Implicit Surface Representations as Layers in Neural Networks

Mateusz Michalkiewicz[2], Jhony K. Pontes[1], Dominic Jack[1], Mahsa Baktashmotlagh[2], Anders Eriksson[2]

[1]School of Electrical Engineering and Computer Science, Queensland University of Technology
[2]School of Information Technology and Electrical Engineering, University of Queensland

Abstract

Implicit shape representations, such as Level Sets, provide a very elegant formulation for performing computations involving curves and surfaces. However, including implicit representations into canonical Neural Network formulations is far from straightforward. This has consequently restricted existing approaches to shape inference, to significantly less effective representations, perhaps most commonly voxels occupancy maps or sparse point clouds.

To overcome this limitation we propose a novel formulation that permits the use of implicit representations of curves and surfaces, of arbitrary topology, as individual layers in Neural Network architectures with end-to-end trainability. Specifically, we propose to represent the output as an oriented level set of a continuous and discretised embedding function. We investigate the benefits of our approach on the task of 3D shape prediction from a single image and demonstrate its ability to produce a more accurate reconstruction compared to voxel-based representations. We further show that our model is flexible and can be applied to a variety of shape inference problems.

1. Introduction

This work concerns the use of implicit surface representations in established learning frameworks. More specifically, we consider how to integrate and treat Level Set representations as singular and individual layers in Neural Networks architectures. In canonical Neural Networks the output of each layer is obtained as the composition of basic function primitives, i.e. matrix multiplication, vector addition and simple non-linear activation functions, applied to its input. By allowing the use of a more expressive surface model, such as Level Sets, our proposed formulation will permit end-to-end trainable architectures capable of inferring richer shapes with much finer details than previously possible, with comparable memory requirements, figure 1.

As a research field, 3D understanding & reconstruction has achieved great progress trying to tackle many categories of problems such as structure from motion [14], multi-view stereo [11] and reconstruction from a single image [5]. The application domain includes, but is not limited to, robotic-assisted surgery, self-driving cars, intelligent robots, and helping visually impaired people to interact with the surrounding world via augmented reality.

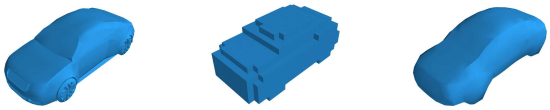
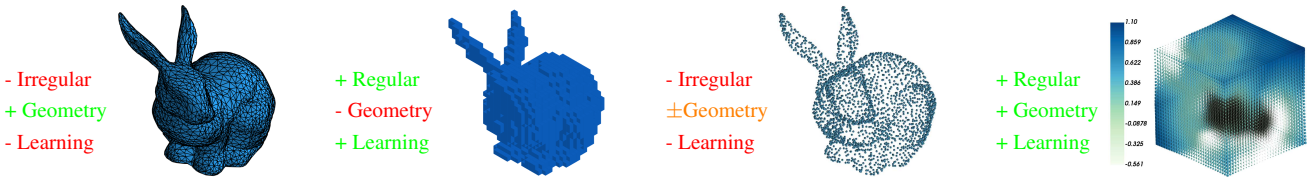


Figure 1. Examples of different 3D shape representations. (left) Ground-truth (polygon mesh), (middle) Voxels occupancy map, (right) Level Set representation. Here the latter two representations are both defined on a discrete Cartesian grid with a resolution of 20^3.

A majority of existing learning-based approaches involving 3D shape or structure are based on voxel occupancy [3, 12, 29, 30], but a considerable amount of attention has also been put on point clouds [10, 28] and explicit shape parameterisation [21]. Each of these representations come with their own advantages and disadvantages, in particular for the application of shape inference in a learning framework, see figure 2. Explicit representations, such as triangle meshes are exceedingly popular in the graphics community as they provide a compact representation able to capture detailed geometry of most 3D objects. However, they are irregular in nature, not uniquely defined, and they cannot be easily integrated into learning frameworks. Voxel occupancy maps on the other hand are defined on fixed regular grids making them exceptionally well suited for learning applications, in particular convolutional approaches. However, unless the resolution of the tessellated grid is high this class of representations typically result in coarse reconstructions. Point clouds are also commonly used to describe the shape of 3D objects. However, this approach suffers from many of the same drawbacks as polygon meshes and is, in addition, only able to provide sparse representations of shapes. In this work we instead argue that implicit representations, or level sets, constitutes a more appropriate choice for the task of learned shape inference. Similar to voxels, level sets are defined on regular grids, making them



(a) Explicit representations (b) Voxels (c) Point cloud (d) Level set

Figure 2. Four common representations of 3D shape along with some of their advantages and disadvantages.

directly suitable for the use with convolutional neural networks. However, this formulation is also more expressive and able to capture more geometrical information of 3D shapes resulting in higher quality inferences. Furthermore, level sets are also equipped with a very refined mathematical formulation that permits the inclusion of additional geometric quantities, such as surface orientation, smoothness and volume, in a very elegant manner [4, 26]. To the best of our knowledge such a direct level set formulation and its geometrical properties have not yet been exploited in previous works.

Convolutional neural networks [9, 12, 16, 22, 29] and generative adversarial models (GANs) [3, 24] have been successfully applied to 3D reconstruction problems by using either volumetric or point cloud representations. The success is mainly due to the availability of large-scale datasets of 3D objects such as Shapenet [2] and ObjectNet3D [34].

All aforementioned approaches require additional step of applying meshing techniques such as SSD or marching cubes to extract the actual 3D mesh. More specifically, one of the main limitation of the existing deep learning approaches for 3D reconstruction is that they are unable to classify pixels lying on the boundary of the object accurately. Thus, the generated boundaries are fuzzy and inaccurate resulting in a coarse and discrete representation of 3-dimensional object. This is specifically due to the fact that a more efficient representations of 3D objects such as polygon mesh do not fit well to deep neural architectures and poses problems in performing back propagation.

In the light of above discussion, we propose to generate a continuous representation of the reconstructed object by integrating level set methods in deep convolutional neural networks. The level set method introduced in [8, 27], and successfully applied in segmentation and medical image analysis [18, 33], is a mathematically elegant way of implicitly representing shape of an object and its boundary evolution in time, which can be represented as a zero level set of an embedding function. To the best of our knowledge, incorporating a level set methods in a deep end-to-end trainable model and representing the 3D output as a level set of a continuous embedding function has never been studied in the literature.

We demonstrate that incorporating the level set representation in an end-to-end trainable network can lead to a more accurate reconstruction. To evaluate this, we used the ShapeNet dataset along with its labeled subset ShapeNet-Core, and compared our approach against three existing voxel-based approaches. We deliberately chose a simple deep architecture which encodes 3-dimensional objects into 64-dimensional vectors and decodes that representation back into the 3-dimensional object. As evidenced in the experiments, our reconstruction is much more accurate than that of using voxel representations, clearly showing that the improvement in representation is due to the level set incorporation, rather than to complex deep architectures. Moreover, representing the output as a level set of a continuous embedding function enables our model to introduce various regularisers, providing further flexibility over classical volumetric methods.

2. Learning Based 3D Shape Inference - Related Work

3D reconstruction is a fundamental problem in computer vision with many potential applications such as robotic manipulation, self-driving cars, and augmented reality. Existing 3D reconstruction methods can be divided into two broad categories: reconstruction from a single image [5], and from multiple images (*e.g.* structure from motion [14]).

One of the important challenges in stepping towards solving this problem is the limited access to the large amount of data required for an accurate reconstruction. Recently, large-scale datasets of 3D objects such as ShapeNet [2] and ObjectNet3D [34] have been made available which allowed the field to make great progress. There have also been attempts on using prior knowledge about the shape of 3D objects [6] in the absence of large amounts of data. Despite its effectiveness, the described approaches relies on hand-crafted features which limits its scalability.

With the advent of deep learning architectures, convolutional neural networks have found to be very useful in 3D reconstruction using only a single image [22]. Recently, [12] and [9] proposed the use of shape and camera features along with the images, respectively. Despite their success, these methods rely on ground truth which is not a realistic scenario.

To tackle this problem, different CNNs-based approaches have been introduced which require only weak su-

pervision [29, 35], and they are able to handle more shape variations. However, they do not scale well when increasing the resolution of the input image. Moreover, more efficient representations of 3D objects like polygon meshes do not easily fit into DNNs architectures.

Recurrent neural networks have recently been proposed to infer 3D shapes. [3] introduced generative adversarial models (GANs) using long short-term memory (LSTM) for reconstructing voxels or point clouds achieving state-of-the-art results. [29] proposed the use of conditional GANs in an unsupervised setting and [32] proposed the use of octrees. An important drawback of GAN-based methods is that they are computationally expensive and not accurate when using metrics such as the Chamfer distance, Earth Mover's distance or intersection over union (IoU). Another drawback of such methods is that they do not allow multiple reconstruction which is sometimes needed when dealing with single image reconstruction. As a response to these shortcomings Delaunay Tetrahydration or voxel block hashing [24] were introduced.

Even though great progress has been achieved in the 3D reconstruction field, the aforementioned approaches suffer from the lack of geometry due to its poor shape representation. In this paper, we propose the use of a continuous 3D shape representation by integrating level sets into CNNs. Our aim is to infer embedding functions to represent the geometry of a 3D shape where we can then extract its level set to have a continuous shape representation, *i.e*. a 3D surface.

3. Preliminaries

Level Set Surface Representations. The Level Set method for representing moving interfaces was proposed independently by [27] and [8]. This method defines a time dependent orientable surface $\Gamma(t)$ implicitly as the zero isocontour, or level set, of a higher dimensional auxiliary scalar function, called the *level set function* or *embedding function*, $\phi(x,t) : \Omega \times \mathbb{R} \mapsto \mathbb{R}$, as,

$$\Gamma(t) = \{x : \phi(x,t) = 0\}, \tag{1}$$

with the convention that $\phi(x,t)$ is positive on the interior and negative on the exterior of Γ. The underlying idea of the level set method is then to capture the motion of the isosurface through the manipulation of the level set function ϕ.

Given a surface velocity v, the evolution of the isosurface Γ is particularly simple, it is obtained as the solution of the partial differential equation (PDE) (known as the *level set equation*)

$$\frac{\partial \phi}{\partial t} = v|\nabla \phi|. \tag{2}$$

In practice, this problem is discretised and numerical computations are performed on a fixed Cartesian grid in some domain. This formulation also permits a natural way to calculate additional interface primitives, *i.e*. surface normals, curvatures and volumes. Such primitives are typically used in applications involving entities with physical meanings, to impose specific variabilities of the obtained solution, for instance to favour smoothness of the surface Γ.

One additional advantage of the level set formulation is that it allows complex topologies as well as changes in topology in a very elegant and simple manner without the need for explicit modelling. This is typically not the case in most parametric approaches, where topological variations needs to be handled explicitly through dedicated procedures.

Minimal Oriented Surface Models. Here we formulate the task of fitting an implicitly defined closed surface Γ to a given oriented surface $\mathcal{S} \subset \mathbb{R}^3$ as that of simultaneously minimising the distance to a discrete number of points $x_i \in \mathcal{S}$ as well as the difference between the orientation of the unit-length surface normals n_i (at x_i) and the normals of Γ. Note that $\mathcal{S}$ does not necessarily have to be a closed surface, hence the orientation of the normals n_i are not uniquely defined and only determined up to a sign ambiguity (*i.e*. $n_i \sim \pm n_i$). Let $\mathcal{S}$ be given as a collection of m data points of $\mathcal{X} = \{x_i\}_{i=1}^m$ and their corresponding normals $\mathcal{N} = \{n_i\}_{i=1}^m$, and let $d_{\mathcal{X}}(x)$ denote as the distance function to $\mathcal{X}$,

$$d(x, \mathcal{X}) = \inf_{y \in \mathcal{X}} \|x - y\|. \tag{3}$$

As in [37], we then define the following energy functional for the variational formulation,

$$E_{\mathcal{X}}(\Gamma) = \left(\int_\Gamma d(s, \mathcal{X})^p ds \right)^{1/p}, 1 \leq p \leq \infty. \tag{4}$$

The above functional measures the deviation as the L_p-norm of the distance from the surface Γ from the point set $\mathcal{X}$.

Similarly, for the normals $\mathcal{N}$ we define an energy functional that quantifies the difference between the normal of the estimated surface Γ and the desired surface normals of the given surface $\mathcal{S}$. The measure we propose is the L_p-norm of the angular distance between the normals of Γ and those of $\mathcal{N}$.

$$E_{\mathcal{N}}(\Gamma) = \left(\int_\Gamma (1 - |N(s) \cdot n_\Gamma(s)|)^p \, ds \right)^{1/p}, 1 \leq p \leq \infty, \tag{5}$$

where $N(s) = n_i$ when x_i is the closest point to s. With the outward unit normal of Γ given by

$$n_\Gamma(s) = \frac{\nabla \phi(s)}{\|\nabla \phi(s)\|}, \tag{6}$$

we can write $E_{\mathcal{N}}(\Gamma)$ as

$$E_{\mathcal{N}}(\Gamma) = \left(\int_{\Gamma} \left(1 - \left| N(s) \cdot \frac{\nabla \phi(s)}{\|\nabla \phi(s)\|} \right| \right)^{p} ds \right)^{1/p}. \quad (7)$$

Note that since both (5) and (7) are defined as surface integral over Γ they will return decreased energies on surfaces with smaller area. Consequently, both these energy functionals contain an implicit smoothing component due to this predilection towards reduced surface areas.

Shape Priors & Regularisation. Attempting to impose prior knowledge on shapes can be a very useful proposition in a wide range of applications. A distinction is typically made between generic (or geometric) priors and object specific priors. The former concerns geometric quantities, generic to all shapes, such as *surface area*, *volume* or *surface smoothness*. In the latter case, the priors are computed from set of given samples of a specific object of interest. Formulations for incorporating such priors in to the level set framework has been the topic of considerable research efforts, for an excellent review see [4].

For the sake of simplicity and brevity, in this section we limit ourselves to two of the most fundamental generic shape priors, *surface area* and *volume*. They are defined as,

$$E_{area} = \int_{\Gamma} ds, \quad (8) \qquad E_{vol} = \int_{\text{int}\Gamma} ds. \quad (9)$$

However, many of the additional shape priors available can presumably be directly incorporated in to our proposed framework as well.

Embedding functions and Ill-Conditioning. It has been observed that in its conventional formulation the level set function often develop complications related to ill-conditioning during the evolution process, [13]. These complications may in turn lead to numerical issues and result in an unstable surface motion. Many of these conditioning issues are related to degraded level set functions, ones that are either too steep or too flat near its zero level set. A class of functions that do not display these properties are the *signed distance functions*. They are defined as

$$f(x) = \pm \inf_{y \in \Gamma} \|x - y\|, \quad (10)$$

where $f(x)$ is > 0 if x is in the interior of Γ and negative otherwise. Signed distance functions have unit gradient, $|\nabla f| = 1$, not only in the proximity to Γ by its entire domain. Consequently, a common approach to overcoming these stability issues is to regularly correct or *reinitialise* the level set function to be the signed distance function of the current zero level set isosurface.

However, in our intended setting of shape inference in a learning framework, such a reinitialisation procedure is not directly applicable. Instead we propose the use of an energy functional, similar to the work of [20], that promotes the unit gradient property,

$$E_{sdf}(\phi) = \int (\|\nabla \phi(x)\| - 1)^2 dx. \quad (11)$$

4. Implicitly Defined Neural Network Layers

In this section we show how an implicit representation of 3D surfaces (or more explicitly the isosurface operator) can be introduced as a distinct layer in neural network architecture through a direct application of the variational formulations of the previous section. We begin by defining the loss function and the structure of the forward pass of our proposed formulation.

Given a set of n training examples I^j and their corresponding ground truth oriented shapes $\mathcal{S}^j = \{\mathcal{X}^j, \mathcal{N}^j\}$, here represented as a collection of discrete points with associated normals, see section 3. Let θ denote the parameters of some predictive procedure, a neural network, that from an input I estimates shape implicitly through a level set function, $\tilde{\phi}(I;\theta)$. At training, we then seek to minimise (with respect to θ) the dissimilarity (measured by a loss function) between the training data and the predictions made by our network. The general variational loss function we propose in this work is as follows,

$$\begin{aligned} L(\theta) = & \sum_{j \in \mathcal{D}} E_{\mathcal{X}^j}(\tilde{\Gamma}(I^j;\theta)) + \alpha_1 \sum_{j \in \mathcal{D}} E_{\mathcal{N}^j}(\tilde{\Gamma}(I^j;\theta)) \\ & + \alpha_2 \sum_{j \in \mathcal{D}} E_{sdf}(\tilde{\phi}(I^j;\theta)) + \alpha_3 \sum_{j \in \mathcal{D}} E_{area}(\tilde{\Gamma}(I^j;\theta)) \\ & + \alpha_4 \sum_{j \in \mathcal{D}} E_{vol}(\tilde{\Gamma}(I^j;\theta)). \end{aligned} \quad (12)$$

Here $\tilde{\Gamma}$ denotes the zero level set of the predicted level set function $\tilde{\phi}$ given input I, that is $\tilde{\Gamma}(I;\theta) = \{x : \tilde{\phi}(I;\theta) = 0\}$, $\mathcal{D} = \{1, ..., n\}$ and $\alpha_1 - \alpha_4$ are weighting parameters.

By introducing the Dirac delta function δ and the Heaviside function H we can write the individual components of (12) as,

$$\begin{aligned} \sum_{j \in \mathcal{D}} E_{\mathcal{X}^j}(\tilde{\Gamma}(I^j;\theta)) = & \\ = \sum_{j \in \mathcal{D}} \left(\int_{\mathbb{R}^3} \delta(\tilde{\phi}(x, I^j;\theta)) d(x, \mathcal{X}^j)^p dx \right)^{1/p}, & \end{aligned} \quad (13)$$

$$\begin{aligned} \sum_{j \in \mathcal{D}} E_{\mathcal{N}^j}(\tilde{\Gamma}(I^j;\theta)) = \sum_{j \in \mathcal{D}} \Bigg(& \int_{\mathbb{R}^3} \delta(\tilde{\phi}(x, I^j;\theta)) \\ & \left(1 - \left| N^j(x) \cdot \frac{\nabla \tilde{\phi}(x; I^j, \theta)}{\|\nabla \tilde{\phi}(x, I^j;\theta)\|} \right| \right)^p dx \Bigg)^{1/p}, \end{aligned} \quad (14)$$

$$\sum_{j \in \mathcal{D}} E_{sdf}(\tilde{\phi}(I^j;\theta)) = \sum_{j \in \mathcal{D}} \int_{\mathbb{R}^3} (\|\nabla \tilde{\phi}(x, I^j;\theta)\| - 1)^2 dx, \quad (15)$$

$$\sum_{j\in\mathcal{D}} E_{area}(\tilde{\Gamma}(\theta, I^j)) = \sum_{j\in\mathcal{D}} \int_{\mathbb{R}^3} \delta(\tilde{\phi}(x, I^j; \theta))\, dx, \quad (16)$$

$$\sum_{j\in\mathcal{D}} E_{vol}(\tilde{\Gamma}(\theta, I^j)) = \sum_{j\in\mathcal{D}} \int_{\mathbb{R}^3} H(\tilde{\phi}(x, I^j; \theta))\, dx. \quad (17)$$

In practise the above loss function is only evaluated on a fixed equidistant grid Ω in the volume of interest. It is then also necessary to introduce continuous approximations of the Dirac delta function and Heaviside function, Following the work of [36] we use the following C^1 and C^2 approximations of δ and H respectively,

$$\delta_\epsilon(x) = \begin{cases} \frac{1}{2\epsilon}\left(1 + \cos(\frac{\pi x}{\epsilon})\right), & |x| \leq \epsilon, \\ 0, & |x| > \epsilon, \end{cases} \quad (18)$$

and

$$H_\epsilon(x) = \begin{cases} \frac{1}{2}\left(1 + \frac{x}{\epsilon} + \frac{1}{\pi}\sin(\frac{\pi x}{\epsilon})\right), & |x| \leq \epsilon, \\ 1, & x > \epsilon, \\ 0, & x < -\epsilon, \end{cases} \quad (19)$$

note that here $H'_\epsilon(x) = \delta_\epsilon(x)$. Inserting (18)-(19) in (13)-(17) we obtain an approximated loss function L_ϵ expressed entirely in $\tilde{\phi}$. With the simplified notation $\tilde{\phi}^j(x) = \tilde{\phi}(x, I^j; \theta))$ and $d^j(x)^p = d(x, \mathcal{X}^j)^p$, we arrive at,

$$\begin{aligned} L_\epsilon(\theta) = &\sum_{j\in\mathcal{D}} \left(\sum_{x\in\Omega} \delta_\epsilon(\tilde{\phi}^j(x)) d^j(x)^p \right)^{1/p} \\ &+ \alpha_1 \sum_{j\in\mathcal{D}} \left(\sum_{x\in\Omega} \delta_\epsilon(\tilde{\phi}^j(x)) \left(1 - \left| N^j(x) \cdot \frac{\nabla\tilde{\phi}^j(x)}{\|\nabla\tilde{\phi}^j(x)\|} \right|\right)^p \right)^{1/p} \\ &+ \alpha_2 \sum_{j\in\mathcal{D}} \sum_{x\in\Omega} (\|\nabla\tilde{\phi}^j(x)\| - 1)^2 + \alpha_3 \sum_{j\in\mathcal{D}} \sum_{x\in\Omega} \delta_\epsilon(\tilde{\phi}^j(x)) \\ &+ \alpha_4 \sum_{j\in\mathcal{D}} \sum_{x\in\Omega} H_\epsilon(\tilde{\phi}^j(x)). \end{aligned} \quad (20)$$

To form the backward pass of a neural network we require the gradient of each individual layer with respect to the output of the previous layer as well as for the resulting loss function. With the above derivation, it proves convenient to calculate the gradient of the isosurface operator and the loss function jointly. That is, we differentiate L_ϵ with respect to $\tilde{\phi}$ on the discrete grid Ω, yielding

$$\begin{aligned} \frac{\partial L_\epsilon}{\partial\tilde{\phi}} = &\sum_{j\in\mathcal{D}} \frac{1}{p} \left(\sum_{x\in\Omega} \delta_\epsilon(\tilde{\phi}^j(x)) d^j(x)^p \right)^{\frac{1-p}{p}} \delta'_\epsilon(\tilde{\phi}^j(x)) d^j(x)^p \\ &+ \frac{\alpha_1}{p} \sum_{j\in\mathcal{D}} \left(\sum_{x\in\Omega} \delta_\epsilon(\tilde{\phi}^j(x)) \left(1 - \left| \frac{N^j(x)\cdot\nabla\tilde{\phi}^j(x)}{\|\nabla\tilde{\phi}^j(x)\|} \right|\right)^p \right)^{\frac{1-p}{p}} \\ &\left(\delta'_\epsilon(\tilde{\phi}^j(x)) \left(1 - \left| \frac{N^j(x)\cdot\nabla\tilde{\phi}^j(x)}{\|\nabla\tilde{\phi}^j(x)\|} \right|\right)^p + \right. \\ &\left. \delta_\epsilon(\tilde{\phi}^j(x)) \frac{\partial}{\partial\tilde{\phi}} \left(1 - \left| \frac{N^j(x)\cdot\nabla\tilde{\phi}^j(x)}{\|\nabla\tilde{\phi}^j(x)\|} \right|\right)^p \right) \\ &+ \alpha_2 \sum_{j\in\mathcal{D}} \sum_{x\in\Omega} (\|\nabla\tilde{\phi}^j(x)\| - 1)\, \nabla\cdot \left(\frac{\nabla\tilde{\phi}^j(x)}{\|\nabla\tilde{\phi}^j(x)\|} \right) \\ &+ \alpha_3 \sum_{j\in\mathcal{D}} \delta'_\epsilon(\tilde{\phi}^j(x)) + \alpha_4 \sum_{j\in\mathcal{D}} \delta_\epsilon(\tilde{\phi}^j(x)). \end{aligned} \quad (21)$$

Obtaining the shape for a given level set function $\tilde{\phi}$, is straightforward, only requiring the isosurface $\tilde{\Gamma}$ to be extracted from $\tilde{\phi}$. This can be done using any of a number of existing algorithms, see [15]. Note that, as a consequence, our proposed framework is entirely agnostic to the choice of isosurface extraction algorithm. This is an important distinction from work such as [21] which is derived from a very specific choice of algorithm.

5. Experimental Validation

In this section we present our empirical evaluation of the proposed formulation applied to the task of 3D shape inference from single 2D images. These experiments were primarily directed at investigating the potential improvement obtained by an implicit representation over that of more conventional representation. This paper was not intended to study the suitability of different types of networks for the task of shape inference. In fact, as discussed further down in this section, we deliberately chose a rather simple network architecture to conduct this study on.

5.1. Implementation Details

We begin by discussing some of the practical aspects of the experimental setup we used in this section.

Dataset & Preprocessing. We evaluated the proposed formulation on data from the ShapeNet dataset [2]. We chose a subset of 5 categories from this dataset: *'bottles'*, *'cars'*, *'chairs'*, *'sofas'* and *'phones'*. As ShapeNet models often do not have an empty interior, we used the manifold surface generation method of [17] as a preprocessing stage to generate closed manifolds of those models and used them as ground truth.

We ended up with approximately 500 models for *'bottles'* and 2000 models each for the remaining categories. Each model is rendered into 20 2D views, (input images) using fixed elevation, and equally spaced azimuth angles. This data was then randomly divided into 80/20 train-test splits. The ground-truth manifolds are also converted to a voxel occupancy map, for training and testing the voxel-based loss functions, using the procedure of [25].

Network Architecture. Motivated by [12], we use a simple 3D auto-encoder network which predicts 3D representation from 2D rendered image, and consists of two components: an auto-encoder as a generator and a CNN as

Table 1. Performance comparison between voxel occupancy and level set representations on test data with two different resolutions, 20^3 and 32^3, measured by IoU (in %). Here Δ denotes the difference in IoU.

IoU [%]	20^3			32^3				
Category	voxels	ϕ	Δ	R2N2	Matr	voxels	ϕ	Δ
Bottle	65.7	**78.4**	(+12.7)	64.0	76.3	71.9	**78.4**	(+6.5)
Car	73.0	**86.6**	(+13.6)	78.5	80.8	81.9	**86.0**	(+4.1)
Chair	57.2	**63.7**	(+6.5)	58.0	61.0	59.2	**61.9**	(+2.7)
Sofa	62.1	**68.9**	(+6.8)	64.3	68.7	68.0	**72.9**	(+4.9)
Telephone	60.5	**73.5**	(+13.0)	63.3	63.2	67.8	**71.9**	(+4.1)

Table 2. Performance comparison between voxel occupancy and level set representations on test data with two different resolutions, 20^3 and 32^3, measured by the Chamfer distance. Here Δ denotes the difference in Chamfer distance.

Chamfer	20^3			32^3				
Category	voxels	ϕ	Δ	R2N2	Matr	voxels	ϕ	Δ
Bottle	0.0895	**0.0593**	(-0.0302)	0.0877	0.0564	0.0669	**0.0520**	(-0.0149)
Car	0.0917	**0.0411**	(-0.0506)	0.0675	0.0517	0.0623	**0.0430**	(-0.0193)
Chair	0.1003	**0.0885**	(-0.0118)	0.0975	**0.0768**	0.0829	0.0899	(+0.0070)
Sofa	0.0936	**0.0649**	(-0.0287)	0.0887	0.0648	0.0709	**0.0595**	(-0.0114)
Telephone	0.0963	**0.0510**	(-0.0453)	0.0799	0.0709	0.0658	**0.0530**	(-0.0128)

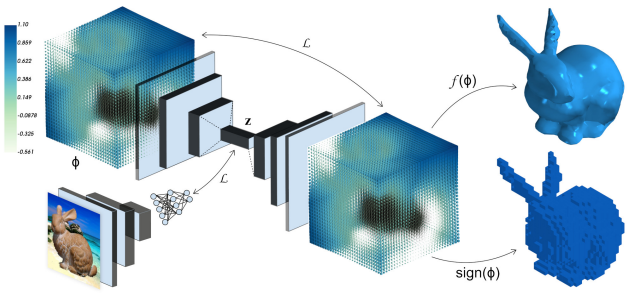


Figure 3. An overview of the network architecture used.

a predictor connected by a 64-dimensional vector embedding space.

Specifically, the autoencoder network with convolution and deconvolution layers, projects a 3D shape to the 64-dimensional space, and decodes it back to a 3D shape. The encoder composed of four convolutional layers and a fully connected layer to project the data to the 64D embedding vector. The decoder consists of five 3D convolutional layers with stride 1 connected by batch norm and ReLU nonlinearities to map the embedding vector back to 3D space.

Similar to MobileNetV2 [31] architecture, the CNN comprised of five convolutional layers, two fully connected layers, and an added 64 dimensional layer, initialised with the ImageNet [7] weights, and projects a 2D image to the 64 dimensional space.

The two components are jointly optimised at training time, taking the 3D CAD model along with its 2D image. At test time, the encoder part of the auto-encoder is removed, and then the ConvNet and the decoder are used to obtain a 3D representation and images in the shared latent space.

Note that the reason behind our choice of such a simple architecture is to demonstrate that the improvement in the representation is due to our representation of shape, rather than adopting complex deep architectures.

Comparison. We compare the proposed implicit shape representation to that of a voxel-based occupancy map representation. This is done by training the network using two distinct loss functions: the variational loss function, defined in section 4 (with $p = 2$, $\epsilon = 0.15$, $\alpha_1 = 0.8$, $\alpha_2 = 1$ and $\alpha_3 = \alpha_4 = 0.1$)[1], and the voxel-based cross-entropy loss defined in [12], Both formulations were trained with 2D images as inputs, for 3000 epochs using a batch size of 64 and a learning rate of 10^{-6}, the ground-truth shapes are represented as manifolds and voxel occupancy maps respectively. We observed the training times and convergence behaviour of the two methods to be comparable. It is im-

[1] A more thorough ablation study of the effect of these parameters is in progress but is considered out of scope for this work.

portant to note that the architecture is identical in both instances, consequently so is the memory requirements and computational costs at evaluation. The only difference is that the ground truth for the voxel-based approach is binary as opposed to real-valued (a polygon mesh) for the variational formulation. For comparison we also include results from two additional existing methods, R2N2 [3] and Matryoschka networks [30], code provided by the authors.

It is important to note that our proposed formulation should not be viewed as a competitor to existing algorithms but rather as a complement. The use of implicit shape representations could readily be incorporated into many of the existing approaches currently in use (such as R2N2 and Matryoschka networks).

Evaluation Metrics. Here we considered two separate metrics for evaluating shapes inferred by our network, the *Intersection over Union* (IoU), also known as the *Jaccard Index* [19], and the Chamfer distance [1].

The IoU measures similarity between two finite sample sets, A and B. It is defined as the ratio between the size of their intersection and union,

$$\text{IoU}(A, B) = \frac{|A \cap B|}{|A \cup B|}. \tag{22}$$

This metric ranges between 0 and 1, with small values indicating low degrees of similarity and large values indicating high degrees of similarity.

The Chamfer distance is a symmetric metric, here used to measure distance between a pair of surfaces, or more accurately between two finite point clouds ($\mathcal{P}_1$ and $\mathcal{P}_2$) sampled from a pair of surfaces.

$$d_{\text{ch}}(\mathcal{P}_1, \mathcal{P}_2) = \sum_{x \in \mathcal{P}_1} \min_{y \in \mathcal{P}_2} ||x - y|| + \sum_{y \in \mathcal{P}_2} \min_{x \in \mathcal{P}_1} ||y - x||. \tag{23}$$

Note that these two measures are defined on distinctly different domains, sets and point-clouds/surfaces for IoU and Chamfer distance respectively. As we are comparing different shape representations, the ground truth and level set representations are available as surfaces and the voxel occupancies as sets, we need to be attentive to how these measures are applied.

The IoU is measured by first applying a voxelization procedure [25] to the ground truth as well as the level sets inferred by the trained network. To ensure that all the finer details of these surfaces are preserved, this voxelization should be performed at a high resolution.[2] Correspondingly, surface representations can be extracted from occupancy grids by means of the Marching Cubes algorithm [23].

[2]We empirically observed that going beyond a resolution of 128^3 did not impact the results noticeably.

5.2. Results

We evaluated the efficiency of our proposed variational-based implicit shape representation both quantitatively and qualitatively. Quantitatively, we measured the similarity/distance between the ground-truth and the inferred shapes using both a voxel and a level set representation. The results are shown in tables 1 and 2. The R2N2 and Matryoschka networks were only evaluated on 32^3 resolution as the publicly available implementations for these two methods did not readily permit a 20^3 resolution.

These results appear very promising and supports our argument that in learning frameworks implicit shape representations are able to express shape more accurately and with more detail than voxel based representations can. We can further see that, as expected, the difference between these two representations is reduced as the resolution increases. Voxels does appear to perform on par with, or even slightly better than level sets in one instance, on the class 'chair'. We believe this can be explained by the topology of that specific class. Many chairs typically have very long thin legs, structures that are difficult to capture at a low resolution, both for voxels and level sets. The area and volume regularisation terms E_{area} and E_{vol} might also explain this reduction in performance, as they inherently discourage long thin structures.

Examples of the qualitative results are shown in figure 4 and it clearly demonstrates the higher quality of the 3D shapes inferred by our proposed approach over those obtained from volumetric representations.[3]

6. Conclusion and Future Work

We proposed a novel and flexible approach for 3D shape inference by incorporating an implicit surface representation into a deep end-to-end neural network architecture. We showed that level set functions in its conventional formulation can become ill-conditioned during the evolution process, resulting in numerical and surface motion instabilities. To alleviate this, we made use of an energy functional to promote the unit gradient property as a regulariser in our learning model. Our experiments demonstrated the ability of our approach to infer accurate 3D shapes with more geometrical details compared to voxel-based representations. In future work, we plan to investigate the the flexibility of our approach to accommodate higher resolution shape inference and segmentation problems; as well as ways of incorporating the proposed implicitly defined layers in graph convolutional neural networks.

Acknowledgements. This work has been funded by the Australian Research Council through grant FT170100072.

[3]These examples were chosen to be representative of the typical predictions generated by the different networks.

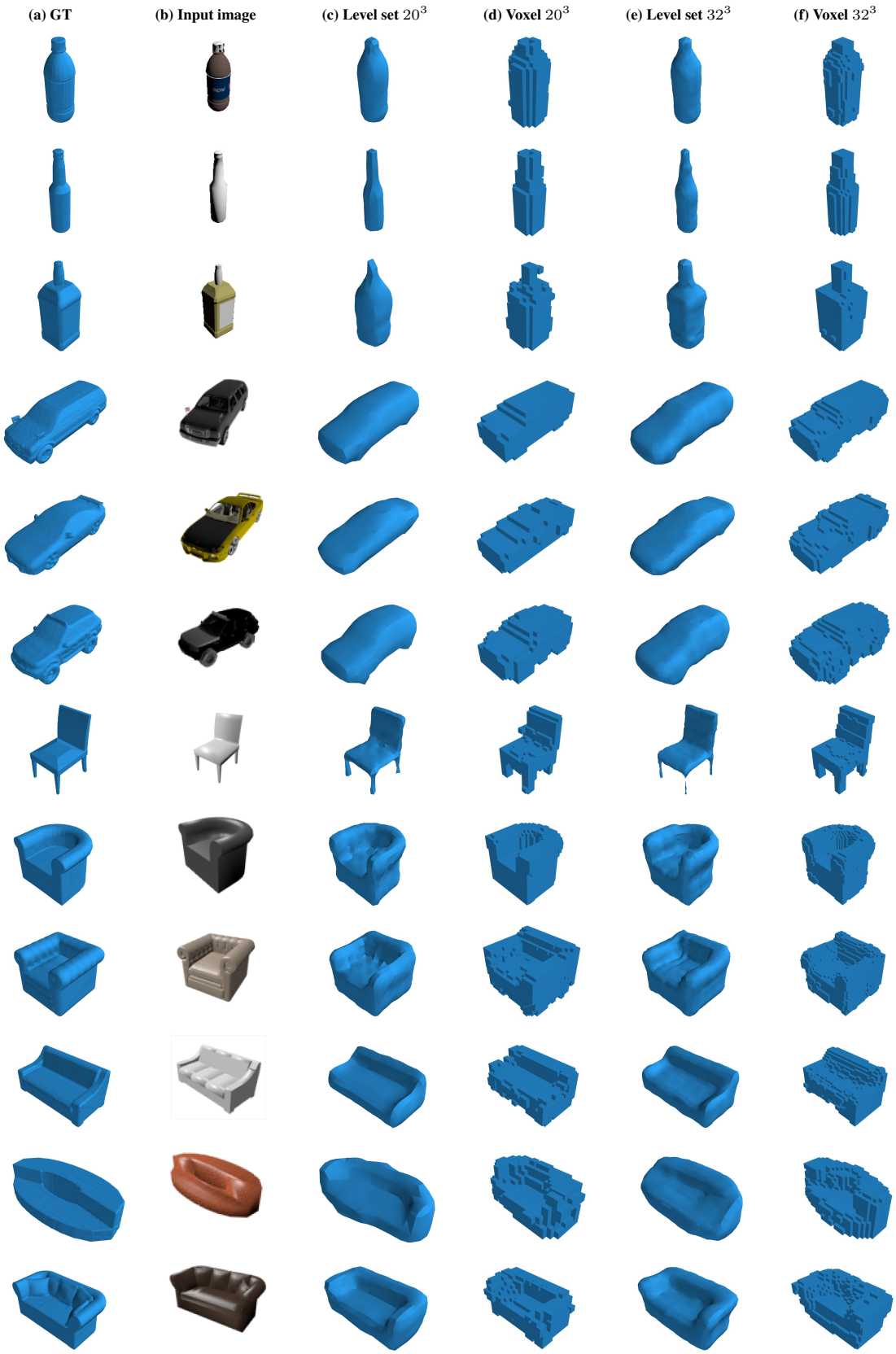


Figure 4. 3D shape inference from a single 2D image. The columns are (a) ground-truth shape, (b) input image, (c) predicted shape, level set, 20^3, (d) predicted shape, voxels, 20^3, (e) predicted shape, level set, 32^3, (f) predicted shape, voxels, 32^3.

References

[1] Harry G Barrow, Jay M Tenenbaum, Robert C Bolles, and Helen C Wolf. Parametric correspondence and chamfer matching: Two new techniques for image matching. Technical report, SRI International, 1977. 7

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2, 5

[3] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 1, 2, 3, 7

[4] Daniel Cremers, Mikael Rousson, and Rachid Deriche. A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape. *International journal of computer vision*, 72(2):195–215, 2007. 2, 4

[5] Antonio Criminisi, Ian Reid, and Andrew Zisserman. Single view metrology. *International Journal of Computer Vision*, 40(2):123–148, 2000. 1, 2

[6] Amaury Dame, Victor A Prisacariu, Carl Y Ren, and Ian Reid. Dense reconstruction using 3d object shape priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1288–1295, 2013. 2

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, pages 248–255. IEEE, 2009. 6

[8] Alain Dervieux and François Thomasset. A finite element method for the simulation of a rayleigh-taylor instability. In *Approximation methods for Navier-Stokes problems*, pages 145–158. Springer, 1980. 2, 3

[9] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. Learning to generate chairs with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1538–1546, 2015. 2

[10] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, volume 2, page 6, 2017. 1

[11] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, pages 1434–1441. IEEE, 2010. 1

[12] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016. 1, 2, 5, 6

[13] José Gomes and Olivier Faugeras. Reconciling distance functions and level sets. *Journal of Visual Communication and Image Representation*, 11(2):209–223, 2000. 4

[14] Klaus Häming and Gabriele Peters. The structure-from-motion reconstruction pipeline–a survey with focus on short image sequences. *Kybernetika*, 46(5):926–937, 2010. 1, 2

[15] Charles D Hansen and Chris R Johnson. *Visualization handbook*. Elsevier, 2011. 5

[16] Ping Hu, Bing Shuai, Jun Liu, and Gang Wang. Deep level sets for salient object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, volume 1, page 2, 2017. 2

[17] Jingwei Huang, Hao Su, and Leonidas Guibas. Robust watertight manifold surface generation method for shapenet models. *arXiv preprint arXiv:1802.01698*, 2018. 5

[18] Satyanad Kichenassamy, Arun Kumar, Peter Olver, Allen Tannenbaum, and Anthony Yezzi. Gradient flows and geometric active contour models. In *ICCV*, pages 810–815. IEEE, 1995. 2

[19] Michael Levandowsky and David Winter. Distance between sets. *Nature*, 234(5323):34, 1971. 7

[20] C. Li, C. Xu, C. Gui, and M. D. Fox. Distance regularized level set evolution and its application to image segmentation. *IEEE Transactions on Image Processing*, 19(12):3243–3254, Dec 2010. 4

[21] Yiyi Liao, Simon Donné, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2916–2925, 2018. 1, 5

[22] Fayao Liu, Chunhua Shen, and Guosheng Lin. Deep convolutional neural fields for depth estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5162–5170, 2015. 2

[23] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM siggraph computer graphics*, volume 21, pages 163–169. ACM, 1987. 7

[24] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics (ToG)*, 32(6):169, 2013. 2, 3

[25] Fakir S. Nooruddin and Greg Turk. Simplification and repair of polygonal models using volumetric techniques. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):191–205, 2003. 5, 7

[26] Stanley Osher and Nikos Paragios. *Geometric Level Set Methods in Imaging,Vision,and Graphics*. Springer-Verlag, Berlin, Heidelberg, 2003. 2

[27] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *Journal of computational physics*, 79(1):12–49, 1988. 2, 3

[28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.*, 1(2):4, 2017. 1

[29] Danilo Jimenez Rezende, SM Ali Eslami, Shakir Mohamed, Peter Battaglia, Max Jaderberg, and Nicolas Heess. Unsupervised learning of 3d structure from images. In *Advances in Neural Information Processing Systems*, pages 4996–5004, 2016. 1, 2, 3

[30] Stephan R Richter and Stefan Roth. Matryoshka networks: Predicting 3d geometry via nested shape layers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1936–1944, 2018. 1, 7

[31] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 6

[32] Frank Steinbrücker, Jürgen Sturm, and Daniel Cremers. Volumetric 3d mapping in real-time on a cpu. In *ICRA*, pages 2021–2028, 2014. 3

[33] Ross T Whitaker. A level-set approach to 3d reconstruction from range data. *International journal of computer vision*, 29(3):203–231, 1998. 2

[34] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3d object recognition. In *European Conference on Computer Vision*, pages 160–176. Springer, 2016. 2

[35] Xinchen Yan, Jimei Yang, Ersin Yumer, Yijie Guo, and Honglak Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *Advances in Neural Information Processing Systems*, pages 1696–1704, 2016. 3

[36] Hong-Kai Zhao, Tony Chan, Barry Merriman, and Stanley Osher. A variational level set approach to multiphase motion. *Journal of computational physics*, 127(1):179–195, 1996. 5

[37] Hong-Kai Zhao, Stanley Osher, Barry Merriman, and Myungjoo Kang. Implicit and nonparametric shape reconstruction from unorganized data using a variational level set method. *Computer Vision and Image Understanding*, 80(3):295–314, 2000. 3