Neural Vector Fields for Surface Representation and Inference

Edoardo Mello Rella[1], Ajad Chhatkuli[1], Ender Konukoglu[1], and Luc Van Gool[1,2]

[1] Computer Vision Lab, ETH Zurich, Switzerland
[2] VISICS, ESAT/PSI, KU Leuven, Belgium

Abstract. Neural implicit fields have recently been shown to represent 3D shapes accurately, opening up various applications in 3D shape analysis. Up to now, such implicit fields for 3D representation are scalar, encoding the signed distance or binary volume occupancy and more recently the unsigned distance. However, the first two can only represent closed shapes, while the unsigned distance has difficulties in accurate and fast shape inference. In this paper, we propose a Neural Vector Field for shape representation in order to overcome the two aforementioned problems. Mapping each point in space to the direction towards the closest surface, we can represent any type of shape. Similarly the shape mesh can be reconstructed by applying the marching cubes algorithm, with proposed small changes, on top of the inferred vector field. We compare the method on ShapeNet where the proposed new neural implicit field shows superior accuracy in representing both closed and open shapes outperforming previous methods.

Keywords: Implicit Function, Vector Transform, Signed Distance Field, Marching Cubes

1 Introduction

Representing 3D shapes has long been a challenge in computer graphics and 3D vision. A 3D surface representation should have high accuracy and should be convenient for any downstream task. Shape analysis tasks such as 3D shape correspondences [49], 3D deformations [46], registration [15] or generation [37] rely on 3D representations suitable for learning. Voxel-based representations can leverage convolutional methods developed in image processing, but can only be used with relatively low resolution as they come with large time and memory requirements. Point clouds and meshes, on the other hand, have lower memory requirements but are far harder to process in a learning pipeline. These are considered explicit representations as they use the actual positions of the structure in 3D. Alternatively, there are hybrid methods that are based on representations that can be used either for implicit processing or explicit visualization. For instance, [13,4] interprets 3D objects as composed of multiple polytopes but lacks the convenience of other representations.

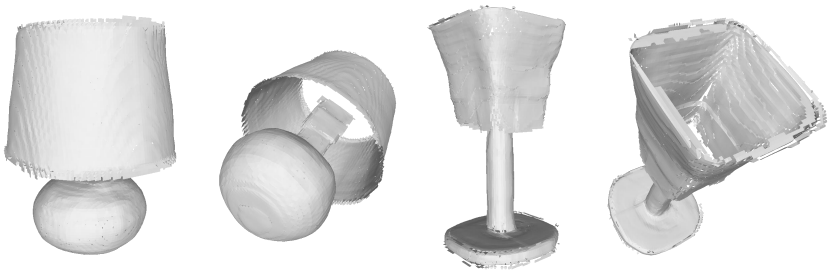

Fig. 1. Open surface visualizations. The proposed VT representation can encode open surfaces and can be converted to a mesh representation with the efficient Marching Cubes algorithm.

In the recent years, neural implicit functions [31,27,38,5] have been proposed to overcome these problems. They are used to represent 3D shapes as continuous functions that map spatial locations to some shape context, such as the signed distance to the surface. If the chosen shape context is sufficient to retrieve the original shape, then it forms a valid shape representation. It thus follows that a neural network that can approximate the aforementioned shape context is able to represent a surface at arbitrary resolution with fixed memory requirements. There are mainly two types of implicit representations that are used in the computer vision and graphics community. Distance based representations [31,8] associate to each point in space the distance from the closest surface of the object to be represented. This is either signed or unsigned, with the former that gives the negative sign to points inside the objects and positive outside. An alternative representation is binary occupancy [27], where each point in space is classified to be either inside or outside an object.

Despite the success [49,3,5,38,39] of implicit representations, popular representation approaches [31,27] cannot formulate the training objective when the inside-outside attribute of space is not clear. Examples of such cases, however, are plentiful, and commonly encountered in open surfaces, non-manifold geometry, or non-orientable surfaces. To overcome this issue, [8] suggests to substitute the SDF with its unsigned version (UDF). With this change it is possible to represent every type of surface as the zero level set of the UDF. However, this comes with the problem that the surface cannot be discovered as a level set anymore due to prediction noise. Rather the minimum is obtained by following the opposite of the gradient of the distance function [8]. However, the requirement of differentiating at test time for inference comes at higher memory and time cost compared to the previous methods.

Inspired by image boundary representation in [35], we propose to overcome these limitations with an implicit function that maps each point in space to the normalized direction towards the closest surface. In accordance with its 2D counterpart, we call this representation VT. This is a significant shift from previous implicit functions that have focused on predicting scalar fields. In contrast, the

proposed vector field allows us to represent any type of surface while being fast in inference. More specifically, surface points are found at the -2 level set of the divergence computed on the vector field and this one-to-one mapping between the surface point set and the divergence level set allows us to apply the marching cubes (MC) algorithm to our representation with minimal changes. We show example mesh visualizations of open surface representations using VT in Fig. 1.

In the rest of the paper, we first look at VT and a modified magnitude version of it (DVT) and show that these representations can be used to represent 3D surfaces. Then we show how the representations can be learned and integrated with MC at test time. Finally, we demonstrate that our proposed representation can achieve superior performance compared to any other representation when applied in equal train-test conditions.

In summary our contributions are threefold:

- We propose to use VT and DVT, two vector field representations for implicit representation of 3D shapes and demonstrate their soundness and higher expressive ability.
- We modify the marching cubes algorithm (Algo. 1) to be applied on a vector field, so that it is able, for the first time, to handle open surfaces, non-manifold geometries, and non-orientable objects.
- With extensive tests, we demonstrate the strong performance of our method, achieving superior accuracy and generalization in multiple testing set-ups.

2 Related Work

We mainly review three areas related to our work; 3D shape representation, implicit functions, and the MC algorithms.

2.1 Voxel, Point Cloud, and Mesh-Based Representations

Voxel grids [19,22] extend to volumetric data the pixel image structure and have been used with straightforward extensions of image processing techniques, such as convolutions, to 3 dimensions. However, the clear drawback of voxel-based methods is that they scale cubically with the resolution in terms of memory usage and computation. Therefore, the first methods proposed [9,42,45] could only work with 32^3 voxel grids. It has been later improved to 128^3 [44,48] with drawbacks in terms of network sizes and training speed, while still being too small to represent 3D data with high fidelity. The dimension of voxel grids can be improved with multi-resolution methods [18] that deal with grid sizes up to 256^3 but have high complexity.

To preserve the same structure but alleviate the computation limitations, octree-based methods [41,36] have been proposed, which could improve resolution up to 512^3. These cannot still produce visually compelling results as they do not predict normals and do not smooth predictions to a sub-voxel resolution. Alternatively, [11,40,23] store the truncated signed distance function (TSDF)

[10] at a voxel level, which allows to represent surfaces at a sub-voxel resolution. However, these methods still require large amounts of memory as the TSDF values are stored in a voxel grid and they are much harder to learn.

Point clouds, an alternative to voxel representations, solve the memory requirement problem as they are constituted of the coordinates of the points that lie on the object surface. This has also no theoretical bound on the resolution that can be obtained, even though, the number of points required grows with accuracy. Despite being used both in discriminative learning tasks [32,33] and for reconstruction [16], they require non-trivial post-processing to be converted to meshes, the preferred format for shape manipulation and rendering. Furthermore, point clouds do not provide important information such as surface normals and connected components in the representation.

Meshes represent surfaces as a combination of polygons, generally triangles, and have often been used to represent classes of similarly shaped objects, like body parts. They are used for classification or segmentation by applying convolutions to their graph structure [1,17], exploiting the information on connectivity and the normals. When used in the reconstruction task, they often create self-intersecting structures and can only produce simple topology [43]. Alternatively, they require a template representative of the class that is reconstructed [21,34].

2.2 Implicit Functions

Learning implicit functions to represent the 3D structure of objects has been proposed as a solution to the memory issues and utility of the 3D representations discussed above. These methods require small amounts of memory to represent objects at arbitrary resolution. The most successful methods in doing this have been based on classifying points in space as inside or outside of objects using either binary occupancy classification [27,7,14,38] or SDF [31,6,20,5]. These have the further advantage that they represent closed surfaces and can be used to generate watertight meshes with the quick MC algorithm. However, they cannot be used for open surfaces or non-manifold structures or objects with non-orientable surfaces such as the Klein bottle. Moreover, they require watertight structures to be trained, which are not trivial to obtain in real world scenarios. To solve this issue, the UDF [8] has been used as an alternative as it is able to represent any type of object. However, it outputs point cloud representations which still have the limitations highlighted before. Note that Neural Radiance Fields [28,47,26] can encode open surfaces with volume density, thanks to the camera rays which enable an inside-outside definition. However, these approaches are more attuned to synthesizing realistic views in presence of image-camera pose examples, more than on inferring 3D shape.

2.3 Marching Cubes Algorithms

Marching cubes [25] is an algorithm which has obtained tremendous success for speeding up the creation of a surface mesh representing a level set in a scalar field sampled on a cuberille grid. Each vertex in the cube-like structure is classified

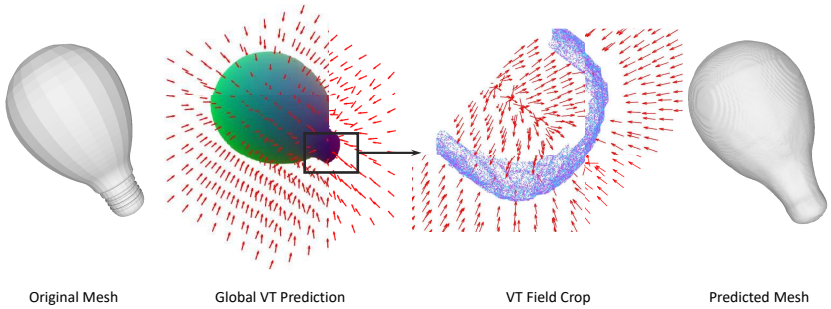

Fig. 2. Overview of our proposed VT representation. From left to right we show an example mesh of an object to represent, the predicted VT field and a zoom-in of a crop of it, and finally the reconstructed object.

as either inside or outside based on whether its value is below or above the level set. The algorithm is based on the fact that, of all the 256 configuration of inside and outside that a cube can have, they can be reduced to 15 unique ones, which represent all of them up to rigid geometrical transformation. This, however, can still lead to holes as the values are not adapted based on the configuration in neighboring cubes. Successive versions of MC have tackled the problem, for example, by introducing trilinear interpolation inside the cubes while improving the overall mesh accuracy [24,29]. We use [24] as starting point to develop an updated version of the algorithm that works on a vector field.

3 Method

In this work, we propose to replace the scalar field implicit functions with a vector field counterpart. The existing implicit representations, are mostly distance based or binary. [31] and [8] compute the SDF or UDF respectively, from the surface to be represented, and [27] uses a single variable to formulate a binary classification of the points in space.

In creating new representations, though, we want to preserve the advantages of existing methods, for example fast inference and accurate mesh reconstruction [31,27]. Unlike in the case of UDF inference [8], we want the inference step to involve only forward passes. To that end, we propose to represent 3D objects by the normalized direction towards the closest surface point, adapting the representation proposed for image boundaries in [35]. In the remainder of the section, we first formally define our vector field function with the properties that make it suitable for the representation task; then, we compare its properties to the commonly used scalar fields and discuss its training procedure. Finally, we show how this particular vector representation allows the fast inference of 3D meshes with only small changes to the MC algorithm.

3.1 Vector Field Representation

Suppose we have a 3D object or scene in a subset of 3D space $\Omega \subseteq \mathbb{R}^3$. A continuous 2D surface therefore lies in it, $\Pi \subset \Omega$, with surface points $\mathsf{x}_S \in \Pi$ and non-surface points $\mathsf{x} \in \Omega \setminus \Pi$. Let us call $\Gamma \subset \mathbb{R}^3$ the space of $\ell 2$-normalized 3D vectors with members $\mathsf{v} \in \Gamma$. We finally define a transform that maps each point x to the normalized direction v pointing towards the closest surface point x_S. Given the definition, the points x_S are discontinuities in the field as positions where the direction towards the surface flips. For practical purposes, they are mapped to either of the two opposite directional normals of the surface, without any downsides. Generalizing to non-differentiable points where no normal is defined, this is equivalent to treating the point x_S as a point $\mathsf{x}' = \mathsf{x}_S + \epsilon$ s.t. $\mathsf{x}' \notin \Pi$ and $\|\epsilon\|_2 \to 0^+$ and computing the transform for point x'.

In accordance with [35], we call the mapping from the positions in space to the directions $\mathsf{f}_{VT} : \Omega \to \Gamma$ Vector Transform (VT). Figure 2 shows the visualization of the VT representation for an example object. We redefine VT formally for the 3D-space as follows:

$$\begin{aligned} \mathsf{f}_{VT}(\mathsf{x}) = \mathsf{v} \quad \text{with} \quad \mathsf{x} \in \Omega, \quad \mathsf{v} \in \Gamma \\ \mathsf{v}_a = \frac{\mathsf{d}_a(\mathsf{x}, \hat{\mathsf{x}}_S)}{\|\mathsf{d}(\mathsf{x}, \hat{\mathsf{x}}_S)\|_2} \quad \text{with} \quad a \in \{x, y, z\} \\ \text{and} \quad \hat{\mathsf{x}}_S = \inf_{\mathsf{x}_S \in \Pi} \|\mathsf{d}(\mathsf{x}, \mathsf{x}_S)\|_2 \quad \text{if} \quad \mathsf{x} \notin \Pi, \\ \text{otherwise,} \quad \mathsf{v}_a = \frac{\mathsf{d}_a(\mathsf{x}', \hat{\mathsf{x}}_S)}{\|\mathsf{d}(\mathsf{x}', \hat{\mathsf{x}}_S)\|_2} \quad \text{with} \quad \mathsf{x}' = \lim_{\|\epsilon\|_2 \to 0^+} \mathsf{x} + \epsilon \quad \text{and} \quad \mathsf{x}' \notin \Pi. \end{aligned} \tag{1}$$

Here d is an operator that gives the vector difference of its operands. Equation (1) defines the transform that maps each point in the subset of space Ω to a normalized direction vector $\mathsf{v} \in \Gamma$. Here $\epsilon \in \mathbb{R}^3$ is an infinitesimal displacement. We note that when multiple points $\hat{\mathsf{x}}_S$ satisfy the inferior condition, one of them is chosen randomly. Similarly, the point x' can be chosen arbitrarily between the two sides of the surface Π with the only constraint that $\mathsf{x}' \notin \Pi$.

We now need to ensure that VT can properly represent surfaces embedded in $\mathbb{R}^3$. Specifically, we establish a one-to-one map between surface points and the -2 level set of divergence computed on the VT field.

Property 1. The vector field $\mathsf{v} = \mathsf{f}_{VT}(\mathsf{x})$ is equal to the unit normal field at the surface.

Property 1, instrumental for proving the following property, does not require extensions and we refer to its proof in [30] and [35].

Property 2. Consider the VT representation $\mathsf{v} = \mathsf{f}_{VT}(\mathsf{x})$ of a piece-wise smooth surface as defined in Eq. (1), and the following transform:

$$g(\mathsf{x}) = \operatorname{div} \mathsf{f}_{VT}(\mathsf{x}). \tag{2}$$

A point $\mathsf{x} \in \mathbb{R}^3$ then is a surface point, $\mathsf{x} \in \Pi$, if and only if it belongs to the zero level set of $g(\mathsf{x}) + 2$,

$$\Pi = L_0(g + 2). \tag{3}$$

Proof. Let us start proving that, if a point belongs to the surface, then it is in the zero level set of Eq. (3). In the infinitesimal sphere around any surface point, using Property 1, the only component of the VT field present is normal to the surface, with tangential components approaching to 0. Around such an infinitesimal neighborhood, the normal fields on the two sides of the surface, pointing in opposite direction, are summed with concordant signs. Thus, they create a divergence flow of -2, as they are pointing inward and with norm equal to 1 by definition. Therefore, we have demonstrated that it belongs to the zero level set $L_0(g + 2)$. The proof for the second part of the statement is provided in the supplementary.

The result just obtained holds for piece-wise smooth surfaces [30]. However, in practice, small changes can occur on the field divergence on the surface. For example, the divergence can go lower than -2 at discontinuous surface points. Nonetheless, the thresholding operation required for surface inference remains unaffected. Thanks to these generalized properties, we can now use VT as a surface representation with the knowledge that there is a theoretical guarantee that shows the correctness of the transform.

3.2 Comparison of Representations

As we now have a formal definition of VT and its main properties, we can compare the proposed representation with the previous ones. From the expressiveness perspective, we have shown that any type of piece-wise smooth surface can be represented through VT; this includes open and non-orientable surfaces. It can be further extended to non-smooth surfaces and non-manifold structures as these show an even lower divergence. It is thus a strong improvement in expressiveness compared to SDF [31] and binary occupancy [27] that could only represent closed watertight surfaces.

While considering distance based representations, it is important to note that [8] can represent any type of surface, similar to VT. However, VT has a higher sensitivity around the boundary compared to SDF and UDF, where small differences in the prediction are not strongly penalized by the distance loss. Furthermore, SDF and UDF require training with a truncated distance field to make the task stable. Besides adding an hyperparameter to optimize, this reduces the shape prior learned far from the surface as shown with the experiment on the shape completion task. Using VT, this is not needed, which results in a richer representation as every point needs to be aware of the surface position.

The proposed VT shares strong relationship to UDF, as the negative gradient of UDF is equal to VT,

$$\mathsf{f}_{VT}(\mathsf{x}) = -\nabla \,\mathrm{UDF}(\mathsf{x}). \tag{4}$$

It may therefore raise the question whether the UDF prediction can achieve the same results as VT by considering the gradients. However, as shown in the experiments, applying the divergence criterion to infer a surface on the UDF

gradients suffers from significant noise on the gradient prediction. Furthermore, as UDF is not symmetric around 0, it suffers from a bias in the prediction around the middle of the range between 0 and the maximum distance threshold. This, together with the low sensitivity around 0 is a factor for the high level of noise on UDF prediction as demonstrated in [35] for image edge detection.

3.3 Distance-Vector Transform

While the VT representation for 3D shapes provides enough context for shape inference, it can be reasoned that more context may be helpful for better results. One way to achieve that is through a modification of VT, by also encoding the unsigned distance from the surface, thus splitting the distance transform into separate components, similarly to [12]. We call this method Distance-VT (DVT). Specifically, we encode the norm of the VT vectors with the distance from the surface. Apart from having more shape context, DVT also makes it possible to learn the representation exactly with a continuous function. To preserve a reliable prediction of direction, we split the loss in a directional component applied on the normalized DVT vectors and a distance component applied on the norm. To avoid the undefined case of zero norm at the surface, the norm of the points used in training has a lower bound set at 10^{-5}. Note that this does not affect the learning in practice as the vast majority of points used for training are further from the surface. We also note that DVT can learn even more information as the distance from the surface is directly available in the representation.

However, the highlighted advantages come with a trade-off as DVT requires the use of a more complex loss and has a reduced sensitivity at the surface. Having a high loss close to the surface allows the network to focus its training and refinement on the surface values. On the other hand, having a continuous representation makes convergence to the target representation easier and the directly available distance value constitutes an advantage in applying the MC algorithm. As shown in the experimental section, VT outperforms DVT when complete observations are used for refinement, suggesting that the higher sensitivity at the surface is more important than the advantages of DVT for representation. However, it also shows that DVT can provide advantages in other conditions.

From a theoretical point of view, DVT can be easily shown to be an implicit representation as it can be simply converted to VT by normalization. Despite the presence of the distance metric that could provide information on the surface position, we use the divergence level set relation as it provides higher robustness and is easier to threshold, as shown in the previous work on images [35].

3.4 Field Creation and Training

Following standard practices [31,49], we want to measure the performance of VT field for shape inference while also considering the generalization capabilities. Therefore, the field prediction at each coordinate is conditioned on a high dimensional embedding vector. This is specific to the object to represent and is

given as input variables to the network together with the inference coordinates to output the representation.

Considering training, the mean squared error (MSE) loss is computed between the predicted vector $\hat{\mathsf{v}}$ with components x, y, and z and the ground truth vector v^{gt}:

$$\ell_{VT} = \|\mathsf{v}^{\mathsf{gt}} - \hat{\mathsf{v}}\|_2^2. \tag{5}$$

Here v^{gt} is the ground truth VT field and $\hat{\mathsf{v}}$ is the prediction. More specifically, for each object, it is computed on a set of coordinates randomly sampled in space with a higher density around the surface [31].

The same mean squared error loss is applied to the normalized DVT with the addition of the $\ell 1$ loss on its norm. Overall, the loss for DVT is the following:

$$\ell_{DVT} = \left\| \mathsf{v}^{\mathsf{gt}} - \frac{\hat{\mathsf{v}}}{\|\hat{\mathsf{v}}\|_2} \right\|_2^2 + \left\| \mathsf{d}^{gt} - \|\hat{\mathsf{v}}\|_2 \right\|_1 . \tag{6}$$

Here v^{gt} is the ground truth VT field and $\hat{\mathsf{v}}$ is the prediction, as before, and d^{gt} is the ground truth distance from the surface. We note that here, compared to Eq. (5), it is possible to add an hyperparameter to weight the two loss components, which may improve performance at the cost of more experimental complexity and, possibly, less generality.

3.5 Mesh Inference and Marching Cubes Algorithm

A challenging part of implicit 3D representation is an easy transformation to standard mesh representation. Mesh allows rendering and manipulation of 3D shapes using standard graphics tools. For the purpose of going from implicit to mesh representation, a traditionally successful algorithm has been MC. Current MC algorithms are developed to produce smooth surfaces without holes and with continuously changing normals. This produces visually appealing representations but constitutes a challenge when adapting the algorithm for different types of representation.

The standard MC algorithm is applied to a scalar field to produce triangles at the positions where the scalar field crosses a predefined value. Considering SDF, the value used is 0 and the resulting polygonal surface approximates the zero level set of SDF. This is done taking by first voxelizing the space and evaluating the field values in these voxels. Furthermore, in order to choose between the multiple possible mesh configurations that arise from the vertex assignments, neighboring voxels are used to ensure surface continuity. To secure smoothly changing normals, the MC algorithms compute them based on a neighborhood around each surface position and locate the mesh faces in each voxel with a trilinear interpolation of the field.

To adapt the MC algorithm to the VT field, we now assess the three aspects just highlighted:

– **Level set definition**: as the vector field does not have a level set that can be used to apply MC, we need to define a way to indicate the voxels which have a

surface. For this, we use Property 2 and identify the surface voxels as the ones that have a divergence smaller or equal to -2. The divergence computation can be computed in a highly parallel manner, either with convolutions on the voxel grid or inside the MC computation.

- **Vertices clustering**: the directions in each vertex of a voxel are then clustered into two groups, based on their cosine similarity. This replaces the inside-outside direction computation in the case of SDF. In this case, the two clusters can be identified by taking the two vectors among the 8 inside the voxel with the lowest cosine similarity between each other. The other vectors are then associated to the cluster with which they share the highest cosine similarity. This is an effective clustering technique with very low computational cost, thanks to the easy nature of the problem.
  The assignment algorithm just explained is enough to generate a surface in each voxel but it does not ensure a continuous result, as the assignment on the two sides of the level set is not consistent between neighboring voxels. Note that this is not a problem with SDF because of the unambiguous inside-outside definition. We solve this for VT by applying a region-growing algorithm which ensures consistency of normal directions within all parts of the object. Starting from a random surface point, one of its two dominant directions is randomly chosen; then, its adjacent surface points have the dominant direction chosen as the one consistent with the one $\hat{\mathsf{v}}$ of the starting point. Among the two opposite normal directions v_1 and v_2, one is chosen so that the bisector between it and $\hat{\mathsf{v}}$ is perpendicular to the displacement vector from the starting point to the considered neighboring point. This relationship is exact for exact prediction but can be approximated to obtain consistent directions. This operation is then recursively repeated starting from the new points. When no new surface point to be set can be reached, a new random surface point is chosen between the unset ones. In this way we achieve normal consistency in object parts. However, we should note that consistency guarantees at a global level cannot be ensured as some objects - like the Klein bottle - do not allow a definition of inside or outside. Non-manifold geometries cannot be represented by MC either; however, excluding the points of connection of three surfaces, the rest of the structure can be faithfully represented.
- **Smooth normal predictions**: the final step to adapt is the use of values for the vertices to exploit the trilinear interpolation. When considering DVT, it is possible to use the predicted distance embedded in the vector field structure; the norm of the vectors is assigned to each voxel vertex with the sign determined as previously explained. In the case of VT field, there is no exact representation as it does not have explicit notion of the distance from the surface. However, similarly to DVT, we can use the continuity property of the representation to have a distance estimation. As the surface is defined by points where field directions flip, we observe that the norm of the field gets reduced around the surface points. We note that, even though this measure cannot represent the actual value of the distance, it monotonically changes close to the surface and hence can be used for our purpose. The

same effect is also used when applying the traditional MC algorithm on the binary occupancy field [27].

Algorithm 1 MCvector $(\mathcal{X}, \mathsf{f}_{VT}(\mathsf{x}))$

1. Sample point set $\mathcal{X} \subset \mathbb{R}^3$ to form a grid.
2. For each vertex $\mathsf{x} \in \mathcal{X}$, compute the VT field, compute the divergence, and evaluate Eq. (3).
3. For voxels in the zero level set of Eq. (3) or below zero, cluster vertices on the opposite side of the surface and make directions consistent with the region growing algorithm.
4. Assign the norm of the predicted VT or DVT to the vertices and generate the surface.

For brevity, our modified MC method is described in Algo. 1. For a quantitative comparison between the proposed algorithm and the one used as reference [24], we refer the reader to the supplementary material.

4 Experiments

First, we explain the network structure and training set-up together with the tasks that we tackle followed by the metrics used. Then we describe the results on each task comparing the performance of VT to other representations. Finally, we show more qualitative results and provide a discussion on the performance.

4.1 Network Structure and Methods

Our work proposes a surface representation rather than a method with its own architecture. We describe the standard network architecture used for all representations here. Every representation is tested on the same architecture and the same training technique. More specifically, we chose to use the same architecture as [31], as it achieves good results while being reasonably lightweight and does not require any operation to be executed on the expensive voxel structure.

We use a fully connected auto-decoder network that takes as input a latent vector together with the prediction coordinates to predict the field in such position. In this context, both the network and the latent code are optimized at training time, while only the latent vector is optimized at test time. The network is trained for 2000 epochs with samples from 64 scenes in each batch and 16384 points per batch. For further details concerning the network structure and the training, please refer to the original work [31].

The described network structure is used to predict a binary occupancy representation (BOR), the thresholded signed and unsigned distance field (SDF and UDF) and the VT field with its magnitude version (DVT). BOR is trained using the binary cross-entropy loss, SDF and UDF are trained using the $\ell 1$ loss and

VT with the mean squared error following previous work on the field [31,27,35]. The threshold value for the signed and unsigned distance field is also set based on previous work [8,31].

Considering inference, we first sample the fields on a 512^3 voxel grid and then apply MC algorithm on it. BOR and SDF can be directly used with the standard version of MC [24] to provide the resulting mesh. Regarding UDF, up to our knowledge, there is no method like MC that can produce a resulting mesh in a comparable way. Therefore, to evaluate the method fairly with respect to the others, we apply the proposed MC variant on its negative gradient as we do on VT and DVT.

4.2 Tasks, Metrics, and Dataset

We apply the representation methods just explained on two different tasks. The first is the traditional shape reconstruction task. Here, the network is first on a shape class, and then used to reconstruct an unseen object belonging to the same class without retraining the network. The second task is focused on reconstructing shapes of unseen objects belonging to the same class used in training, using only partial observations of the object at test time. This is called the completion task. Specifically, each object is observed frontally and the ground truth values are sampled around the observable points. For more analysis on the completion task changing the view point, we refer to the supplementary material.

All the tasks are evaluated using the symmetric Chamfer distance (CD) on 30000 points with the results written as $CD \times 1000$. Specifically, 30000 points are uniformly sampled on the ground truth and the predicted mesh, and the average distances from each point in the set to the closest in the other are summed. To reduce the random effect of points sampling, each result is obtained by averaging the results over three different samplings.

We evaluate every method on multiple classes of the popular ShapeNet [2] dataset. Specifically, we evaluate the reconstruction performance on the *chairs*, *lamps*, *planes*, and *sofas* classes separately. Among these classes, *lamps* has the smallest amount of training data available and therefore it is not used in the shape completion task. For every task, each class is divided into a training, validation and testing set, with respectively 70%, 10%, and 20% of the data.

4.3 Shape Reconstruction

First, we test on the shape reconstruction task. Table 1 shows that the two best performing representations, on average, are SDF and VT, with BOR being also competitive with an appropriate amount of training data. This shows that the vector representation on neural implicit shape representation can perform as well as SDF. Furthermore, VT can consistently outperform UDF which is the only other method with the same expressive power that can represent non-watertight shapes. We also note that VT can perform better than DVT, which suggests that having a high sensitivity at the surface is possibly more important than modeling

Method	*chairs*		*lamps*		*planes*		*sofas*	
	mean	median	mean	median	mean	median	mean	median
BOR	0.4199	0.1732	3.0023	0.4481	0.1615	0.0266	0.1492	0.0931
SDF	0.3373	0.1558	0.7977	**0.1831**	0.1004	0.0473	**0.1197**	**0.0734**
UDF	0.8522	0.4230	1.2385	0.4401	0.6448	0.3989	0.6113	0.4044
VT	**0.3222**	**0.1542**	**0.6244**	0.1896	**0.0739**	**0.0243**	0.1689	0.0755
DVT	0.6290	0.3474	0.8436	0.2777	0.3472	0.1614	0.5748	0.2064

Table 1. Reconstruction results on 4 ShapeNet [2] categories. The latent vector used to represent the object has size 256 and is optimized for 800 iterations.

Method	*chairs*		*planes*		*sofas*	
	mean	median	mean	median	mean	median
BOR	12.4128	11.8306	6.1921	5.6087	13.4265	12.0069
SDF	8.6721	8.2049	4.1413	3.2754	8.4861	7.3419
UDF	3.0013	2.5488	1.1495	0.6468	4.3452	3.1917
VT	4.4817	3.6761	1.5929	1.0629	4.3001	2.8116
DVT	**2.9433**	**2.4462**	**1.0770**	**0.4157**	**3.6209**	**2.1890**

Table 2. Completion results on partial observations of 3 ShapeNet [2] categories. *Lamps* are excluded as they are a class with high variability and a very limited training set. The latent vector representing the object has size 128 and it is optimized for 100 iterations. These values are reduced from the previous task to give more importance to the learned prior.

a continuous function. Finally VT's slight edge over SDF in some cases can be accounted to the sharp sensitivity of the field divergence in recognizing surfaces.

4.4 Shape Completion

The results on the shape completion task, shown in Table 2, further show the suitability of the VT representation. As hypothesised, using a vector field can provide a stronger prior on the learned shapes compared to the similarly performing scalar fields. Supporting the same hypothesis, BOR is the method that performs worse as it is the method with the least prior on the shape as each point inside an object is treated equally. Furthermore, the better performance of UDF with respect to SDF can be explained by the different parameters typically used as distance threshold [31,8]. As UDF has a higher threshold, its shape prior is stronger than SDF, which allows it to better complete shapes.

On the contrary, the very high loss at the surface, which creates a large gradient for the latent vector optimization can be a drawback with a limited view of the objects. This is suggested also by the stronger performance of DVT compared to all other methods. In fact, DVT has a rich vector representation and does not suffer from excessively large gradients given the smooth transition at the surface.

4.5 Qualitative Results

In Figure 3 we show some qualitative predictions obtained using VT for all the classes in the reconstruction part. The predictions closely follow the target

Fig. 3. Qualitative comparison. In each line we show from left to right the target mesh, the reconstruction mesh obtained with VT and DVT and the completion mesh of the two methods. Please refer to the supplementary material for more comparisons.

shapes in most cases despite the challenging examples. Furthermore, the lack of holes in the predictions and the consistency in the surface normal suggests that the proposed variant to the MC algorithm is effective in outputting high quality meshes. For more qualitative results on different tasks, please refer to the supplementary material.

5 Conclusion

In this paper we revisited the neural implicit representations for representing 3D shapes and analyze the drawbacks of currently available formulations. We started with the observation that popular representations such as the SDF or binary occupancy either lack the expressive power to represent open or non-watertight surfaces while the UDF representation has problems with fast inference or accurate representation. We therefore, proposed to solve the issue of open surface representation by considering vector transform in place of the popular signed/unsigned distance transform. We then modified the popularly used Marching Cubes algorithm to work with the proposed vector field implicit representation. Our complete method provides fast and accurate 3D shape inference along with mesh computation without the requirement of backpropagation. This is possible while representing a much larger class of shapes compared to the popular SDF implicit representation.

Acknowledgements. This research was funded by Align Technology Switzerland GmbH (project AlignTech-ETH). Research was also funded by the EU Horizon 2020 grant agreement No. 820434.

References

1. Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: going beyond euclidean data. IEEE Signal Processing Magazine **34**(4), 18–42 (2017) 4
2. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015) 12, 13, 20, 21, 22, 23, 25
3. Chen, Z., Zhang, Y., Genova, K., Fanello, S., Bouaziz, S., Häne, C., Du, R., Keskin, C., Funkhouser, T., Tang, D.: Multiresolution deep implicit functions for 3d shape representation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13087–13096 (2021) 2
4. Chen, Z., Tagliasacchi, A., Zhang, H.: Bsp-net: Generating compact meshes via binary space partitioning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 45–54 (2020) 1
5. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: CVPR (2019) 2, 4
6. Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019) 4
7. Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit functions in feature space for 3d shape reconstruction and completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6970–6981 (2020) 4
8. Chibane, J., mir, M.A., Pons-Moll, G.: Neural unsigned distance fields for implicit function learning. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 21638–21652. Curran Associates, Inc. (2020), https://proceedings.neurips.cc/paper/2020/file/f69e505b08403ad2298b9f262659929a-Paper.pdf 2, 4, 5, 7, 12, 13
9. Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) Computer Vision – ECCV 2016. pp. 628–644. Springer International Publishing, Cham (2016) 3
10. Curless, B., Levoy, M.: A volumetric method for building complex models from range images. In: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques. pp. 303–312 (1996) 4
11. Dai, A., Ruizhongtai Qi, C., Nießner, M.: Shape completion using 3d-encoder-predictor cnns and shape synthesis. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5868–5877 (2017) 3
12. Danielsson, P.E.: Euclidean distance mapping. Computer Graphics and image processing **14**(3), 227–248 (1980) 8
13. Deng, B., Genova, K., Yazdani, S., Bouaziz, S., Hinton, G., Tagliasacchi, A.: Cvxnet: Learnable convex decomposition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 31–44 (2020) 1
14. Deng, B., Lewis, J.P., Jeruzalski, T., Pons-Moll, G., Hinton, G., Norouzi, M., Tagliasacchi, A.: Nasa neural articulated shape approximation. In: European Conference on Computer Vision. pp. 612–628. Springer (2020) 4
15. Eisenberger, M., Lahner, Z., Cremers, D.: Smooth shells: Multi-scale shape registration with functional maps. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12265–12274 (2020) 1

16. Fan, H., Su, H., Guibas, L.J.: 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. pp. 605–613 (2017) 4
17. Guo, K., Zou, D., Chen, X.: 3d mesh labeling via deep convolutional neural networks. ACM Transactions on Graphics (TOG) **35**(1), 1–12 (2015) 4
18. Häne, C., Tulsiani, S., Malik, J.: Hierarchical surface prediction for 3d object reconstruction. In: 2017 International Conference on 3D Vision (3DV). pp. 412–420 (2017). https://doi.org/10.1109/3DV.2017.00054 3
19. Ji, M., Gall, J., Zheng, H., Liu, Y., Fang, L.: Surfacenet: An end-to-end 3d neural network for multiview stereopsis. 2017 IEEE International Conference on Computer Vision (ICCV) (Oct 2017). https://doi.org/10.1109/iccv.2017.253, http://dx.doi.org/10.1109/ICCV.2017.253 3
20. Jiang, C., Sud, A., Makadia, A., Huang, J., Nießner, M., Funkhouser, T., et al.: Local implicit grid representations for 3d scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6001–6010 (2020) 4
21. Kanazawa, A., Black, M.J., Jacobs, D.W., Malik, J.: End-to-end recovery of human shape and pose. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7122–7131 (2018) 4
22. Kar, A., Häne, C., Malik, J.: Learning a multi-view stereo machine. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/file/9c838d2e45b2ad1094d42f4ef36764f6-Paper.pdf 3
23. Ladicky, L., Saurer, O., Jeong, S., Maninchedda, F., Pollefeys, M.: From point clouds to mesh using regression. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3893–3902 (2017) 3
24. Lewiner, T., Lopes, H., Vieira, A.W., Tavares, G.: Efficient implementation of marching cubes' cases with topological guarantees. Journal of graphics tools **8**(2), 1–15 (2003) 5, 11, 12, 25
25. Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. ACM siggraph computer graphics **21**(4), 163–169 (1987) 4
26. Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7210–7219 (2021) 4
27. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019) 2, 4, 5, 7, 11, 12
28. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European conference on computer vision. pp. 405–421. Springer (2020) 4
29. Nielson, G.M., Hamann, B.: The asymptotic decider: Rosolving the ambiguity in marching cubes. eScholarship, University of California (1991) 5
30. Osher, S., Fedkiw, R.P.: Level set methods and dynamic implicit surfaces, vol. 153. Springer (2003) 6, 7
31. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2019) 2, 4, 5, 7, 8, 9, 11, 12, 13, 19

32. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017) 4
33. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems **30** (2017) 4
34. Ranjan, A., Bolkart, T., Sanyal, S., Black, M.J.: Generating 3d faces using convolutional mesh autoencoders. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 704–720 (2018) 4
35. Rella, E.M., Chhatkuli, A., Liu, Y., Konukoglu, E., Gool, L.V.: Zero pixel directional boundary by vector transform. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=nxcABL7jbQh 2, 5, 6, 8, 12
36. Riegler, G., Osman Ulusoy, A., Geiger, A.: Octnet: Learning deep 3d representations at high resolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017) 3
37. Rozen, N., Grover, A., Nickel, M., Lipman, Y.: Moser flow: Divergence-based generative modeling on manifolds. Advances in Neural Information Processing Systems **34** (2021) 1
38. Saito, S., Huang, Z., Natsume, R., Morishima, S., Kanazawa, A., Li, H.: Pifu: Pixel-aligned implicit function for high-resolution clothed human digitization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2304–2314 (2019) 2, 4
39. Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. Advances in Neural Information Processing Systems **33**, 7462–7473 (2020) 2
40. Stutz, D., Geiger, A.: Learning 3d shape completion from laser scan data with weak supervision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1955–1964 (2018) 3
41. Tatarchenko, M., Dosovitskiy, A., Brox, T.: Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (Oct 2017) 3
42. Tulsiani, S., Zhou, T., Efros, A.A., Malik, J.: Multi-view supervision for single-view reconstruction via differentiable ray consistency. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (July 2017) 3
43. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: Proceedings of the European conference on computer vision (ECCV). pp. 52–67 (2018) 4
44. Wu, J., Wang, Y., Xue, T., Sun, X., Freeman, B., Tenenbaum, J.: Marrnet: 3d shape reconstruction via 2.5d sketches. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/file/ad972f10e0800b49d76fed33a21f6698-Paper.pdf 3
45. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015) 3
46. Yifan, W., Aigerman, N., Kim, V.G., Chaudhuri, S., Sorkine-Hornung, O.: Neural cages for detail-preserving 3d deformations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 75–83 (2020) 1

47. Yu, A., Fridovich-Keil, S., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. arXiv preprint arXiv:2112.05131 (2021) 4
48. Zhang, X., Zhang, Z., Zhang, C., Tenenbaum, J., Freeman, B., Wu, J.: Learning to reconstruct shapes from unseen classes. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31. Curran Associates, Inc. (2018), https://proceedings.neurips.cc/paper/2018/file/208e43f0e45c4c78cafadb83d2888cb6-Paper.pdf 3
49. Zheng, Z., Yu, T., Dai, Q., Liu, Y.: Deep implicit templates for 3d shape representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1429–1439 (2021) 1, 2, 8

A Qualitative Results Analysis

Here we provide additional results visualization comparing each representation. From Figure 4, we can see the different properties of each representation. As volumetric representations, SDF and BOR predict watertight meshes. This characteristic, produces smooth and watertight results which can be an advantage when representing shapes with such properties, such as sofas. However, this shows to be a limitation when representing thin objects parts or thin cylinders, such as in the lamp example.

On the other hand, VT, UDF, and DVT represent the surfaces itself which gives a strong advantage when representing this or non-watertight parts, but can lead to non-smooth meshes or noise. The effect is apparent in the meshes obtained using UDF representation and suggests that predicting the direction directly, as in VT and DVT, leads to less noisy results. The advantage on thin shapes, instead, can be observed in the lamps class with a very large performance gap between VT or DVT and SDF. Overall, VT shows to achieve highly accurate prediction across all examples, combining high sensitivity at the surface, the ability to represent any type of shape, and low levels of noise.

Regarding the qualitative examples on the completion task, every method has a significant decrease in qualitative accuracy. However, Figure 5 shows that the same type of properties discussed for the reconstruction task are still valid. When provided with limited observations, SDF and BOR tend to produce overly smoothed results, losing precision at the level of the details. On the other hand, UDF, VT, and DVT tend to suffer from higher levels of noise and often predict open surface. However, the latter group of methods, and particularly DVT, produce results that resemble more the target shape. This significant gap in performance, which is visible across the different shape classes, is consistent with the shown quantitative results.

B Completion Task Analysis

Here we provide two ablations regarding the completion task. First, in Table 3, we study the effect of changing the point of observation of the targets. We consider the surface as if only the part visible from the specified view-point was available, and sample point in space close to it. As represented in Figure 6, we consider 8 different points of view around each object. Since we use the same approach as DeepSDF [31] for generalization, at test time the latent vector needs to be optimized or refined further using the predictions. The second ablation, in Table 4, shows how the results change with varying number of refinement iterations on the prediction using VT method.

Table 3 shows the varying performance of the methods when the view-point is changed. Despite the expected oscillations in performance, the observations suggest that back views of objects generally lead to better reconstructions result. As the front view (V1) is generally richer in details that diverge from the average object structure, this indicates that trying to mimic such details harms the ability

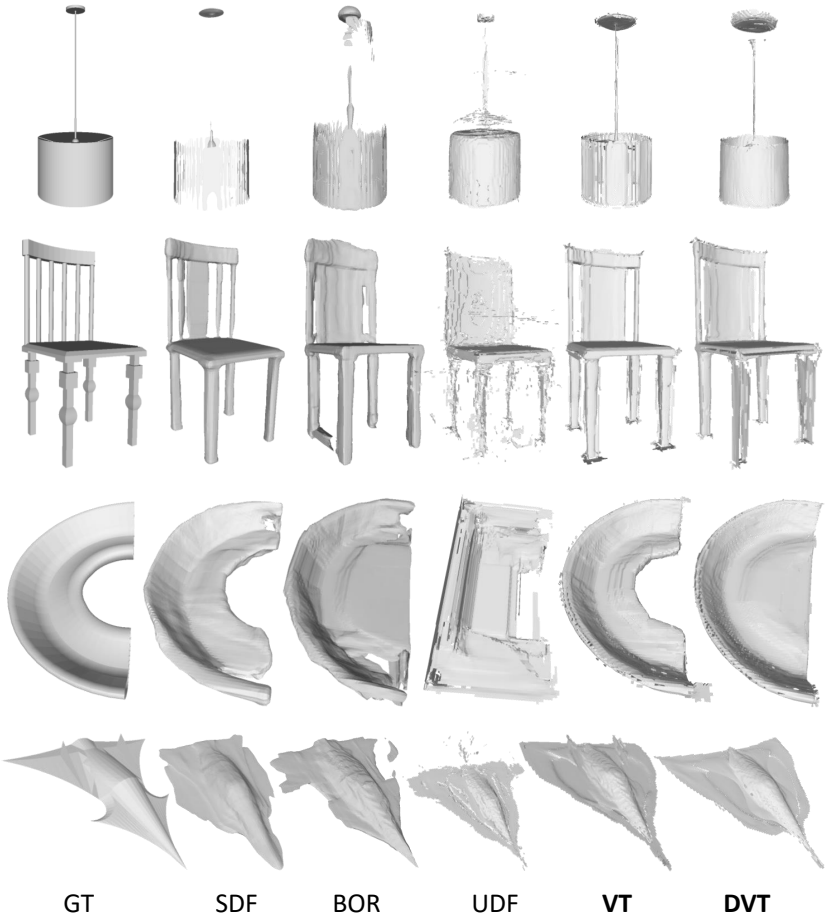

Fig. 4. Qualitative reconstruction. We compare each representation method in reconstructing shapes from each used ShapeNet [2] class. In each line, we show from left to right the target mesh (GT), followed by SDF, BOR, UDF, VT, and DVT.

to represent the overall unseen structure. The smaller variation in performance of VT, DVT, and UDF with respect to SDF and BOR may be explained by the stronger prior learned by the first group of methods.

Table 4 supports the hypothesis that frontal view is harder to use in the completion task. In fact, the frontal view (V1) is the one in which 10 refinement iterations perform best and 250 worst. In contrast, the back views show the smallest difference in performance when changing the number of iterations. Overall, the results suggest that 100 iterations is enough to learn some object-specific detail without losing the previously learned overall structure.

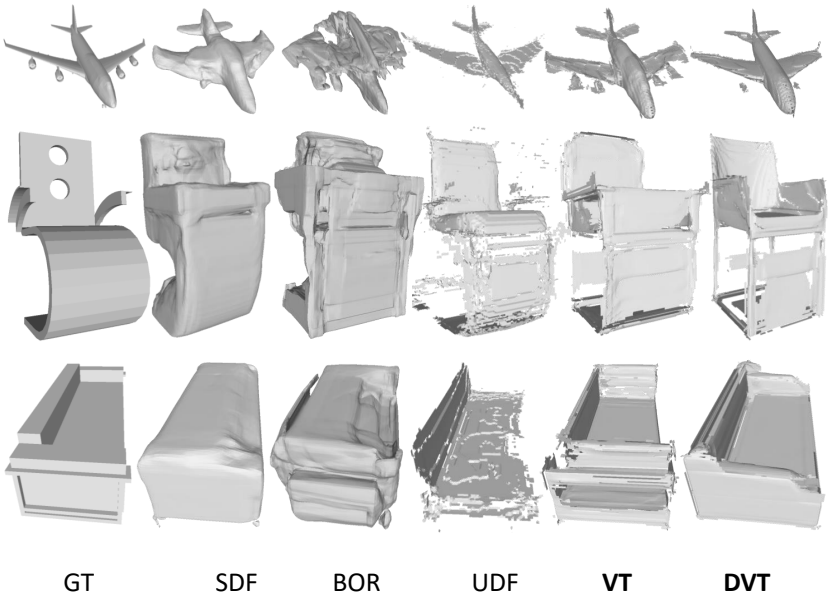

Fig. 5. Qualitative completion. We compare each representation method in the completion task on shapes from each used ShapeNet [2] class. In each line, we show from left to right the target mesh (GT), followed by SDF, BOR, UDF, VT, and DVT.

View	Score	SDF	BOR	UDF	**VT**	**DVT**
V1	mean	8.6721	12.4128	3.0013	4.4817	**2.9433**
	median	8.2049	11.8306	2.5488	3.6761	**2.4662**
V2	mean	6.0960	11.4218	3.1250	3.7523	**2.8243**
	median	5.3626	11.2832	2.6337	3.2037	**2.3726**
V3	mean	4.1596	10.6581	2.8453	2.8813	**2.6283**
	median	3.8719	10.4418	2.3114	2.4367	**2.2596**
V4	mean	4.1566	10.6918	**2.7138**	3.0207	2.7492
	median	3.6067	10.5676	**2.2801**	2.4546	2.3019
V5	mean	4.1707	9.7885	**2.8342**	3.3771	2.9035
	median	3.6694	5.5234	**2.3514**	2.8988	2.4539
V6	mean	3.8994	8.6477	2.8700	3.5332	**2.8565**
	median	3.5007	8.2587	2.2881	3.0080	**2.2191**
V7	mean	4.3471	8.5147	2.7578	3.8569	**2.6089**
	median	3.9181	8.1264	2.3590	3.3221	**2.2458**
V8	mean	5.9195	10.6512	**2.8403**	4.3937	2.9559
	median	5.4887	10.4090	**2.3371**	3.9224	2.4951

Table 3. Completion task on ShapeNet [2] chairs class with observations taken from different positions around the object.

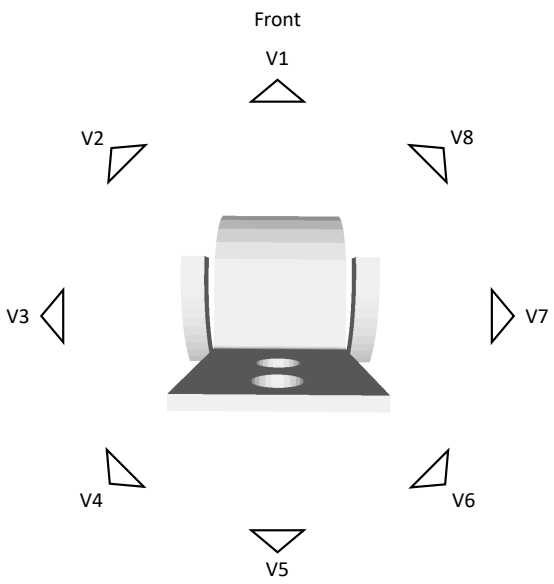

Fig. 6. Completion views. Visualization of the view-points around an object used in the completion task.

View	Score	VT		
		iters = 10	*iters* = 100	*iters* = 250
V1	mean	**4.2490**	4.4817	6.9978
	median	**3.6599**	3.6761	6.1620
V2	mean	4.1972	**3.7523**	4.9274
	median	3.6211	**3.2037**	4.3533
V3	mean	4.2037	**2.8813**	3.4547
	median	3.7838	**2.4367**	2.8115
V4	mean	4.1850	**3.0207**	3.7967
	median	3.7196	**2.4546**	2.9279
V5	mean	4.6637	**3.3771**	3.9095
	median	4.1660	**2.8988**	3.3561
V6	mean	4.9031	**3.5332**	3.7005
	median	4.2356	**3.0080**	3.1213
V7	mean	5.0822	**3.8569**	4.1342
	median	4.4716	**3.3221**	3.5358
V8	mean	5.0677	**4.3937**	5.3902
	median	4.4216	**3.9224**	4.9315

Table 4. Ablation on the number of iterations during inference on the completion task on ShapeNet [2] chairs class. We report all results with our proposed representation VT.

C Reconstruction Robustness to Noise

We evaluate the robustness of different representations when reconstructing object shapes from noisy observations. More specifically, Gaussian noise is added to the coordinates of the observed predictions used for refinements.

Method	$\sigma = 0$		$\sigma = 0.01$		$\sigma = 0.05$		$\sigma = 0.1$	
	mean	median	mean	median	mean	median	mean	median
BOR	0.4199	0.1732	0.4430	0.1942	2.0989	2.0018	6.7815	6.9108
SDF	0.3373	0.1558	0.3228	0.1584	1.1989	1.1006	5.0831	5.1076
UDF	0.8522	0.4230	0.8953	0.4499	1.5296	1.1945	**1.9992**	1.8744
VT	**0.3222**	**0.1542**	**0.3192**	**0.1529**	**1.0256**	**0.9499**	2.4574.	2.0616
DVT	0.6290	0.3474	0.6482	0.3713	1.4792	1.1542	2.1165	**1.8623**

Table 5. Comparison between methods in terms of robustness to noise on ShapeNet [2] chairs class. σ is the standard deviation of the Gaussian noise added to the observation coordinates.

Table 5 show how different methods are affected by varying level of noise on the observations. We note that the results show a pattern similar to the completion task. This can be explained with the stronger prior on the objects shape learned by VT, DVT, and UDF.

D Property 2 proof (continued)

In this Section, we continue the proof of Property 2 from that of the main text. Specifically, we provide the proof that only the surface points are included in the level-set, Eq. (3) of the main text.

Proof. To prove the implication that only surface points are in the level set, we can identify two cases of points $\hat{\mathbf{x}}$ not belonging to the surface. We consider case I when the given point $\hat{\mathbf{x}}$ is equidistant to multiple surface points, multiple outward vectors stem from it, consequently producing a positive divergence value. We consider case II when a point $\hat{\mathbf{x}}$ has a single closest surface point. Given that the VT field is oriented towards the closest surface point from Eq. (1), following the field direction from any point, it remains constant unless it encounters a surface. Hence, the divergence amounts to a value approximately 0. A more rigorous measure can be obtained by trying to mimimize the divergence measure for the given case II. By elimination, it can be seen that the minimum divergence in case II occurs when multiple fields in the infinitesimal region around $\hat{\mathbf{x}}$ are converging to a single point as shown in Figure 7. To obtain a divergence measure we can consider the infinitesimal surface around the point $\hat{\mathbf{x}}$ drawn with green color. Note that the surface is made of spherical domes and planar sections in 3D and forms a closed volume as required by the divergence theorem. For such a closed surface the field is either parallel (spherical domes) or perpendicular (conic surface) to the field at any point. The flux through the conic surface is parallel and thus 0. The divergence for the infinitesimal closed surface and thus the point $\hat{\mathbf{x}}$ can be minimized by maximizing the difference of surface area between the spherical domes. To establish the minimum divergence, we first establish the areas of the spherical domes. The surface area of a spherical dome defined by the solid angle θ and radius r is given by,

$$S = 2\pi r^2 \left(1 - \cos\frac{\theta}{2}\right). \tag{7}$$

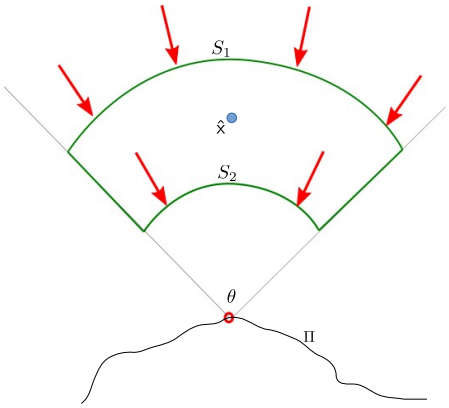

Fig. 7. Illustration of case II for the proof. We illustrate the case in order to measure the divergence around the point $\hat{x}$ which has a single closest point on the surface Π. S_1 and S_2 are two different spherical domes with radii r_1 and r_2 respectively, not to be confused with curves.

This gives the following areas for the two spherical domes:

$$\begin{aligned} S_1 &= 2\pi r_1^2 \left(1 - \cos\frac{\theta}{2}\right). \\ S_2 &= 2\pi r_2^2 \left(1 - \cos\frac{\theta}{2}\right). \end{aligned} \tag{8}$$

We now consider the proof by contradiction and assume the that the divergence measured by the infinitesimal surfaces S_1, S_2 is -2, as follows:

$$S_1 - S_2 = -2, \quad \text{s.t. } S_1, S_2 \to 0. \tag{9}$$

Eq. (9) emerges from the fact that the divergence is the difference of flux between the two domes and that the field vectors are perpendicular unit vectors on those surfaces. Thus, the divergence is the measure of the difference of the surface areas. Finally combining Eq. (8) and Eq. (9), we obtain the following condition.

$$\left(r_1^2 - r_0^2\right) = \frac{1}{\pi\left(1 - \cos\frac{\theta}{2}\right)} \quad \text{s.t. } r_1 - r_2 \to 0 \text{ and } \theta \to 0. \tag{10}$$

Evaluating both the limits of Eq. (10) leads to the contradictory result $0 = \infty$. Consolidating the theoretical results of cases I and II, the proof demonstrates the statement that a point outside the surface cannot be in the level set $L_0(g + 2)$.

E Adapted Marching Cubes Evaluation

In this section, we support the validity of the proposed modified MC algorithm. More specifically, we test the accuracy of surfaces of objects reconstructed on ground truth observations by the two MC algorithms. When applying the proposed variant, we use the DVT representation as it provides distance information. For the standard MC [24], we use the SDF observations. We use a small subset of 50 random shapes for the evaluation. Table 6 shows the performance achieved by the proposed MC algorithm when compared to a standard implementation [24]. The results show that the accuracy in both cases are the same. Therefore, we show that the proposed MC algorithm can be successfully used to retrieve high quality meshes on vector fields.

Method	mean	median
MC [24]	0.0013	0.0012
MCvector	0.0013	0.0013

Table 6. Comparison on mesh generation between a standard MC implementation [24], and the proposed MC adaptation. Both algorithms are evaluated on a subset of 50 examples from ShapeNet [2] with ground truth observations. We refer to the proposed MC algorithm as MCvector.