Hierarchical Surface Prediction

Christian Häne, Shubham Tulsiani, Jitendra Malik *Fellow*

Abstract—Recently, Convolutional Neural Networks have shown promising results for 3D geometry prediction. They can make predictions from very little input data such as a single color image. A major limitation of such approaches is that they only predict a coarse resolution voxel grid, which does not capture the surface of the objects well. We propose a general framework, called hierarchical surface prediction (HSP), which facilitates prediction of high resolution voxel grids. The main insight is that it is sufficient to predict high resolution voxels around the predicted surfaces. The exterior and interior of the objects can be represented with coarse resolution voxels. This allows us to predict significantly higher resolution voxel grids around the surface, from which triangle meshes can be extracted. Additionally it allows us to predict properties such as surface color which are only defined on the surface. Our approach is not dependent on a specific input type. We show results for geometry prediction from color images and depth images. Our analysis shows that our high resolution predictions are more accurate than low resolution predictions.

Index Terms—Single View Reconstruction, High Resolution, Voxel Grid, Geometry Prediction

1 Introduction

We live in a world composed of 3D objects bounded by 2D surfaces. Fundamentally, this means that we can either represent geometry implicitly as 3D volume or explicitly as 2D mesh surface which lives within the 3D space. When working with 3D geometry in practice for many tasks a specific representation is more suited than the other.

Recently, 3D prediction approaches which directly learn a function to map an input image to the output geometry have emerged [6], [15], [47]. The function is represented as Convolutional Neural Network (CNN) and the geometry is represented as voxel grid, which is a natural choice for CNNs. The advantage of such an approach is that it can represent arbitrary topology and allows for large shape variations. However, such approaches have a major limitation. Due to the cubic growth of the volume with increasing resolution only a coarse voxel grid is predicted. A common resolution is 32^3 (cf. [6], [47]). Using higher resolutions very quickly becomes computationally infeasible due to the cubic growth of the volume. However, given that surfaces are two dimensional the growth should ideally only be quadratic. Therefore, the underlying reason why earlier approaches are restricted to fairly coarse resolution is that they do not exploit the fact that surfaces are only two dimensional. Can we build a system that does?

In this paper we introduce a general framework, hierarchical surface prediction (HSP), for high resolution 3D object reconstruction which is organized around the observation that only a few of the voxels are in the vicinity of the object's surface i.e. the boundary between free and occupied space, while most of the voxels will be "boring", either completely inside or outside the object. The basic principle therefore is to only predict voxels around the surface, which we can now afford to do at higher resolution. The key insight to enable our method to achieve this is to change the standard prediction of free and occupied space into a three label prediction with the labels *free space*, *boundary* and *occupied space*. Furthermore, we do not directly predict high resolution voxels. Instead, we hierarchically predict small blocks of voxels from coarse to fine resolution in an octree, which we call voxel block octree. Thereby we have at each resolution the signal of the *boundary* label which tells us that the descendants of the current voxel will contain both, *free space* and *occupied space*. Therefore, only nodes containing voxels with the *boundary* label assigned need higher resolution prediction. By hierarchically predicting only those voxels we build up our octree. This effectively reduces the computational cost and hence allows us to predict significantly higher resolution voxel grids than previous approaches. In our experiments we predict voxel occupancy and colors at a resolution of 256^3 (c.f. Fig. 1).

Besides the geometry we also consider surface properties, i.e. surface color. For the shape there is a well defined transformation between a surface mesh and an occupancy volume. However, this is not necessarily the case for surface property. In the case of a surface color the voxel colors for voxels which intersects the mesh can be defined but it is unclear which color a voxel far away from the surface should get assigned. Hence, the fact that our hierarchical surface prediction allows for predictions only around the surface becomes especially beneficial as the colors are only defined around the surface.

One of the fundamental questions which arises once higher resolution predictions are feasible, is whether or not a higher resolution prediction leads to more accurate results. In order to analyze this we compare our high resolution occupancy volumes to upsampled low resolution predictions in our quantitative evaluation and determine that our method outperforms these baselines. Our geometry prediction results are not only quantitatively more accurate, they are also qualitatively more detailed and have a higher surface quality. For the case of colored 3D object reconstructions we show that the predicted colors contain more details than a low resolution prediction could represent, underlying

• *C. Häne, S. Tulsiani and J. Malik are with the Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA, 94720.*
E-mail: {chaene,shubhtuls,malik}@eecs.berkeley.edu

Manuscript received May 31, 2018; revised August 05, 2018.

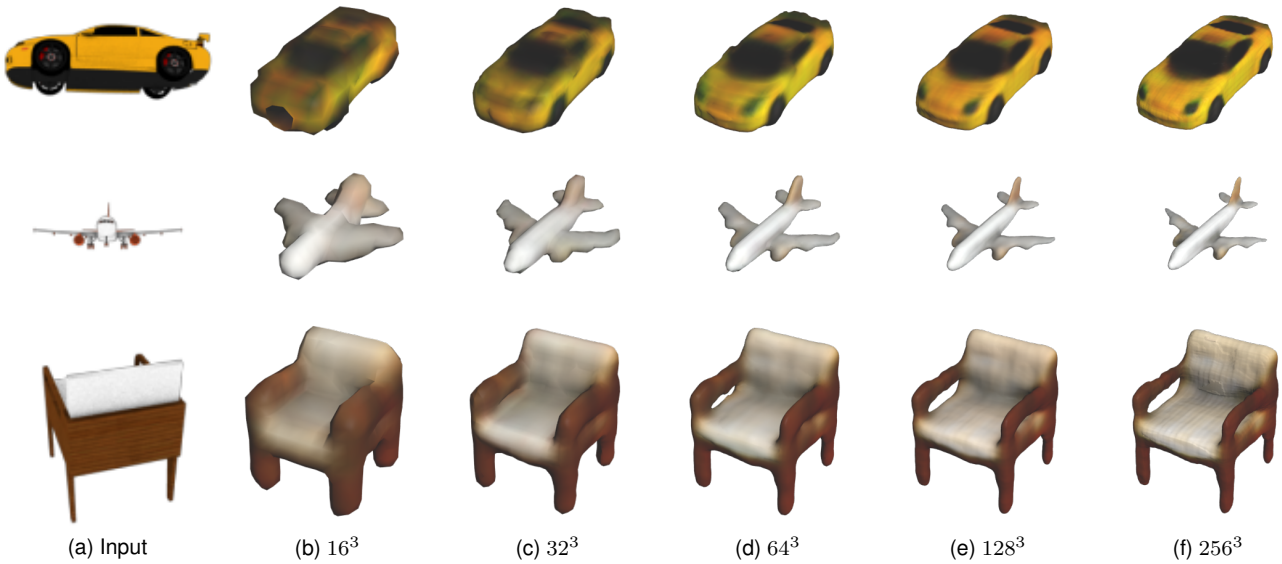


Fig. 1: Examples result form our hierarchical surface prediction framework for the task of predicting geometry and surface color from a single RGB image. Our hierarchical prediction allows us

the importance of being able to predict at high resolution.

This paper is an extension of our earlier version of the algorithm [17]. Compared to the previous version with category specific networks, in this paper we present results from networks trained on multiple categories jointly. We further extended the framework to also predict surface color. The evaluations are more extensive. The remainder of the paper is organized as follows. In Sec. 2 we discuss the related work. We then introduce our framework in Sec. 3. Quantitative and qualitative evaluations are done in Sec. 5 and eventually we draw the conclusions in Sec. 6.

2 Related Work

Traditionally dense 3D reconstruction from images has been done using a large collection of images. Geometry is extracted by dense matching or direct minimization of reprojection errors. The common methods can be broadly grouped as implicit volumetric reconstruction [7], [22], [23], [26], [51] or explicit mesh based [10], [14] approaches. All these approaches facilitate a reconstruction of arbitrary geometry. However, in general a lot of input data is required to constrain the geometry enough. In difficult cases it is not always possible to recover the geometry of an object with multi-view stereo matching. This can be due to challenging materials such as transparent and reflective surfaces or lack of images from all around an object. For such cases object shape priors have been proposed [9], [16], [49].

All the methods mentioned above in general use multiple images as input data. Approaches which are primarily designed to reconstruct 3D geometry from a single color image were also studied. Blanz and Vetter [1] propose to build a morphable shape model for faces, which allows for the reconstruction of faces from just a single image. In order to build such a model a lot of manual interaction and high quality scanning of faces is required. Other works propose to learn a shape model from silhouettes [3] or silhouettes and keypoints [20]. Silhouettes have also been utilized to model 3D objects based on the image silhouette and contour lines without utilizing a deformable shape model [31], [32], [44].

Recently, CNNs have also been used to predict geometry from single images. Choy et al. [6] propose to use an encoder decoder architecture in a recurrent network to facilitate prediction of coarse resolution voxel grids from a single or multiple color images. An alternative approach [15] proposes to first train an autoencoder on the coarse resolution voxel grids and then train a regressor which predicts the shape code from the input image. It has also been shown that CNNs can be leveraged to predict alternative primitive based representations [45]. While these approaches rely on ground truth shapes, CNNs can also be trained to predict 3D shapes using weaker supervision [34], [46], [47] for example image silhouettes. These types of supervisions utilize ray formulations which originally have been proposed for multi-view reconstruction [28], [37].

Predicting geometry from color images has also been addressed in a 2.5D setting where the goal is, given a color image to predict a depth map [12], [18], [25], [38]. These approaches can generally produce high resolution output but do only capture the geometry from a single viewpoint. Tatarchenko et al. [42] propose to predict a collection of RGBD (color and depth) images from a single color image and fuse them in a post-processing step. While in the aforementioned works the 2D space of the depth map is given by the camera, another alternative proposed in [39] is to map the 3D object to a geometry image and then predict a height map in this space. This facilitates reconstruction of objects with a single height map, is however limited to a sphere topology which many objects do not have. Also, [13] proposes to use a different output space, namely point clouds. They have the advantage that arbitrary topology can be represented. However, a point cloud does not represent a surface and hence they still need to fall back to coarse

voxel grid predictions as post processing for the evaluation. For the related problem of shape completion [8] predicts a coarse resolution signed distance field and uses a shape database at test time to upsample the coarse resolution predictions to a finer resolution.

The volumetric approach has the major limitation that using high resolution voxel grids is computationally demanding and memory intensive. Multi-view approaches alleviate this, by the use of data adaptive discretization of the volume with Delaunay tedrahedrization [24], voxel block hashing [30] or octrees [5], [41]. Our work is inspired by these earlier works on octrees. A crucial difference is that in these works the structure of the octree can be determined from the input geometry but in our case we are looking at a more general problem where the structure of the tree needs to be predicted. Similarly, a very recent work [36] which proposes to use octrees in a CNN, also assumes that the structure of the octree is given as input. We predict the structure of the tree together with its content.

Our hierarchical surface prediction framework also relates to coarse-to-fine approaches in optical flow computation. Similar to volumetric 3D reconstruction also dense optical flow is computationally demanding due the the large label space. By computing optical flow hierarchically on a spatial pyramid, e.g. [2], real-time computation of optical flow is feasible [50]. The benefits of using a coarse-to-fine approach in learning based optical flow prediction has also recently been pointed out [19], [33]. Another application where hierarchical prediction has shown to be beneficial is semantic segmentation. Li et al. [27] show performance improvements by utilizing stage wise prediction.

Concurrently to our work [35], [43] have also utilized similar ideas to predict high resolution voxel grids. While the underlying ideas are similar, these approaches use a different formulation and network architecture. Moreover the considered problems are different. Riegler et al. [35] formulate their approach for depth map fusion and predict a truncated signed distance filed. Tatarchenko et al. [43] show some results on single view 3D reconstruction but focus on autoencoders and shape interpolation.

3 Formulation

In this Section we describe our hierarchical surface prediction framework, which allows for high resolution voxel grid predictions. We first briefly discuss how the state-of-the-art, coarse resolution baseline works and then introduce the voxel block octree structure which we are predicting and detail the proposed approach for its prediction. Further, we show how our approach can be used to predict surface color and detail our efficient training procedure using subsampling.

3.1 Voxel Prediction

The basic voxel prediction framework is adapted from [6], [15]. We consider an encoder/decoder architecture which takes an input $\mathcal{I}$, which in our experiments is either a color image or a depth image. A convolutional encoder encodes the input to a feature vector which we call shape code $\mathcal{C}$. An up-convolutional decoder then decodes $\mathcal{C}$ into a predicted voxel grid $\mathcal{V}$, i.e. an occupancy probability is predicted for each voxel. In addition to the standard voxel occupancy prediction we also consider the problem where the network predicts voxel occupancies and voxel colors within the same network. A labeling of the voxel space into free and occupied space can be determined using a suitable threshold τ (c.f. Sec. 5.4.1) or alternatively a mesh can be extracted using marching cubes [29] directly on the predicted voxel occupancies at an iso value σ. In the case of color prediction the vertices of the mesh can be colored according to the predicted voxel colors using trilinear interpolation.

The main problem which prevents us from directly utilizing this formulation with high resolutions for $\mathcal{V}$ is that each voxel, even if it is far from the surface, needs to be represented and a prediction is made for it. Given that the surface area grows quadratically and the volume cubically with respect to edge division, the ratio of surface to non-surface voxels becomes smaller with increasing resolution. In the case of voxel color prediction the ground truth color is only defined on the surface which makes this an even more important problem.

3.2 Voxel Block Octree

In our hierarchical surface prediction method, we propose to predict a data structure with an up-convolutional decoder architecture, which we call "voxel block octree". It is inspired from octree formulations [5], [41] used in traditional multi-view reconstruction approaches. The key insight which allows us to use such a data structure in a prediction framework is to extend the standard two label formulation to a three label formulation with labels *inside*, *boundary* and *outside*. As we will see later our data structure allows us to generate a complete voxel grid at high resolution while only making predictions around the surface. This leads to an approach which facilitates efficient training of an encoder/decoder architecture end-to-end.

An octree is a tree data structure which is used to partition the 3D space. The root node describes the cube of interest in the 3D space. Each internal node has up to 8 child nodes which describe the subdivision of the current node's cube into the eight octants. Note that we slightly deviate from the standard definition of the octree where either none or all the 8 child nodes are present.

We consider a tree with levels $\ell \in \{1, \dots, L\}$. Each node of the tree contains a "voxel block" – a subdivision of the node's 3D space into a voxel grid of size b^3. Each voxel in the voxel block at tree levels $\ell < L$ contains classifier responses for the three labels *occupied space*, *boundary* and *free space*. The nodes in layers $\ell \in \{1, \dots, L-1\}$, therefore, contain voxel blocks $\mathcal{B}^{\ell,s} \in [0,1]^{b^3 \times 3}$ where the index s describes the position of the voxel block with respect to the global voxel grid. In the lowest level L we do not need the boundary label, therefore it is composed of nodes which contain a voxel block $\mathcal{B}^{\ell,s} \in [0,1]^{b^3}$, where each element of the block describes the classifier response for the binary classification into free or occupied space. Note that each child node describes a space which has a cube side length of only half of the current node. By keeping the voxel block resolution fixed at b^3 we also divide the voxel side length by a factor of two. This means that the prediction in the child nodes are

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2896296, IEEE Transactions on Pattern Analysis and Machine Intelligence

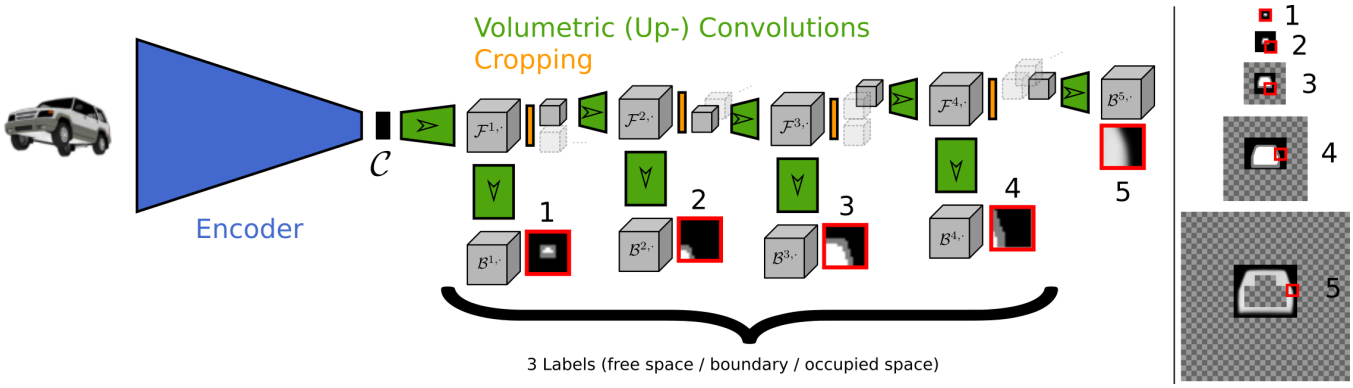


Fig. 2: Overview of our system. The input gets encoded to a shape code $\mathcal{C}$. The decoder predicts the output by hierarchically predicting all the output blocks $\mathcal{B}^{\cdot,\cdot}$. The arrows indicate the direction in which the information flows through the decoder. From each feature block $\mathcal{F}^{\cdot,\cdot}$ the octants which contain the boundary label in the corresponding output block are upsampled. Octants which need higher resolution prediction but are not part of the depicted path of the tree are visualized faded. For the inner levels of the tree three labels are predicted, free space (black), occupied space (white) and boundary (gray). On the finest level the occupancy probability is predicted. A single slice through the center of each output block is depicted. The right hand side shows how the individual voxel blocks get assembled to the final prediction for each level of the tree. The visualization shows the center slice of the volume and the red squares indicate the positions of the blocks corresponding to the depicted path. The checkered parts of the volume do not lie on the boundary and are hence not predicted.

of higher resolution. In addition to the occupancy labels we also experiment with networks which in addition predict three color channels. This is handled by adding additional output channels and is discussed in Sec. 3.4. The voxel block resolution b is chosen such that it is big enough for efficient prediction using an up-convolutional decoder network and at the same time small enough such that local predictions around the surface can be made. In our experiments we use voxel blocks with $b = 16$ and a tree with depth $L = 5$. A visualization of the voxel block octree is given in Fig. 3.

The predictions stored at any given level of the tree might be sparse. In order to reconstruct a complete model in high resolution, we upsample all the voxels which have not been predicted on the highest resolution from the closest predicted resolution. As we will see later, the voxels which get upsampled are in the inside and outside of the object, i.e. not directly next to the boundary. Therefore the resolution of the actual surface remains high. In order to be able to extract a smooth high quality surface using marching cubes it is crucial that all the voxels which are close to the surface are predicted at high resolution and the predicted values are smooth. Therefore we aim to always evaluate the highest resolution around the boundary. At this point we would like to note that we can also extract a model at intermediate resolution by considering the boundary label as occupied space. This choice is consistent with the choice of considering any voxel as occupied space which intersects the ground truth mesh during voxelization (c.f. Sec. 5.2.2).

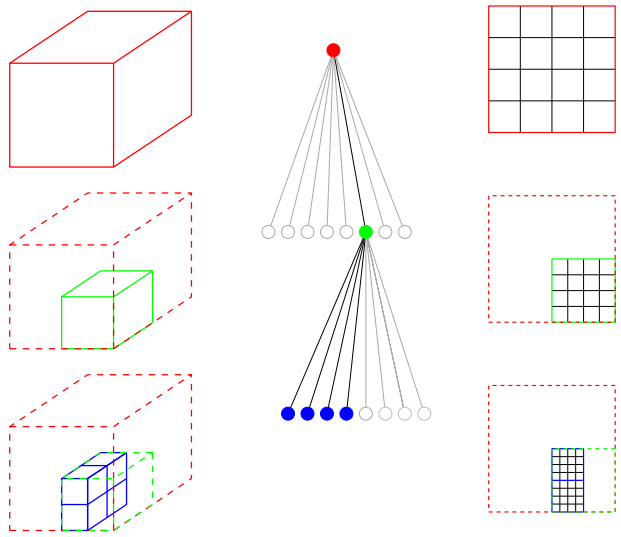


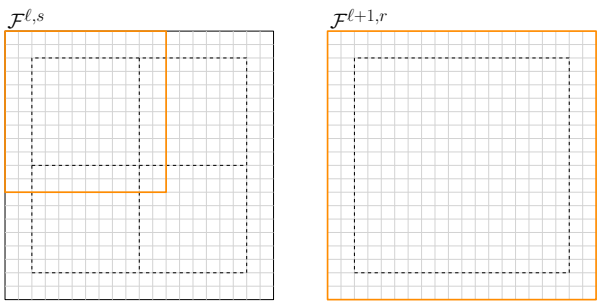
Fig. 3: Visualization of a voxel block octree with three levels. The rows correspond to the three levels of the tree and are color coded into red, green and blue. The (left) part depicts the space subdivision which is induced by the octree in the (middle). The gray nodes visualize nodes that do not belong to the tree. Each of the nodes of the octree contains a voxel block which is visualized in 2D on the (right) with a voxel block size of $b = 4$.

3.3 Network Architecture

In the previous section we introduced our voxel block octree data structure. We will now describe the encoder / decoder architecture which we are using to predict voxel block octrees using CNNs. An overview of our architecture is depicted in Fig. 2.

The encoder part is identical to the one of the basic voxel prediction pipeline from Sec. 3.1, i.e. the input $\mathcal{I}$ gets first transformed into a shape code $\mathcal{C}$ by a convolutional encoder network. An up-convolutional also called de-convolutional decoder architecture [11], [52] predicts the voxel block octree. In order to be able to predict the blocks $\mathcal{B}^{\ell,s}$ and at the same time also being able to capture the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2896296, IEEE Transactions on Pattern Analysis and Machine Intelligence



(a) Cropping (b) Upsampling

Fig. 4: 2D visualization of the cropping and upsampling module. The cropping module (left) crops out the part of the feature block centered around the child node's octant. The upsampling module (right) then upsamples the feature map using up-convolutional layers to a new feature block with higher spatial resolution. The dashed lines indicate the size of the output blocks and the octants.

information which allows for predictions of the higher resolution levels $\ell' \in \{\ell+1, \dots, L\}$, we introduce feature blocks $\mathcal{F}^{\ell,s} \in \mathbb{R}^{(b+2p)^3 \times c}$. The spatial extent of the feature blocks is bigger or equal to the ones of the voxels blocks to allow for a padding $p \geq 0$. The padding allows for an overlap when predicting neighbouring octants which leads to smoother results. The fourth dimension is the number of feature channels which can be chosen depending on the application. In our experiments we use c between 32 and 64 and $p = 2$. Detailed architectures are given in Sec. 4.

The most important part for the understanding of our decoder architecture is how we predict level $\ell + 1$ given level ℓ. In the following we will discuss this procedure in detail. It consist of three basic steps.

1) Feature Cropping
2) Upsampling
3) Output Generation

Feature Cropping. At this point we assume that we have given the feature block $\mathcal{F}^{\ell,s}$. The goal is to generate the output for the child node corresponding to a specific octant $\mathcal{O}$. The feature block $\mathcal{F}^{\ell,s}$ contains information to generate the output for all the child nodes. In order to only process the information relevant for the prediction of the child node corresponding to $\mathcal{O}$ we extract a $((b/2 + 2p)^3 \times c)$ region out of the four dimensional tensor $\mathcal{F}^{\ell,s}$ spatially centered around $\mathcal{O}$. An illustration of this process is given in Fig. 4. If neighboring octants are processed, the extracted feature channels will have some overlap, which makes the output smoother around the boundaries between the octants.

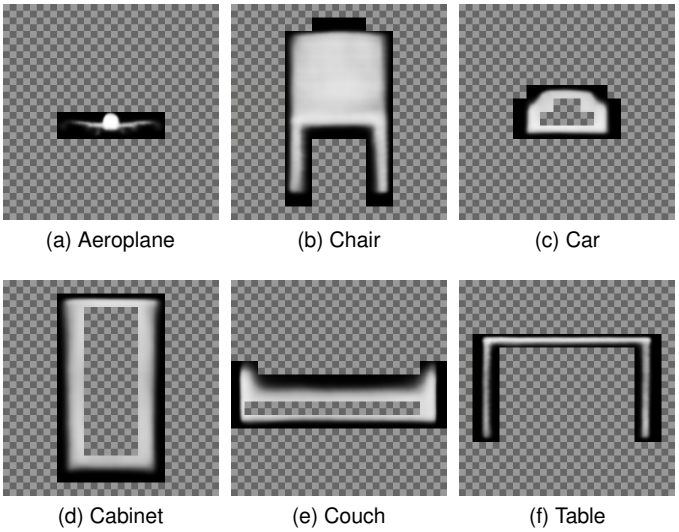
Upsampling. The upsampling module takes as input the cropped octant of block s in level ℓ. This octant is then upsampled to a block at position r in level $\ell + 1$. Therefore position r at level $\ell + 1$ is aligned with one of the octants of block s at level ℓ. The upsampling module then predicts a new $(b + 2p)^3$ feature block $\mathcal{F}^{\ell+1,r}$ via up-convolutional and convolutional layers.

Output Generation. The output network takes the prediction of the feature block $\mathcal{F}^{\ell+1,r}$ from the upsampling module and generates the voxel block $\mathcal{B}^{\ell+1,r}$. This is done using a sequence of convolutional layers. The supervision is given in form of the three ground truth labels for the voxel block $\mathcal{B}^{\ell+1,r}$, i.e. there is supervision for each level of the tree. Once the output is generated the child nodes for the next level get generated. The decision on whether to add a child node and hence a higher resolution prediction is based on the boundary prediction in the corresponding octant of the voxel block $\mathcal{B}^{\ell+1,r}$. We compute the maximum boundary prediction response of the corresponding octant $\mathcal{O}'$

$$C^{\ell+1,r}_{\mathcal{O}'} = \max_{i,j,k \in \mathcal{O}'} \mathcal{B}^{\ell+1,r}_{i,j,k,2} \ , \tag{1}$$

with label 2 the *boundary* label. The child node is generated if $C^{\ell+1,r}_{\mathcal{O}'}$ is above a small threshold γ. The intuition behind this choice is that as soon as there is some evidence that the surface is going through a specific block we should predict a higher resolution output. On the other hand if the prediction on a specific level is very certain that there is no boundary within that subtree, a higher resolution prediction is not necessary. Fig. 5 shows qualitatively that our approach effectively reduces the number of voxels that need to be predicted.

In the first level of the tree, which predicts the root node from the shape code $\mathcal{C}$, a small decoding network directly predicts the first feature block without a cropping module in between. Likewise, in the deepest level of the tree, no explicit feature block is needed. Therefore the output is directly generated from the cropped features of the previous level. Also note that the output and upsampling modules have their individual filters at each level of the tree. Furthermore, thanks to this architecture all the convolutions and up-convolutions are standard layers and no special versions for the octree are required. The detailed architecture which we used for our experiments is given in Sec. 4.1.



(a) Aeroplane (b) Chair (c) Car

(d) Cabinet (e) Couch (f) Table

Fig. 5: Predictions at the highest resolution for six different categories, the checkered areas indicate not predicted. For all the examples only the area around the boundary is predicted.

3.4 Surface Color Prediction

In this section we extend our framework to predict surface color, which we call "HSP color". This is done by adding additional output channels to each of the levels of the tree. Therefore, the dimensions of the voxel blocks are adapted. Formally, for the levels $\ell \in \{1, \dots, L-1\}$ the nodes contain voxel blocks $\mathcal{B}^{\ell,s} \in [0,1]^{b^3 \times 6}$, where three of the channels are for the geometry and three for the voxel colors. The voxel blocks in lowest level L only needs a single channel for the geometry and is hence defined as $\mathcal{B}^{\ell,s} \in [0,1]^{b^3 \times 4}$. The occupancy prediction does not depend on the voxel colors and hence the framework does not need to be adapted further.

The ground truth colors are only given for the voxels which intersect the mesh and are defined as the average color within each voxel (c.f. Sec. 5.2.2). The network predicts a color for all the voxels within each voxel block that gets predicted based on the geometry. However, the loss for the voxel color is only applied for voxels which lie on the ground truth surface. In practice this does not pose a problem the network simply learns to extrapolate the colors in the small region which is predicted around the surface. The main reason for predicting voxel colors is to be able to color the voxels of an extracted mesh, therefore a color for just the finest resolution may seem sufficient. However, in rare cases it can happen that the extracted surface is not in a region which is predicted with the finest resolution. In such a case no voxel color would be predicted if colors where only predicted for level L. To ensure that an appropriately surface color can be assigned to all meshes that can be extracted form the predicted occupancy volumes we predict voxel colors for each level of the tree.

3.5 Efficient Training with Subsampling

As we have seen above, only predicting voxels on the boundary at each level largely reduces the complexity of the prediction task (c.f. Fig. 5). In the beginning of the training, this is not the case because the boundaries are not yet predicted correctly. Moreover, even once the training has advanced enough and the boundaries are mostly placed correctly, an evaluation of the complete tree is still (too) slow for efficient training and as we will see in our experimental evaluation not necessary (see Sec. 5.4.2)

Therefore, we propose to utilize a subsampling of the child nodes during training. This is possible thanks to our hierarchical structure of the prediction. The tree gets traversed in a depth first manner. Each time the boundary label is present in an octant according to Eq. 1 the child node is traversed with a certain probability ρ. Different schedules for ρ can be used. We experimented with both, having a fixed probability ρ or start with a very low ρ and gradually increase it during training. The first version leads to a simpler training procedure and was hence used in our experiments the second version can lead to faster training once a suitable schedule is found. Thanks to the depth first traversal the memory consumption (weights and filter responses) grows linearly with the number of levels.

When training with mini-batches we have the additional difficulty that for each example a different tree needs to be evaluated. Therefore we traverse the tree for each example individually, sum up all the gradients and only do a gradient step as soon as we have done a forward and backward traversal of the subsampled tree for the training examples of the whole mini-batch.

4 Implementation Details

We implemented our system with Torch[1]. All our networks are trained using Adam [21] for stochastic optimization, with a mini-batch size of four. The number of training iterations varies depending on the task and dataset.

As loss function we use Cross-Entropy for the occupancy predictions and the mean absolute difference for the color prediction. The color ground truth is given as RGB values normalized to $[0,1]$ for the voxels on the boundary. For voxels not on the boundary we assign 0 loss. For "HSP color" both losses are used together. We multiply the mean absolute difference loss for the color prediction by 10 and add the cross entropy loss from the occupancy prediction to get the final loss.

In our method each level has its individual ground truth and hence a loss is computed for each level. Therefore we need to balance these losses with respect to each other. When traversing the tree at training time we sum up the gradients of all child nodes. Given that the octree has up to 8 child nodes we compensate this by dividing the loss by $8^{\ell-1}$. Note, for surface properies such as color the growth is only quadratical and hence the factor changes to $4^{\ell-1}$. We also compensate the influence of the subsampling by dividing the loss by $\rho^{\ell-1}$. Sometimes not all child nodes are evaluated due to the fact that they do not lie on the boundary. We chose to not get compensate this effect in the loss as we expect that in these cases the prediction is generally very confidently inside or outside and the gradients are therefore very small.

For determining whether or not a child node contains the boundary we use a threshold γ as explained in Sec. 3.1. At training time we use $\gamma = 0.04$ and additionaly we treat an octant as containing the boundary if it contains the ground truth boundary. At test time we use thresholds $\gamma \in [0.04, 0.08]$ and the ground truth label is not used. Qualitatively some noisy predictions far from the boundary can be avoided by using a slightly higher threshold γ at test time compared to training time. Quantitatively this has very little influence.

4.1 Detailed Network Architectures

In this section we give the detailed layer configurations of our networks, including the baselines. We use the following abbreviations:

Conv	Convolution
UpConv	Up-convolution
FC	Fully connected
MP	Max Pooling
R	ReLU
RS	Reshape
IN	Instance normalization

1. http://torch.ch/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2896296, IEEE Transactions on Pattern Analysis and Machine Intelligence

Type	kW	kH	sW	sH	oC	oW	oH
Input	-	-	-	-	1/3	128	128
Conv	5	5	1	1	16	128	128
MP + IN + R	2	2	2	2	16	64	64
Conv	3	3	1	1	32	64	64
MP + IN + R	2	2	2	2	32	32	32
Conv	3	3	1	1	64	32	32
MP + IN + R	2	2	2	2	64	16	16
Conv	3	3	1	1	128	16	16
MP + IN + R	2	2	2	2	128	8	8
Conv	3	3	1	1	256	8	8
MP + IN + R	2	2	2	2	256	4	4
Conv	3	3	1	1	512	4	4
MP + IN + R	2	2	2	2	512	2	2
Conv + IN	3	3	1	1	1024	2	2
RS + R	-	-	-	-	4096	1	-

TABLE 1: Color/Depth Encoder

Type	kW	kH	kD	sW	sH	sD	oC	oW	oH	oD
FC	-	-	-	-	-	-	4096	1	-	-
RS + IN + R	-	-	-	-	-	-	512	2	2	2
UpConv + IN + R	4	4	4	2	2	2	256	4	4	4
UpConv + IN + R	4	4	4	2	2	2	128	8	8	8
UpConv + R	4	4	4	2	2	2	128	16	16	16
Conv + R	3	3	3	1	1	1	64	16	16	16
UpConv + R	4	4	4	2	2	2	64	32	32	32
Conv	3	3	3	1	1	1	1	32	32	32

TABLE 2: Baseline Decoder

kW, kH, kD Kernel sizes in the three dimensions

sW, sH, sD Strides in the three dimensions

oC Number of output feature channels

oW, oH, oD Output sizes in the three dimensions

All the architectures use the same image encoder which is given in Tab. 1. The number of input channels depends on the input modality which can be either a three channel color image or a one channel depth image. The decoder of the low resolution baseline (32^3) is given in Tab. 2. The decoder of the hierarchical surface prediction for a target resolution of 256^3 is formed by stacking the decoder given in Tab. 3 and three upsampling modules given in Tab. 4. The first module and the first layer of the second module uses 64 feature channels the remainder 32 feature channels. The outputs are generated by four output modules from Tab. 5. The first two use 32 channels and the last two 16 channels. The final output for the finest resolution is computed by the module specified in Tab. 6. The number of output channels depends on whether color output is predicted or not. Coarser resolution HSP networks are formed by leaving out parts of the upsampling and output modules.

5 Experiments

We use two variants of our method "HSP" wich only predicts an occupancy volume and "HSP Color" which also predicts colors in addition to the voxel occupancies. Most of our experiments are conducted on the task of predicting high resolution occupancy volumes (HSP) from a single RGB or Depth image of an object. This is a very ambiguous task and hence we cannot expect perfect reconstruction of the objects. However as the evaluation below shows despite the difficulty of the task, using a high resolution still leads to more accurate predictions with more details recovered and a higher surface quality. Additionally, we conducted an experiment where we in addition to the occupancy volume also predict surface color (HSP Color). This experiment further show the benefits of the hierarchical surface prediction.

Type	kW	kH	kD	sW	sH	sD	oC	oW	oH	oD
FC	-	-	-	-	-	-	13824	-	-	-
RS + IN + R	-	-	-	-	-	-	512	3	3	3
UpConv + IN + R	4	4	4	2	2	2	256	6	6	6
UpConv + IN + R	4	4	4	2	2	2	128	12	12	12
UpConv + R	4	4	4	2	2	2	128	22	22	22
Conv + R	3	3	3	1	1	1	64	20	20	20

TABLE 3: Decoder module, bottleneck to feature block $\mathcal{F}^{1,1}$

Type	kW	kH	kD	sW	sH	sD	oC	oW	oH	oD
UpConv + R	4	4	4	2	2	2	64/32	22	22	22
Conv + R	3	3	3	1	1	1	64/32	20	20	20

TABLE 4: Upsampling module

5.1 Baselines

We consider two baselines. Both of the baselines predict a dense occupany volume at coarse resolution, i.e. 32^3 using the voxel prediction framework from Sec. 3.1. Both baselines use the identical network architecture. The only difference is the ground truth data with which they are trained. The first baseline follows the standard approach of labeling all voxels which intersect the ground truth mesh surface as occupied space and then fill in the interior. This can be achieved by downsampling the high resolution ground truth and label all low resolution voxels which contain at least one high resolution occupied space voxel as occupied space and all the other ones as free space. We call this baseline "Low Resolution Hard" (LR H). The second baseline uses a soft assignment for the low resolution voxels, the label is given by the fraction of high resolution occupied space voxels contained in the low resolution voxel. Therefore these labels can have fractional assignments. We call this baseline "Low Resolution Soft" (LR S). Note that the baseline LR S makes use of the high resolution voxels but the baseline LR H is equivalent to using low resoltuion voxels. As we are interested in high resolution prediction we trilinearly upsample the raw classification output of the baselines from 32^3 to 256^3 and conduct the evaluation at high resolution.

5.2 Datasets

Our experiments are conducted using the ShapeNet CAD model database [4], which is commonly used to evaluate geometry prediction networks. The whole dataset contains

Type	kW	kH	kD	sW	sH	sD	oC	oW	oH	oD
Conv + R	3	3	3	2	2	2	32/16	18	18	18
Conv	3	3	3	1	1	1	3/6	16	16	16

TABLE 5: Intermediate output module

Type	kW	kH	kD	sW	sH	sD	oC	oW	oH	oD
UpConv + R	4	4	4	2	2	2	16	18	18	18
Conv	3	3	3	1	1	1	1/4	16	16	16

TABLE 6: Full output module

57449 3D models separated into 57 categories. However, some of the categories contain very few 3D models. Therefore we follow the general practice of only using a subset of the categories. Specifically, we use the following subsets of categories.

ShapeNetCar 7497 3D models from the category Car

ShapeNet3 18320 3D models from the categories: Car, Chair, Aeroplane

ShapeNet13 43784 3D models from the categories: Car, Chair, Aeroplane, Table, Couch, Rifle, Lamp, Vessel, Bench, Speaker, Cabinet, Display, Telephone

We randomly assign 70%, 10% and 20% of the models to the train, val and test split, respectively. This is done per category such that the splits for the categories which are in multiple subsets are identical.

At this point it is important to note that most of the recent papers which used ShapeNet for 3D prediction from color images use differently sampled viewpoints and rendered images to one another. Moreover, some papers rescale the models or align them on a common ground plane. Also the volumetric ground truth is not computed the same way in different papers. All these differences can change the difficulty of the task and make the evaluations incomparable. For a clean evaluation we therefore, train our own baselines on the same data as our approach. For a direct comparison to other approaches we participated in a recently organized challenge [48], which provides rendered images and ground truth voxels at 256^3 resolution. Our algorithm is one of the two winners of the challenge.

5.2.1 Input Data

To obtain the RGB/Depth images which are used as input for training and testing our CNNs, we render the CAD models using Blender[2]. For each CAD model, we render 20 images from random viewpoints obtained via uniformly sampling azimuth from $[0, 360)$ degrees and elevation from $[-20, 30]$ degrees. We also use random lighting variations to render the RGB images.

5.2.2 Voxelization

The raw data from the ShapeNet CAD model database are textured 3D mesh models. To produce the volumetric ground truth we voxelize all the shapes in a prepossessing step. A common approach is to consider all the voxels which intersect the ground truth mesh as occupied space and then fill in the interior by labeling all voxels which are not reachable through free space from the boundary as occupied space. While this works well for the commonly used 32^3 resolution it does not directly generalize to high resolutions. The reason is that the CAD models are generally not watertight meshes and with increasing resolution the risk that a hole in the mesh prevents filling the interior increases. In our case we are voxelizing the models at a resolution of 256^3. In order to have filled interiors of the objects at high resolution we utilize a multi-scale approach. We first fill the interior at 32^3 resolution. Then erode the occupied space by one voxel. We then fix every high resolution voxel which falls within this eroded low resolution occupied space as occupied space and also fix every high resolution voxel which falls within the original low resolution free space as free space. Furthermore, we fix all the high resolution voxels which intersect the original mesh surface as occupied space. To determine the ground truth label for the remaining high resolution voxels we run a graph-cut based regularization with a small smoothness term and a preference to keep the unlabeled voxels as free space.

This procedure generates a voxelization at 256^3 resolution with filled interiors of the objects. In order to get the three label ground truth for the intermediate levels of our hierarchical prediction we first extract the boundary voxels of the occupied space at the highest resolution. Then we hierarchically compute the ground truth for the next lower resolution by assigning the boundary label to all the voxels which contain a least one boundary voxel at the next higher resolution. Our coarsest resolution is 16^3.

In order to make color predictions we also need to acquire color ground truth. To this end we unproject the rendered RGBD input images into a point cloud and determine the voxel color as average color of all the points wich unproject to that voxel. The same procedure in done for all the levels. The color ground truth is only defined for voxels which contain at least one unprojected point for the other voxels we define the loss to be 0.

5.3 Effectiveness

In this section we evaluate how effectively our approach reduces the amount of voxels that get evaluated in the high resolution volume. In order to analyze this, we compute the total number of voxels for which a prediction is made at different resolutions and compare these numbers to a dense baseline where all the voxels get evaluated. The plot shown in Fig. 6 shows that HSP is able to reduce the amount of computations and predicts only about 10% of the highest resolution voxels at test time. Note that at 32^3 resolution almost always the full volume gets predicted at test time. This is due to the fact that most of the models have parts of their surface in all the eight octants of the coarsest resolution. As indicated in Sec. 3.5 at training time we further subsample the amounts of voxels that get used. On the plot we show the amount of voxels that get evaluated with a subsampling of 30%, which we used for our experiments (c.f. 5.4.2). We also plot the amount of voxels which are actually contained in the ground truth voxel block octrees which we are predicting. It turns out that our approach only predict little more voxels than necessary.

2. http://www.blender.org

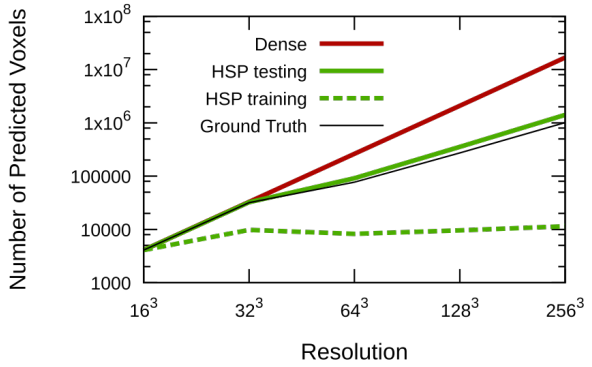


Fig. 6: Number of predicted voxels at different resolutions for a dense baseline and our hierarchical prediction. As additional reference we also plot the number of voxels the ground truth voxel block octrees contain. The numbers were computed on the dataset ShapeNet13 with RGB images as input data.

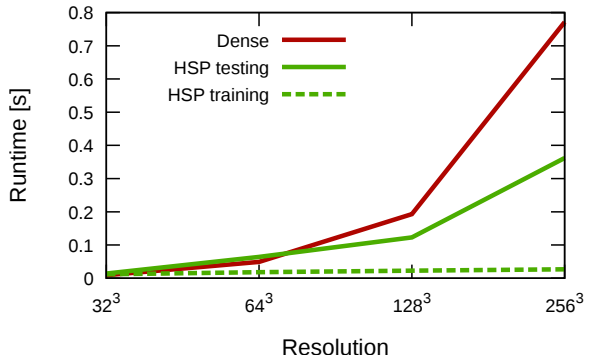


Fig. 7: Runtime of a forward pass for different resultions using an NVIDIA Quadro M6000 GPU. The numbers were computed on the ShapeNet13 dataset with RGB images as input data.

We also analyze the runtime of our method in Fig. 7. We plot the average runtime of a forward pass between our approach and a dense baseline. Desipte the significant overhead of the cropping and output modules our hierarchical surface prediction is able to make faster predictions at test time for resulutions starting at 128^3. At training time the runtime advantage of our method is much more pronounced thanks to our proposed subsampling strategy. This runtime advantage for training makes training at high resolution feasible. The runtime estimates for the dense baseline networks are measured using an untrained network. We also would like to note that our lua/torch implementation of the hierarchical surface prediction is currently does not fully utilize the GPU whereas the dense baseline fully utilizes the GPU.

5.4 Quantitative Analysis

5.4.1 Metrics

We use two metrics to measure the quality of our 3D predictions, Intersection over Union (IoU) and Chamfer Distance (CD). The IoU metric measures the volume overlap between the ground truth and predicted occupied space and the CD measures the distance between the ground truth and predicted surface. IoU is in $[0, 1]$ and higher is better and for CD lower is better. The exact definitions of the metrics used in our evaluation are:

IoU The Intersection over Union is defined as

$$\mathrm{IoU}(pred, gt) = \frac{|\,\mathrm{occ}(pred) \cap \mathrm{occ}(gt)|}{|\,\mathrm{occ}(pred) \cup \mathrm{occ}(gt)|}, \quad (2)$$

where $\mathrm{occ}(\cdot)$ returns the set of occupied voxels and $|\cdot|$ is the set cardinality.

CD We first define the asymmetric Chamfer Distance between volumes vol_1 and vol_2

$$\mathrm{CD}_{\mathrm{as}}(v_1, v_2) = \frac{1}{|\,\partial(v_1)|} \sum_{p \in \partial(v_1)} \min_{q \in \partial(v_2)} \|p - q\|_2. \quad (3)$$

$\partial(\cdot)$ returns the set of voxel centers for the voxels which lie on the boundary of the occupied space. The coordinates of the points are given in the coordinate frame of the reconstruction volume and the volume's side length is defined to be 1, which is the same coordinate frame in which the original meshes are given. The symmetric Chamfer Distance is then defined as

$$\mathrm{CD}(pred, gt) = \frac{\mathrm{CD}_{\mathrm{as}}(pred, gt) + \mathrm{CD}_{\mathrm{as}}(gt, pred)}{2}. \quad (4)$$

The definitions of the metrics are for a single 3D model. In order to get an accumulated measure for the whole dataset we compute the mean per category and take the mean of the per category values as a measure for the whole dataset.

Both metrics take binary occupancy volumes as input. However, our method and the baselines predict soft assignments, which need to be thresholded in order to compute the error metrics. The specific choice of threshold has a significant influence on the performance (see Fig. 8), which intuitively is expected as choosing fairly extreme thresholds would lead to empty reconstruction or badly inflated reconstructions. In order to have a clean evaluation which does not require a manual selection of the thresholds we use the following procedure. During training we regularly save snapshots. Subsequently we exhaustively evaluate a discrete set of thresholds on all the snapshot using the IoU metric on a subset of the validation set. This gives us the snapshot which we use for the evaluation and the threshold for the IoU metric. We then evaluate the same discrete set of thresholds using the CD metric on the a subset of the validation set. Further, we always select the voxel with the strongest score for interior as occupied to avoid the very rare case of empty 3D models for which the CD is not defined. Evaluating the CD on all the snapshots is computationally too expensive hence we use the snapshot selected with the IoU metric. We therefore end up with a single snapshot for each of the trained networks and a potentially different threshold for each of the evaluation metrics. Due to the fact that the IoU metric is a measure for the volume overlap and the CD a measure for the surface position error there is no natural way how to combine the two measures. Which measure to use might even depend on a potential application. Therefore we decided to use per metric thresholds in our evaluation. However, note that the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2896296, IEEE Transactions on Pattern Analysis and Machine Intelligence

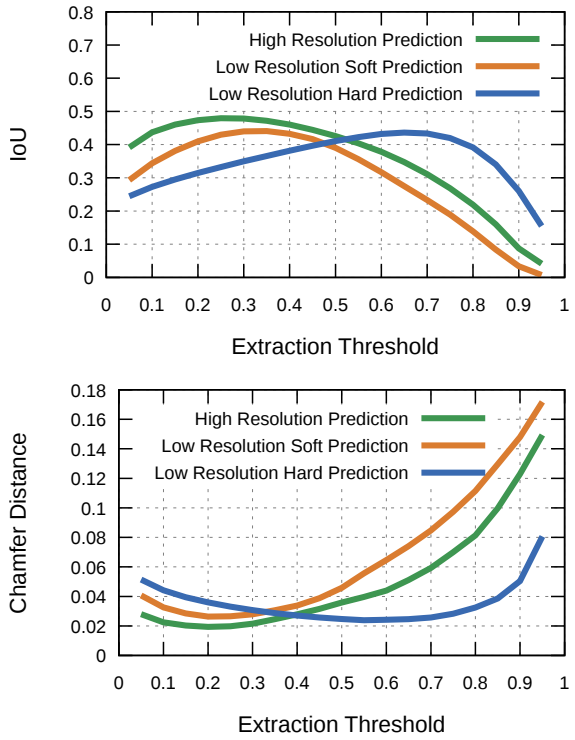


Fig. 8: Dependence of the performance on the extraction threshold. The numbers are computed on ShapeNet13 using a subset of the validation set. The task is predicting occupancy volumes from an RGB input image.

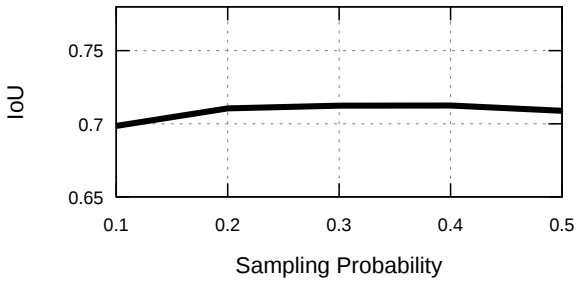


Fig. 9: Performance achieved on ShapeNetCar for the task of occupancy prediction from RGB input with different amounts of subsampling ρ.

Metric	Method	Car	Chair	Aero	Mean
IoU	LR H	0.642	0.372	0.443	0.486
	LR S	0.678	0.385	0.505	0.523
	HSP	**0.709**	**0.414**	**0.557**	**0.560**
	HSP Color	0.691	0.379	0.519	0.530
CD	LR H	0.0161	0.0229	0.0171	0.0187
	LR S	0.0189	0.0269	0.0202	0.0220
	HSP	**0.0116**	**0.0201**	**0.0131**	**0.0149**
	HSP Color	0.0121	0.0241	0.0165	0.0176

TABLE 7: Results for RGB input on the ShapeNet3 dataset.

thresholds are in general close together and determining a joint threshold which provides good results for both metrics is possible (c.f. Fig. 8).

5.4.2 Subsampling

In this experiment we investigate the influence on the performance of the classifier to the amount of subsampling of the child nodes at training time (c.f. Sec. 3.5). We conduct this experiment on the task of predicting occupancy volumes from RGB images on the ShapeNetCar dataset. We train our HSP using a constant probability ρ for using a child node which contains the boundary. To study the influence of ρ to the maximal performance that can achieve be achieved, we train the network from scratch using different choices for ρ. We train each of the methods for the same amount of time and regularly save snapshots. Subsequently we find the snapshot with maximal performance on the validation set and eventually plot the test set performance in Fig. 9. When only using 10% of the child nodes at each level the performance is lower but already after 20% the performance does not increase further. The training time to reach the maximal performance is similar for the range $[20\%, 40\%]$, hence we conducted our evaluations using 30% of the child nodes.

5.4.3 Quantitative Evaluation

In this experiment we compare the reconstruction accuracy of the our method to the baselines. In a first evaluation we only look at the task of predicting occupancy volumes from either an RGB input image or a depth input image. We train all the networks for the same number of iterations on the ShapeNet13 dataset. The metrics are computed with the method described in Sec. 5.4.1. The numbers in Tabs 8 and 9 show the benefits of using high resolution predictions. Note that the accuracy for the baselines trained on depth images are lower compared to RGB images, however for HSP results trained on depth images are better than on RGB. As this seems counter intuitive we also evaluated the cross entropy loss on the validation set and verified that consistently for predictions from depth maps the validation loss is lower for all methods.

In a second experiment we additionally evaluate the geometry prediction accuracy of HSP Color, i.e. predicting color and voxel occupancy within the same network. We conduct this experiment on the dataset ShapeNet3 with RGB input. For this experiment we train both baselines and HSP for the same number of iterations for the task of only predicting voxel occupancy. We train HSP Color for 1.5 times as many iterations as the additional color prediction needs a longer training time. The numbers in Tab. 7 shows two things. When predicting only 3 categories compared to the 13 from the experiment above the numbers are slightly better and when predicting color and occupancy within the same network the accuracy of the geometry is slightly lower.

5.5 Qualitative Analysis

In Figs 10 and 11 we show qualitative results for the task of occpuancy volume prediction on the ShapeNet13 dataset. We show a selected example for each of the categories. Qualitatively the difference is mainly in the surface quality.

Metric	Method	Car	Chair	Aero	Table	Couch	Rifle	Lamp	Vessel	Bench	Speaker	Cabinet	Display	Phone	Mean
IoU	LR H	0.624	0.389	0.411	0.349	0.556	0.383	0.232	0.437	0.277	0.511	0.547	0.377	0.604	0.438
	LR S	0.675	0.374	0.487	0.351	0.589	0.354	0.241	0.436	0.166	0.530	0.583	0.383	0.585	0.443
	HSP	**0.696**	**0.408**	**0.531**	**0.412**	**0.600**	**0.423**	**0.280**	**0.457**	**0.312**	**0.542**	**0.605**	**0.406**	**0.616**	**0.484**
CD	LR H	0.0205	0.0223	0.0199	0.0226	0.0267	0.0208	0.0417	0.0264	0.0222	0.0294	0.0220	0.0273	**0.0183**	0.0246
	LR S	0.0198	0.0288	0.0228	0.0267	0.0288	0.0213	0.0495	0.0296	0.0263	0.0340	0.0249	0.0326	0.0276	0.0287
	HSP	**0.0121**	**0.0223**	**0.0150**	**0.0195**	**0.0235**	**0.0155**	**0.0337**	**0.0227**	**0.0197**	**0.0271**	**0.0176**	**0.0270**	0.0185	**0.0211**

TABLE 8: Results for RGB input on the ShapeNet13 dataset.

Metric	Method	Car	Chair	Aero	Table	Couch	Rifle	Lamp	Vessel	Bench	Speaker	Cabinet	Display	Phone	Mean
IoU	LR H	0.589	0.370	0.386	0.320	0.542	0.355	0.226	0.416	0.223	0.495	0.537	0.365	0.556	0.414
	LR S	0.636	0.358	0.430	0.321	0.550	0.334	0.234	0.417	0.155	0.516	0.554	0.378	0.541	0.417
	HSP	**0.717**	**0.455**	**0.555**	**0.454**	**0.661**	**0.441**	**0.318**	**0.511**	**0.340**	**0.581**	**0.637**	**0.463**	**0.708**	**0.526**
CD	LR H	0.0273	0.0272	0.0249	0.0271	0.0298	0.0236	0.0436	0.0291	0.0297	0.0329	0.0276	0.0324	0.0256	0.0293
	LR S	0.0215	0.0329	0.0290	0.0294	0.0299	0.0298	0.0632	0.0323	0.0494	0.0349	0.0274	0.0329	0.0282	0.0339
	HSP	**0.0111**	**0.0192**	**0.0129**	**0.0161**	**0.0179**	**0.0149**	**0.0395**	**0.0192**	**0.0172**	**0.0235**	**0.0141**	**0.0214**	**0.0119**	**0.0184**

TABLE 9: Results for depth input on the ShapeNet13 dataset.

At low resolution with soft ground truth one of the most common issues is that thin structures do not show up in the reconstruction. This is due to the fact that these structures will already have relatively small occupancy values assigned in the ground truth due to the small fraction of high resolution occupied space voxels that will be within the low resolution voxels. For the low resolution with hard ground truth the opposite happens often thin structures are too thick which is explained by the fact that the ground truth already contains such a bias. In our high resolution reconstructions using hierarchical surface prediction we can avoid this problems thanks to the higher resolution. However, note that it can still happen that thin structures are not correctly reconstructed. One of the reasons for this is that the predicted occupancy grid also has to store the positional uncertainty of the predicted geometry and if there is two much uncertainty the activation of the occupied label falls below the extraction threshold. However, thanks to the higher resolution we are often able to extract more detail. One specific example is that in high resolution the rear view mirrors of cars are very often reconstructed.

We also evaluated how well our HSP network works on real images. To this end we run the network which was trained on ShapeNet13 on a few examples images with white background from the Online Products Dataset [40]. The results depicted in Fig. 12 show that by only training on synthetic data we are able to make high resolution predictions for real world images. The benefit of higher resolution for surface color prediction is shown in Fig. 1. Additional selected examples are depicted in Fig. 13.

6 Conclusion

We presented a framework which facilitates high resolution voxel prediction by utilizing the 2D nature of surfaces. Our approach hierarchally predidicts from coarse to fine resolution. Thereby predictions of high resolution voxels are only made around the surface which leads to an octree representation. We extensively evaluated our approach on the synthetic ShapeNet dataset on the task of predicting occupancy grids from a single color or depth image as input. Additionally we also showed that surface properties can be predicted on the example of surface color. In all our experiments we observed that HSP predicts more accurate and qualitatively better reconstructions compared to low resolution baselines.

Acknowledgments

C. Häne received funding from the Early Postdoc.Mobility fellowship No. 165245 from the Swiss National Science Foundation. The project received funding from the Intel/NSF VEC award IIS-153909 and NSF award IIS-1212798.

References

[1] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," in *Conference on Computer graphics and interactive techniques (SIGGRAPH)*, 1999.

[2] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *European conference on computer vision (ECCV)*, 2004.

[3] T. J. Cashman and A. W. Fitzgibbon, "What shape are dolphins? building 3d morphable models from 2d images," *Transactions on pattern analysis and machine intelligence (TPAMI)*, 2013.

[4] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An Information-Rich 3D Model Repository," Stanford University — Princeton University — Toyota Technological Institute at Chicago, Tech. Rep. arXiv:1512.03012 [cs.GR], 2015.

[5] J. Chen, D. Bautembach, and S. Izadi, "Scalable real-time volumetric surface reconstruction," *Transactions on Graphics (TOG)*, 2013.

[6] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3d-r2n2: A unified approach for single and multi-view 3d object reconstruction," in *European Conference on Computer Vision (ECCV)*, 2016.

[7] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Conference on Computer graphics and interactive techniques (SIGGRAPH)*, 1996.

[8] A. Dai, C. R. Qi, and M. Nießner, "Shape completion using 3d-encoder-predictor cnns and shape synthesis," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Fig. 10: Selected examples on the task of occupancy prediction form RGB and Depth input on the ShapeNet13 dataset.

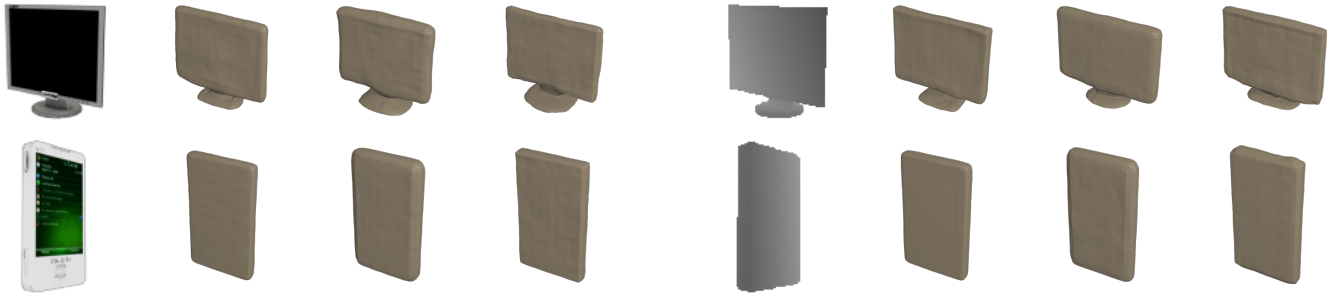


Fig. 11: Selected examples on the task of occupancy prediction form RGB and Depth input on the ShapeNet13 dataset, continued.

Fig. 12: Qualitative examples on real images from Online Products Dataset. The network was trained on the synthetic ShapeNet13 dataset and run on the real images without any finetuning or domain adaptation.

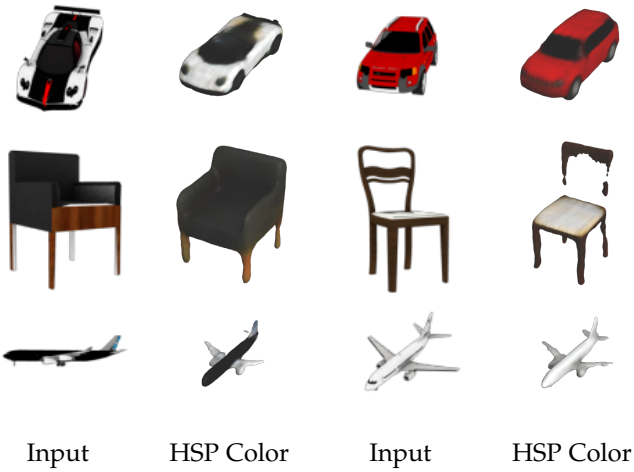


Fig. 13: Selected examples on the task of occupancy and color prediction from RGB input on the ShapeNet3 datasets.

[9] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. Reid, "Dense reconstruction using 3d object shape priors," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.
[10] A. Delaunoy, E. Prados, P. G. I. Piracés, J.-P. Pons, and P. Sturm, "Minimizing the multi-view stereo reprojection error for triangular surface meshes," in *British Machine Vision Conference (BMVC)*, 2008.
[11] A. Dosovitskiy, J. T. Springenberg, M. Tatarchenko, and T. Brox, "Learning to generate chairs, tables and cars with convolutional networks," *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 2017.
[12] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," in *Advances in neural information processing systems (NIPS)*, 2014.
[13] H. Fan, H. Su, and L. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[14] P. Gargallo, E. Prados, and P. Sturm, "Minimizing the reprojection error in surface reconstruction from images," in *International Conference on Computer Vision (ICCV)*, 2007.
[15] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta, "Learning a predictable and generative vector representation for objects," in *European Conference on Computer Vision (ECCV)*, 2016.
[16] C. Häne, N. Savinov, and M. Pollefeys, "Class specific 3d object shape priors using surface normals," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
[17] C. Häne, S. Tulsiani, and J. Malik, "Hierarchical surface prediction for 3d object reconstruction," in *Conference on 3D Vision (3DV)*, 2017.
[18] D. Hoiem, A. A. Efros, and M. Hebert, "Automatic photo pop-up," *Transactions on graphics (TOG)*, 2005.
[19] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[20] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, "Category-specific object reconstruction from a single image," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
[21] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.
[22] K. Kolev, M. Klodt, T. Brox, S. Esedoglu, and D. Cremers, "Continuous global optimization in multiview 3d reconstruction," in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, 2007.
[23] K. N. Kutulakos and S. M. Seitz, "A theory of shape by space carving," *International Journal of Computer Vision (IJCV)*, 2000.
[24] P. Labatut, J.-P. Pons, and R. Keriven, "Efficient multi-view recon-

struction of large-scale scenes using interest points, delaunay triangulation and graph cuts," in *International Conference on Computer Vision (ICCV)*, 2007.
[25] L. Ladicky, J. Shi, and M. Pollefeys, "Pulling things out of perspective," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.
[26] V. Lempitsky and Y. Boykov, "Global optimization for shape fitting," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.
[27] X. Li, Z. Liu, P. Luo, C. C. Loy, and X. Tang, "Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[28] S. Liu and D. B. Cooper, "A complete statistical inverse ray tracing approach to multi-view stereo," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.
[29] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *Siggraph computer graphics*, 1987.
[30] M. Nießner, M. Zollhöfer, S. Izadi, and M. Stamminger, "Real-time 3d reconstruction at scale using voxel hashing," *Transactions on Graphics (TOG)*, 2013.
[31] M. R. Oswald, E. Töppe, K. Kolev, and D. Cremers, "Non-parametric single view reconstruction of curved objects using convex optimization," in *Meeting of the German Association for Pattern Recognition (DAGM)*, 2009.
[32] M. Prasad and A. Fitzgibbon, "Single view reconstruction of curved surfaces," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.
[33] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[34] D. J. Rezende, S. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess, "Unsupervised learning of 3d structure from images," in *Advances In Neural Information Processing Systems (NIPS)*, 2016.
[35] G. Riegler, A. O. Ulusoy, H. Bischof, and A. Geiger, "Octnetfusion: Learning depth fusion from data," in *Conference on 3D Vision (3DV)*, 2017.
[36] G. Riegler, A. O. Ulusoys, and A. Geiger, "Octnet: Learning deep 3d representations at high resolutions," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[37] N. Savinov, C. Häne, L. Ladicky, and M. Pollefeys, "Semantic 3d reconstruction with continuous regularization and ray potentials using a visibility consistency constraint," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
[38] A. Saxena, S. Chung, and A. Ng, "Learning depth from single monocular images," in *Neural Information Processing Systems (NIPS)*, 2005.
[39] A. Sinha, A. Unmesh, Q. Huang, and K. Ramani, "Surfnet: Generating 3d shape surfaces using deep residual networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[40] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
[41] F. Steinbrücker, J. Sturm, and D. Cremers, "Volumetric 3d mapping in real-time on a cpu," in *International Conference on Robotics and Automation (ICRA)*, 2014.
[42] M. Tatarchenko, A. Dosovitskiy, and T. Brox, "Multi-view 3d models from single images with a convolutional network," in *European Conference on Computer Vision (ECCV)*, 2016.
[43] ——, "Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs," in *International Conference on Computer Vision (ICCV)*, 2017.
[44] E. Töppe, M. R. Oswald, D. Cremers, and C. Rother, "Image-based 3d modeling via cheeger sets," in *Asian Conference on Computer Vision (ACCV)*, 2010.
[45] S. Tulsiani, H. Su, L. J. Guibas, A. A. Efros, and J. Malik, "Learning shape abstractions by assembling volumetric primitives," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[46] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik, "Multi-view supervision for single-view reconstruction via differentiable ray consistency," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
[47] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee, "Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision," in *Advances in Neural Information Processing Systems*, 2016.
[48] L. Yi, H. Su, L. Shao, M. Savva *et al.*, "Large-scale 3d shape reconstruction and segmentation from shapenet core55," Tech. Rep., 2017.
[49] S. Yingze Bao, M. Chandraker, Y. Lin, and S. Savarese, "Dense object reconstruction with semantic priors," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.
[50] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime tv-l 1 optical flow," in *Joint Pattern Recognition Symposium (DAGM)*, 2007.
[51] ——, "A globally optimal algorithm for robust tv-l 1 range image integration," in *International Conference on Computer Vision (ICCV)*, 2007.
[52] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

Christian Häne is a postdoctoral scholar at the University of California, Berkeley in the Department of Electrical Engineering and Computer Sciences. Prior to that he received his Bsc, MSc and doctoral degree from ETH Zürich, Switzerland. His research interests include dense 3D reconstruction of challenging scenarios, such as semantic 3D reconstruction and 3D reconstruction from single or few images. His dissertation was awarded with the ETH medal for outstanding doctoral theses and he is the recipient of an Early Postdoc.Mobility Fellowship from the Swiss National Science Foundation.

Shubham Tulsiani received the B.Tech degree in Computer Science and Engineering from Indian Institute of Technology, Kanpur in 2013. He is currently a graduate student with the EECS Department, University of California, Berkeley. His research interests lie at the intersection of recognition, pose estimation and reconstruction from a single image. He is a recipient of the Berkeley Graduate Fellowship, CVPR Best Student Paper Award and International Physics Olympiad Gold medal.

Jitendra Malik is Arthur J. Chick Professor of EECS at UC Berkeley, and has published widely in computer vision, computer graphics and machine learning. Several well-known concepts and algorithms arose in this research, such as anisotropic diffusion, normalized cuts, high dynamic range imaging, shape contexts and R-CNN. Jitendra received the Distinguished Researcher in Computer Vision Award from IEEE, the K.S. Fu Prize from IAPR, and the Allen Newell award from ACM and AAAI. He is Fellow of ACM and IEEE and has been elected to the National Academy of Sciences, the National Academy of Engineering and the American Academy of Arts and Sciences.