is possible with multi GPU training. After one-third of the training time, the learning rate decays by a factor of 0.1. We follow the training recipes reported in their respective papers and their official implementation for all the baselines.

[1] https://github.com/tomfunkhouser/gaps

Figure 3: Poisson distance learning results. We show the results of our approach on the first row, against the ground truth meshes in the second row. Notice that our method produces smooth surface from no supervision.



4.1 POISSON DISTANCE FIELD LEARNING

In our experiment, we train our full model using the two loss functions from section 3.4.3. We allow our neural network learn the DCT coefficients that solve our formulation of Poisson equation. Our results show that our architecture can accurately approximate the surface of the models in the dataset from just the input point cloud.

We summarize our results in Table 1 and Figure 3. We report the Chamfer-^{2} loss, and the normal consistency. We compare our results to all baselines. Note that the surface of our predicted meshes are smooth and do not show surface noise like the SDF Regress baseline. Note also that our results look visually comparable to the supervised baselines despite not using ground-truth for the distance field.

5 CONCLUSIONS

We propose a novel semi-supervised method to implicit 3D reconstruction from point clouds. Our method defines a shape's surface as the solution of a Poisson partial differential equation in $\mathbb{R}^3$. We approach the PDE solution using spectral methods through a neural network architecture that learns the target scalar field's spectral coefficients. To the best of our knowledge, this is the only method that has taken this approach. Our experiments show that our model can recover the discrete cosine transform of a function in space from a point cloud and that it can be trained with no distance ground-truth supervision through our PDE y and boundary condition loss functions.

REFERENCES

Panos Achlioptas, O. Diamanti, Ioannis Mitliagkas, and L. Guibas. Representation learning and adversarial generation of 3d point clouds. *ArXiv*, abs/1707.02392, 2017. 1

Aseem Agarwala. Efficient gradient-domain compositing using quadtrees. *ACM Trans. Graph.*, 26(3):94–es, July 2007. ISSN 0730-0301. doi: 10.1145/1276377.1276495. URL https://doi.org/10.1145/1276377.1276495. 3

Aseem Agarwala, Mira Dontcheva, Maneesh Agrawala, Steven Drucker, Alex Colburn, Brian Curless, David Salesin, and Michael Cohen. Interactive digital photomontage. *ACM Trans. Graph.*, 23(3):294–302, August 2004. ISSN 0730-0301. doi: 10.1145/1015706.1015718. URL https://doi.org/10.1145/1015706.1015718. 3

Gilles Aubert and Jean Franois Aujol. Poisson skeleton revisited: A new mathematical perspective. *Journal of Mathematical Imaging and Vision*, 48(1):149–159, 2014. ISSN 09249907. doi: 10.1007/s10851-012-0404-5. 4

Alex Beatson, Jordan Ash, Geoffrey Roeder, Tianju Xue, and Ryan P Adams. Learning composable energy surrogates for pde order reduction. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 338–348. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/0332d694daab22e0e0eaf7a5e88433f9-Paper.pdf. 3

Julius Berner, Markus Dablander, and Philipp Grohs. Numerically solving parametric families of high-dimensional kolmogorov partial differential equations via deep learning. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 16615–16627. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/c1714160652ca640877447381076595O-Paper.pdf. 3

A. Brock, T. Lim, J. Ritchie, and N. Weston. Generative and discriminative voxel modeling with convolutional neural networks. *ArXiv*, abs/1608.04236, 2016. 1

Claudio Canuto, M. Yousuff Hussaini, Alfio Quarteroni, and Thomas A. Zang. *Spectral Methods in Fluid Dynamics*. Springer Berlin Heidelberg, 1988. doi: 10.1007/978-3-642-84108-8. URL https://doi.org/10.1007/978-3-642-84108-8. 2, 5

Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015. URL http://arxiv.org/abs/1512.03012. 7

Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. *CoRR*, abs/1812.02822, 2018. URL http://arxiv.org/abs/1812.02822. 2

Angela Dai, C. R. Qi, and M. Nießner. Shape completion using 3d-encoder-predictor cnns and shape synthesis. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6545–6554, 2017. 1

J. H. Elder and R. M. Goldberg. Image editing in the contour domain. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):291–296, 2001. doi: 10.1109/34.910881. 3

Haoqiang Fan, Hao Su, and Leonidas J. Guibas. A point set generation network for 3d object reconstruction from a single image. *CoRR*, abs/1612.00603, 2016. URL http://arxiv.org/abs/1612.00603. 1

Raanan Fattal, Dani Lischinski, and Michael Werman. Gradient domain high dynamic range compression. *ACM Trans. Graph.*, 21(3):249–256, July 2002. ISSN 0730-0301. doi: 10.1145/566654.566573. URL https://doi.org/10.1145/566654.566573. 3

Graham D. Finlayson, Steven D. Hordley, and Mark S. Drew. Removing shadows from images. In *Proceedings of the 7th European Conference on Computer Vision-Part IV*, ECCV '02, pp. 823–836, Berlin, Heidelberg, 2002. Springer-Verlag. ISBN 3540437487. 3

Matheus Gadelha, Subhransu Maji, and R. Wang. 3d shape induction from 2d views of multiple objects. *2017 International Conference on 3D Vision (3DV)*, pp. 402–411, 2017. 1

Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T. Freeman, and Thomas A. Funkhouser. Learning shape templates with structured implicit functions. *CoRR*, abs/1904.06447, 2019. URL http://arxiv.org/abs/1904.06447. 2

Georgia Gkioxari, Jitendra Malik, and Justin J Johnson. Mesh r-cnn. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9784–9794, 2019. 1

Lena Gorelick, Meirav Galun, and Eitan Sharon. Shape Representation and Classification Using the Poisson Equation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(12): 1991–2005, 2006. 3, 4

P. Grohs and Lukas Herrmann. Deep neural network approximation for high-dimensional elliptic pdes with boundary conditions. *ArXiv*, abs/2007.05384, 2020. 3

Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. In *ICML*, 2020. 1, 3, 7

Kunal Gupta and M. Chandraker. Neural mesh flow: 3d manifold mesh generationvia diffeomorphic flows. In *NeurIPS*, 2020. 1

Tong He, J. Collomosse, H. Jin, and Stefano Soatto. Geo-pifu: Geometry and pixel aligned implicit functions for single-view human reconstruction. *ArXiv*, abs/2006.08072, 2020. 2

Philipp Holl, V. Koltun, and N. Thürey. Learning to control pdes with differentiable physics. *ArXiv*, abs/2001.07457, 2020. 3

Jingwei Huang, Hao Su, and Leonidas Guibas. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698*, 2018. 7

Izumi Ito. A new pseudo-spectral method using the discrete cosine transform. *Journal of Imaging*, 6(4), 2020. ISSN 2313-433X. doi: 10.3390/jimaging6040015. URL https://www.mdpi.com/2313-433X/6/4/15. 6

Jiaya Jia, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum. Drag-and-drop pasting. *ACM Trans. Graph.*, 25(3):631–637, July 2006. ISSN 0730-0301. doi: 10.1145/1141911.1141934. URL https://doi.org/10.1145/1141911.1141934. 3

S. Johnson. Notes on fft-based differentiation. 2011. 2, 5

A. Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 1

Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson Surface Reconstruction. In Alla Sheffer and Konrad Polthier (eds.), *Symposium on Geometry Processing*. The Eurographics Association, 2006. ISBN 3-905673-24-X. doi: 10.2312/SGP/SGP06/061-070. 3

Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Andrew Stuart, Kaushik Bhattacharya, and Anima Anandkumar. Multipole graph neural operator for parametric partial differential equations. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6755–6766. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/4b21cf96d4cf612f239a6c322b10c8fe-Paper.pdf. 3

Ziqi Liu, W. Cai, and Zhi-Qin John Xu. Multi-scale deep neural network (mscalednn) for solving poisson-boltzmann equation in complex domains. *ArXiv*, abs/2007.11207, 2020. 3

John Makhoul. A fast cosine transform in one and two dimensions. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28, 1980. doi: 10.1109/TASSP.1980.1163351. 6

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2, 7

Mateusz Michalkiewicz, Jhony K. Pontes, Dominic Jack, Mahsa Baktashmotlagh, and Anders Eriksson. Implicit surface representations as layers in neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. Laplacian mesh optimization. In *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*, GRAPHITE '06, pp. 381–389, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595935649. doi: 10.1145/1174429.1174494. URL https://doi.org/10.1145/1174429.1174494. 3

Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Occupancy flow: 4d reconstruction by learning particle dynamics. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019. 2

Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM Trans. Graph.*, 22(3):313–318, July 2003. ISSN 0730-0301. doi: 10.1145/882262.882269. URL https://doi.org/10.1145/882262.882269. 3

Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CoRR*, abs/1612.00593, 2016. URL http://arxiv.org/abs/1612.00593. 1, 6

A. Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J. Black. Generating 3d faces using convolutional mesh autoencoders. *ArXiv*, abs/1807.10267, 2018. 1

Ramesh Raskar, Adrian Ilie, and Jingyi Yu. Image fusion for context enhancement and video surrealism. In *Proceedings of the 3rd International Symposium on Non-Photorealistic Animation and Rendering*, NPAR '04, pp. 85–152, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138873. doi: 10.1145/987657.987671. URL https://doi.org/10.1145/987657.987671. 3

Shunsuke Saito, , Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. *arXiv preprint arXiv:1905.05172*, 2019. 1

Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *Proc. NeurIPS*, 2020. 3

O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, SGP '04, pp. 175–184, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 3905673134. doi: 10.1145/1057432.1057456. URL https://doi.org/10.1145/1057432.1057456. 3

Jian Sun, Jiaya Jia, Chi-Keung Tang, and Heung-Yeung Shum. Poisson matting. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pp. 315–321, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 9781450378239. doi: 10.1145/1186562.1015721. URL https://doi.org/10.1145/1186562.1015721. 3

A. O. Ulusoy, A. Geiger, and M. J. Black. Towards probabilistic volumetric reconstruction using ray potentials. In *2015 International Conference on 3D Vision*, pp. 10–18, 2015. doi: 10.1109/3DV.2015.9. 1

Kiwon Um, Robert Brand, Yun (Raymond) Fei, Philipp Holl, and Nils Thuerey. Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6111–6122. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/43e4e6a6f341e00671e123714de019a8-Paper.pdf. 3

H. Wang, Zetian Jiang, Li Yi, Kaichun Mo, H. Su, and L. Guibas. Rethinking sampling in 3d point cloud generative adversarial networks. *ArXiv*, abs/2006.07029, 2020. 1

Nanyang Wang, Y. Zhang, Zhuwen Li, Yanwei Fu, W. Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018. 1, 7

Qiangeng Xu, Weiyue Wang, Duygu Ceylan, Radomir Mech, and Ulrich Neumann. Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 492–502. Curran Associates, Inc., 2019. 1, 2

Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Mesh editing with poisson-based gradient field manipulation. *ACM TRANS. GRAPH*, 23:644–651, 2004. 3

A. Zomet, A. Levin, S. Peleg, and Y. Weiss. Seamless image stitching by minimizing false edges. *IEEE Transactions on Image Processing*, 15(4):969–977, 2006. doi: 10.1109/TIP.2005.863958. 3