Supplementary Material for
Deep Marching Cubes: Learning Explicit Surface Representations

Yiyi Liao[1,2] Simon Donné[1,3] Andreas Geiger[1,4]
[1]Autonomous Vision Group, MPI for Intelligent Systems and University of Tübingen
[2]Institute of Cyber-Systems and Control, Zhejiang University
[3]imec - IPI - Ghent University [4]CVG Group, ETH Zürich
{yiyi.liao,simon.donne,andreas.geiger}@tue.mpg.de

Abstract

In this supplementary document, we first present implementation details for the experiments in the main paper. Then we show more qualitative results for estimating 3D shape from unstructured point cloud. We further demonstrate that our method is able to "lift" the binary occupancy maps to sub voxel resolution by learning to regress the 3D shape from the occupancy map.

1. Implementation Details

1.1. Deep Marching Squares

The marching cubes algorithm can be easily adapted to 2D, where the topology encodes the number and connectivity of line segments in each 2D cell. Similarly, our *Deep Marching Cubes* can also be applied in 2D, resulting in an algorithm which we refer to as *Deep Marching Squares*. Specifically, we estimate the occupancy field $\mathbf{O} \in [0,1]^{N \times N}$ and the vertex displacement field $\mathbf{X} \in [0,1]^{N \times N \times 2}$. There are $2^2 = 4$ occupancy variables corresponding to the 4 corners of a grid cell, and the number of all possible topologies is $2^4 = 16$. Again, we only consider the 14 singly connected topologies (highlighted in yellow in Fig. 1).

1.2. Network Architecture

We show the details of our Deep Marching Cubes network architecture in Fig. 2. Following common practice [1, 5], We exploit skip connections to preserve details. The decoder head is split into two branches for estimating the occupancy field $\mathbf{O}$ and the vertex displacement field $\mathbf{X}$. Note that we apply feature concatenation to these two branches individually.

1.3. Hyperparameters

The hyperparameters in our experiments are specified in Table 1. We use the Adam optimizer [3], and we train the network for 20 epochs.

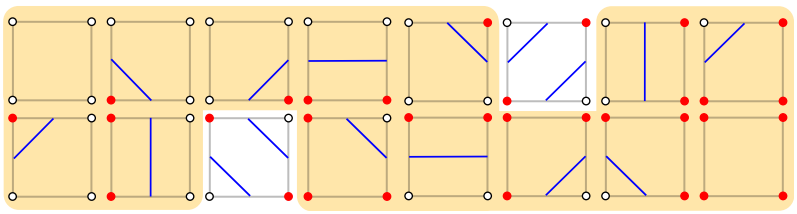


Figure 1: **Surface Topologies for the Deep Marching Squares Algorithm in 2D.** There are $2^4 = 16$ possible topologies. In this paper, we consider all singly connected topologies (highlighted in yellow).

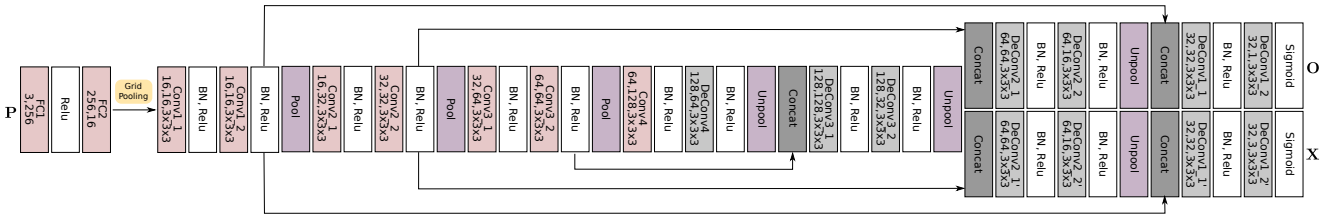


Figure 2: **Detailed Network Architecture.** We specify the *input channel*, *output channel* and *kernel size* for all convolution and deconvolution layers. We apply Batch Normalization [2] and ReLU [4] after each convolution and deconvolution layer.

Description	Notation	Value
Weight on $\mathcal{L}^{\text{mesh}}$	w_1	$1.0/N^3$
Weight on $\mathcal{L}^{\text{occ}}$	w_2	$0.4/N^3$
Weight on $\mathcal{L}^{\text{smooth}}$	w_3	$0.6/N^3$
Weight on $\mathcal{L}^{\text{curve}}$	w_4	$0.6/N^3$
Learning rate		$5e-4$
Batch size		8

Table 1: **Hyperparameters.**

	Chamfer	Accuracy	Complete.
Occupancy + MC	0.277	0.217	0.337
TSDF + MC	0.271	**0.191**	0.350
TSDF (weighted) + MC	0.276	0.200	0.352
Ours	**0.265**	0.211	**0.318**

Table 2: **3D Shape Prediction from Occupancy Maps.**

2. 3D Shape Prediction from X

In this section, we show more qualitative results for reconstructing 3D shapes from unstructured point clouds. In addition, we show that our method can be applied to other input modalities such as volumetric occupancy maps.

2.1. 3D Shape from Point Cloud

Fig. 3 illustrates more 3D shapes predicted from point cloud. For car and sofa, we rendered the objects from two views for clarity. As can be seen, our method performs better in preserving details, such as the wheels of cars (row 1-2 in (a)), the legs of the sofa (row 1-2 in (b)) and the top of the bottles (row 2-3 in (c)). Our method is also robust to objects with holes such as cars missing windows (row 3 in (a)), and objects with irregular shapes (row 4-5 in (a)). More interestingly, our method can predict the complete object shape when the car underbody is partially missing (row 6-10 in (a)) or completely missing (row 11-12 in (a)), or in the presence of thin surfaces such as in the sofa example (row 3-5 in (b)), while the baseline methods fail completely. Further note that our method not only predicts the observed part when the car underbody is completely missing, but also reconstructs a reasonable underbody, demonstrating the ability of our model to learn reasonable 3D shape models even in the presence of sparse, noisy and incomplete ground truth.

2.2. 3D Shape from Occupancy

Our approach can also be used to "lift" binary occupancy grids to sub-voxel accurate meshes. For this experiment, we replace the point encoder by a standard volumetric encoder which takes a binary occupancy map as input. We train this voxel super-resolution network using the point cloud derived from the high resolution mesh. As the input occupancy map might be incorrect when the object is not closed as shown in Fig. 3, we filtered the occupancy maps by selecting an object only when its occupied region is larger than a given threshold (2% of the full cube).

Again, We compare with the occupancy and TSDF prediction baselines. The results are shown in Table 2 and a qualitative comparison is presented in Fig. 4. Our method achieves best performance in terms of the Chamfer distance. Note that our

method reduces both accuracy and completeness distances compared to the occupancy baseline, which means we effectively learn a sub-voxel accurate mesh given the binary occupancy grids, while the TSDF baselines sacrifices completeness to achieve a higher accuracy. As can be seen from the qualitative comparison in Fig. 4, our method learns to complete the wheels (row 1-4) and fill missing windows (row 5-8) even when this information is missing from the input. Further note that our method is able to predict a complete shape even when there is a larger part missing in the input (row 9-12).

References

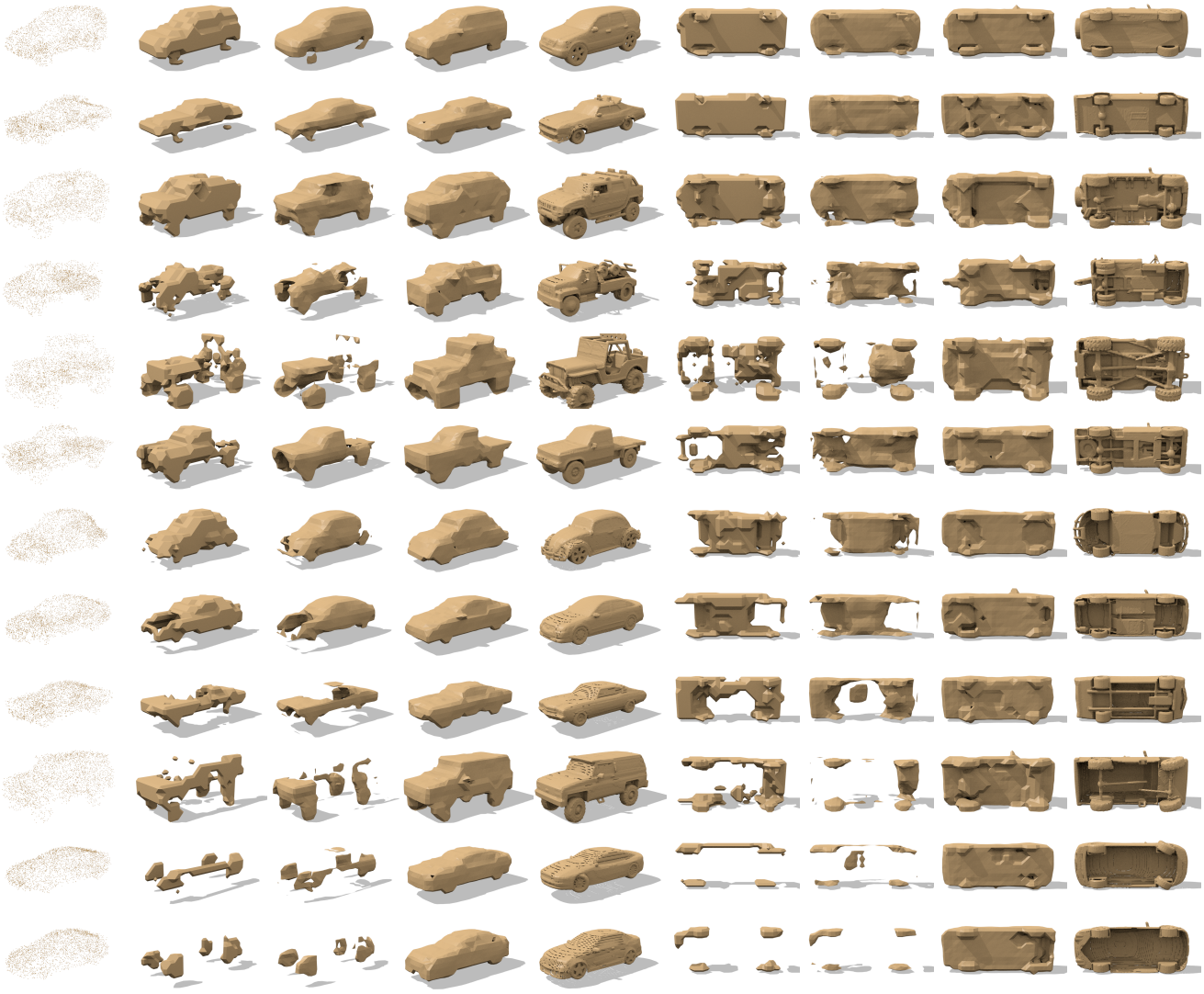
[1] A. Dosovitskiy, P. Fischer, E. Ilg, P. Haeusser, C. Hazirbas, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2015. 1

[2] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. of the International Conf. on Machine learning (ICML)*, 2015. 2

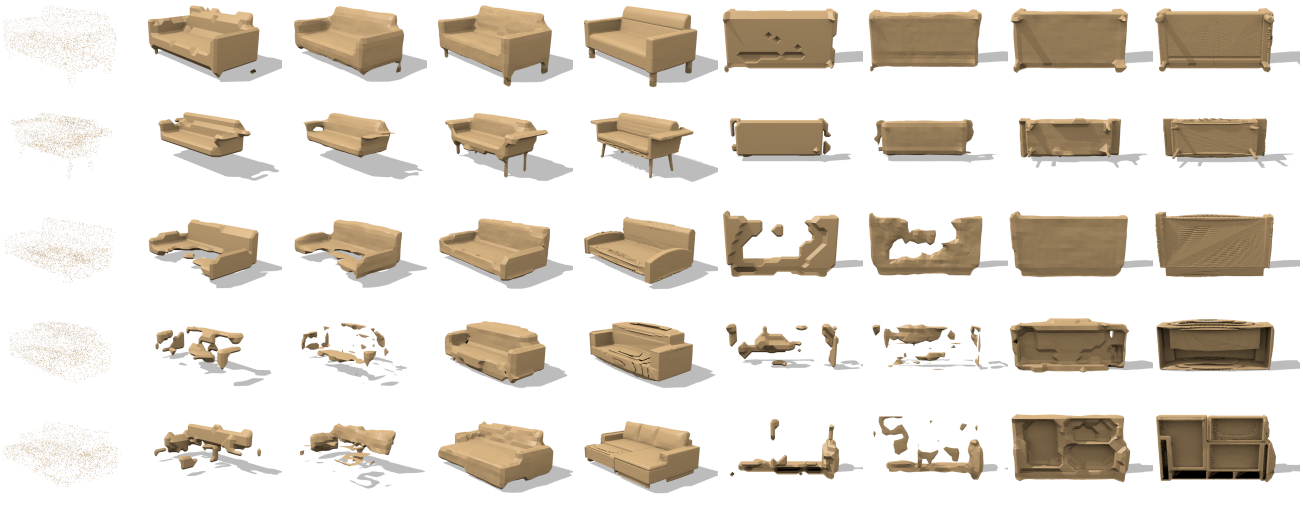
[3] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. of the International Conf. on Learning Representations (ICLR)*, 2015. 1

[4] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. of the International Conf. on Machine learning (ICML)*, 2010. 2

[5] G. Riegler, A. O. Ulusoy, and A. Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1


Input Points
Occ+MC
wTSDF+MC
Ours
Ground Truth
Occ+MC
wTSDF+MC
Ours
Ground Truth

(a) **Car**


Input Points
Occ+MC
wTSDF+MC
Ours
Ground Truth
Occ+MC
wTSDF+MC
Ours
Ground Truth

(b) **Sofa**

(c) **Bottle**

Figure 3: **3D Shape Prediction from Point Clouds.** Here, wTSDF+MC denotes the weighted TSDF baseline.

Figure 4: **3D Shape Prediction from Occupancy.**