Structured Prediction of Unobserved Voxels From a Single Depth Image

Michael Firman Oisin Mac Aodha Simon Julier Gabriel J. Brostow
University College London
http://visual.cs.ucl.ac.uk/pubs/depthPrediction

Abstract

Building a complete 3D model of a scene, given only a single depth image, is underconstrained. To gain a full volumetric model, one needs either multiple views, or a single view together with a library of unambiguous 3D models that will fit the shape of each individual object in the scene.

We hypothesize that objects of dissimilar semantic classes often share similar 3D shape components, enabling a limited dataset to model the shape of a wide range of objects, and hence estimate their hidden geometry. Exploring this hypothesis, we propose an algorithm that can complete the unobserved geometry of tabletop-sized objects, based on a supervised model trained on already available volumetric elements. Our model maps from a local observation in a single depth image to an estimate of the surface shape in the surrounding neighborhood. We validate our approach both qualitatively and quantitatively on a range of indoor object collections and challenging real scenes.

1. Introduction

We broadly categorize space in our world as being 'occupied' and opaque, or 'empty' and transparent. Depth cameras such as the Microsoft Kinect are able to give an estimate of which regions of a scene are composed of free, empty space. However, each pixel in a depth image only makes an estimate of occupancy in front of the first solid surface encountered along that camera ray. Occlusion prevents any information from being measured about the occupancy of space beyond that first surface.

There are many applications, however, which critically require a complete representation of the world geometry. When a robot hand or autonomous vehicle interacts with an unknown object in an unknown environment, a full 3D understanding is required to navigate and prevent collisions. In photo-editing, the full geometry would enable realistic shadows from a new light source to be automatically added to an image or stereo pair after capture.

A large amount of computer vision research has been devoted to reconstructing a full 3D world model from RGB or depth images of a scene captured from multiple viewpoints, thus coping with the effects of occlusion (e.g. [45, 28, 26]). Instead, we focus on the task of classifying each voxel in a local 3D scene as being either 'occupied' or 'empty,' given just a single depth image from one viewpoint. An example result of our algorithm is displayed in Figure 1.

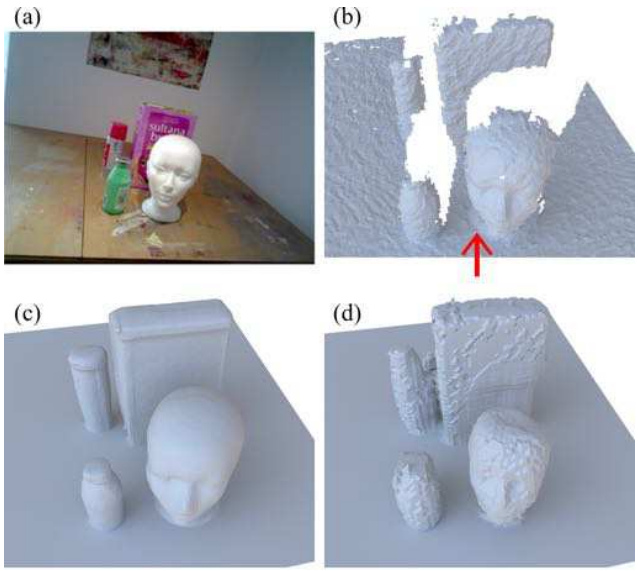


Figure 1. **Our volumetric completion.** (a) Intensity image, for illustration only. (b) (Input) 3D projection of the depth image, captured from the red arrow's perspective, where occlusions induce large empty spaces. (c) Ground truth occupancy captured using KinectFusion with multiple views. (d) (Output) Our **Voxlets** algorithm predicts a plausible completion of the occluded geometry.

In effect, we strive to predict the voxelized output of KinectFusion [28], but without moving around. We achieve this by learning a mapping from features on a depth image to a structured prediction of geometry in the region of a query point. We take inspiration from recent work that segments objects from images using silhouettes learned from different object classes [34]. They showed that shape can transcend class categories, enabling shape predictions to be made without requiring semantic understanding. As we care about shape, independent of semantic understanding, we are free to use training objects that are different from

the objects present at test time.

The key contributions that underpin our novel depth image to voxel geometry framework are:

i) *Voxlets*: a representation of local multi-voxel geometry. We use a structured Random Forest to learn a mapping from a point in a 3D reprojection of a depth image to a structured prediction of the geometry in the region around that point without requiring any semantic information.

ii) *Dataset*: we introduce both a real world dataset and a new measure for evaluating volumetric completion algorithms. The dataset contains 90 scans of different object configurations.

iii) *Fitted predictions*: we perform experiments evaluating the efficacy of different methods of selecting structured elements to use in the scene. We demonstrate that our proposed method outperforms naive alternatives.

2. Related work

Here we review related methods for completing unknown regions of visual data. While similar, we do not cover the problem of 2D image completion. Work in 2D completion usually relies on the availability of extremely large numbers of similar images [24], or on the assumption that the necessary structure for completion is present in the input data [5]. Image completion typically aims for a visually plausible output, as opposed to how well it predicts the unobserved ground truth. As we are concern with full 3D occupancy estimation we do not cover works related to 2D scene shape estimation e.g. [15, 13]. Additionally, our approach utilizes standard consumer hardware, so we do not review work that requires specialized equipment [59].

3D primitives 'Geons' were proposed by [2] as a set of primitives, such as cylinders and cuboids, used by humans in their recognition of object shapes. While in theory, geons could be used by computers as building blocks to describe natural objects, in practice, this was found to be challenging [10] due to their "idealized nature", requirement for part segmentation, labeling errors, and the coarseness of features used to extract geons in the first place. However, fitting bounding boxes has recently become a popular method to explain the arrangement of objects in a scene. Recent work has successfully incorporated high-level information such as gravity, intersection, and stability [46, 30, 29]. Other work has also made use of trained detectors [25, 55] and semantics [39] to help propose bounding box locations. Gupta et al. [20] estimate voxel occupancy from a 2D image, which is regularized using cuboid bounding box hypotheses. The obvious problem with bounding box style methods is that they can only give coarse shape information, which is ill suited for geometry completion.

Our work also makes use of 3D primitives. However, unlike geons which are fixed, we learn a flexible distribution of shape from training data, and are thus able to make higher quality predictions compared to bounding boxes.

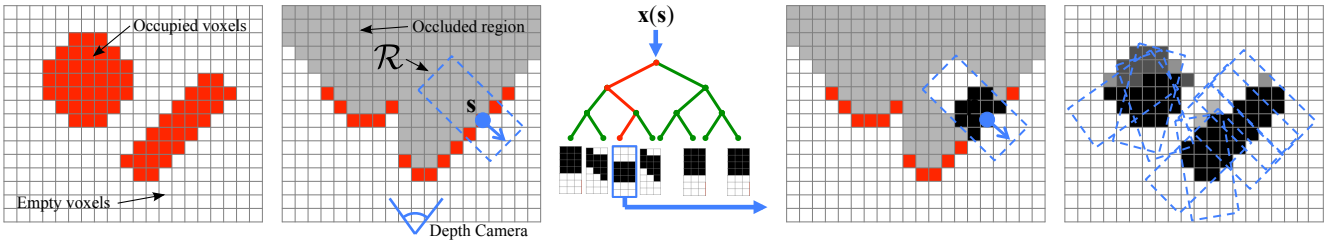
Specific shape models If prior knowledge is available, in the form of exact 3D models of all the objects present in the scene, then an instance-level model can be fitted to the observed depths. When aligned correctly, this can produce a perfect prediction of the unobserved geometry [27, 12, 3]. However, this alignment can be challenging in the presence of heavy occlusion. If an exact model of the object of interest is not present in the database, it is possible to fit objects of the same class [54, 21]. Global reasoning can be applied to find the best layout of objects, but this is still limited to the objects and primitives available in the proposal set [17]. Deformation based methods such as [4, 44] directly deform a target mesh to the observed data but can fail when an incorrect model is retrieved from the database. It is possible to apply these deformation based approaches on a part level as opposed to the whole object level [47, 57]. Generative models of 3D shape can be more expressive, but also require segmented individual objects for training [43, 60].

All of these methods rely on the availability of some form of segmented training data, and on accurate detection to localize each object or part of interest in the scene during testing. We set out to get as much shape information as possible *without semantics*, remaining free of having to accurately localize a predefined set of classes at test time.

Surface completion Silberman et al. [51] tackle the completion of an incomplete multi-view reconstruction as a 2D surface completion problem. By detecting planes, they can complete their contours in a 2D projection using a novel CRF method. However, they assume piecewise-planar scenes, and require multiple views as input. Davis et al. [9] complete surfaces by operating directly on the *signed distance field*, the zero level-set of which defines the surface location. They diffuse the signed distance field across holes in the mesh to fill in the gaps. [23] use a data-driven approach, finding matches in the mesh to fill the missing region. Symmetry can be leveraged to complete some types of objects, e.g. [38, 58, 35]. However, this can be brittle, and if symmetry is not detected, no predictions can be made.

In contrast to our approach, all of these methods are only suitable when the amount of missing data is small relative to the observed data.

Voxel space reasoning Finally, the two algorithms most similar to ours both make predictions of full scene geometry from a single depth image. Kim et al. [33] use a 'voxel CRF' model with an aim of improving 3D semantic segmentation, which they evaluate on 2D floor plans and image reprojections. For training, they use semantically labeled floor plans and images. They model the probability of a voxel being occupied as a Gaussian centered on the first observed voxel along a camera ray. Higher-order terms in the CRF are used to enforce planar structures and to encourage



(a) Ground truth scene (b) Depth occlusion (c) Forest prediction (d) Insertion into voxel grid (e) Multiple predictions

Figure 2. **2D overview of our algorithm.** (a) We model the world as a grid of voxels representing the signed distance to the nearest surface. Here, we show an overhead view of a scene featuring two objects. (b) When observed by a depth camera, only the first voxel along each ray is seen. This leaves a region of unknown occupancy extending beyond the depth surface. At test time, we define a cuboid region of voxels, $\mathcal{R}$, around each query point, $\mathbf{s}$, aligned with the normal at $\mathbf{s}$. (c) Our structured Random Forest makes a prediction for the signed distance of each of the voxels in $\mathcal{R}$ given a feature $\mathbf{x}(\mathbf{s})$ computed from the observed geometry. (d) This prediction is placed into the scene, and used to update the values of the voxels. (e) The aggregation of multiple such predictions forms our final occupancy estimate.

'objects' to remain contiguous.

Similarly, Zheng et al. [61] go from a single depth image to a voxel representation of a scene. They complete missing voxels by extruding visible points in the detected Manhattan World directions of the scene, related to the completion method of [35]. While we compare against a version of this baseline, such voxel completion by extrusion is fundamentally limited to the Manhattan World propagation of the observed volume.

Unlike [33], our algorithm does not require any semantic or appearance information. Also, in contrast to the rule-based approach of [61], we make *structured* predictions in 3D space, and reason about shape variation by learning from 3D training data.

Depth datasets While large datasets such as NYU-Depth V2 [50] and the recently introduced SUN RGB-D dataset [53] exist for single depth images, few real world datasets are available containing complete 3D reconstructions [14]. At an object level, turntable datasets exist that capture the full 360° shape of individual objects. For example, the Washington RGBD Object dataset [37, 36] contains hundreds of individual objects, but without detailed camera poses, making reconstruction difficult. The Bigbird dataset [52] is comprised of household objects along with ground truth camera poses and registered meshes. At a scene level, the few datasets featuring full reconstructions only have a limited number of examples e.g. [49]. Finally, existing synthetic datasets tend to consist of single objects in isolation [60].

In this work, we introduce a new dataset for benchmarking purposes, consisting of 90 different configurations of real objects, captured in tabletop scenarios, with complete 360° 3D reconstructions.

3. Voxlets algorithm overview

We model the geometry around an object as a regular grid of voxels $\mathcal{V} = \{v_i\}$. Following works such as [8, 28, 43], each $v_i \in [-d_{\max}, d_{\max}]$ denotes the value of the Truncated Signed Distance Function (TSDF) at that location in the volume, where the zero level-set of $\mathcal{V}$ represents a surface. Each voxel, v_i, stores the distance to the nearest surface, truncated to a maximum value of $\pm d_{\max}$. Here, v_i is negative if it is inside solid opaque matter, and positive if it is in free space. Our algorithm maps a 3D point $\mathbf{s}$, from just the observed depth image $\mathcal{D}$, to a prediction of the TSDF in a voxel neighborhood about that point. The aggregation of such predictions for multiple points in the input gives our final TSDF estimate for the scene. A 2D overview of our approach is depicted in Figure 2.

Support regions The support region $\mathcal{R} \subset \mathcal{V}$ is a set of voxels in the neighborhood of $\mathbf{s}$, for which our model can make a prediction of the TSDF. Each $\mathcal{R}$ is a fixed-size cuboid of voxels, whose x-axis is aligned with the measured normal direction at $\mathbf{s}$ (Figure 2(b)). The size of $\mathcal{R}$ is defined so that it is large enough to capture local occupancy information at an object level, but not so large that it would span the entire scene. In a 2D world, the location of $\mathbf{s}$ and the direction of its normal can unambiguously define the location and orientation of $\mathcal{R}$. In 3D however, there is an unconstrained degree of freedom, namely rotation of the cuboid about the axis of the normal. We resolve this by aligning the cuboid such that its z direction is coincident with the world z-axis, i.e. the 'up' direction of the scene. The top and bottom limits of each cuboid region $\mathcal{R}$ are therefore parallel with the world's ground plane.

Voxlets At test time, we extract a feature description for $\mathcal{R}$ from the observed geometry. Using a trained discriminative model, we can then make a prediction of the occluded geometry inside of $\mathcal{R}$. We call this prediction of geometry a *voxlet*. The voxlet, which comes out of the forest in canonical alignment, is then transformed from its local coordinate system into world space to fill the voxels in $\mathcal{R}$ (Figure 2(d)). The accumulation of multiple such predictions at different locations forms our final estimate of the full TSDF.

4. Learning a mapping from features to voxlets

We pose unobserved geometry estimation, given partial observed information, as a supervised learning problem. More specifically, our goal is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$, that maps a feature vector $\mathbf{x} \in \mathcal{X}$, computed from partially observed geometry at a point, to the output space $\mathbf{y} \in \mathcal{Y}$ representing the corresponding 3D geometry in a local region $\mathcal{R}$. Unlike standard classification, where the goal is to predict a category label for each $\mathbf{x}$, our output space is a multi-dimensional vector $\mathbf{y} \in \mathbb{R}^{w \times d \times h}$ that encodes the TSDF values in $\mathcal{R}$. The dimensionality of $\mathbf{y}$ is prohibitively large, making it difficult to use standard multivariate regression approaches, e.g. [6]. Inspired by the recent work of Dollár and Zitnick [11], we use a structured Random Forest to learn the function f.

4.1. Training

Our training set, $\{(\mathbf{x}_1, \mathbf{y}_1), ..., (\mathbf{x}_n, \mathbf{y}_n)\}$, comprises regions sampled from full 360° 3D reconstructions of objects, captured across several different scenes. To train the structured forest, we pass a bagged subset of the training set to each tree, starting at the root node. Each node is then tasked with splitting the data so that the $\mathbf{x}$'s sent to its children are as similar as possible in shape, i.e. have similar $\mathbf{y}$'s. Instead of minimizing the structured loss directly, [11] approximates this loss at each node using a classification loss. To split the data at a node, we sample a different random subset of the dimensions of each $\mathbf{y}_i$, reduce their dimensionality to M dimensions, and then cluster into two temporary classes. Then a standard classification loss can be used on this new discretization to evaluate the quality of different candidate splits for each $\mathbf{x}_i$. In practice, we efficiently perform this dimensionality reduction and clustering at each node using randomized PCA [22]. Each training example at that node is then assigned to one of the two possible clusters based on the sign of the values of its first principal component. This process is repeated until we cannot split the data any further. Finally, as in [11], each leaf node stores the medoid of all the examples that have arrived there. We refer to this as a *voxlet*. We store the medoid for efficiency reasons, but it is also possible to store multiple modes, e.g. [18].

4.2. Features

To extract a feature descriptor for a given point $\mathbf{s}$ in the scene, we first re-project the entire observed depth image $\mathcal{D}$ into 3D space using the known camera intrinsics. We create a TSDF voxel grid $\mathcal{V}_\mathcal{D}$ from these re-projected points using the method described in [28]. Our feature vector $\mathbf{x}$ is extracted directly from $\mathcal{V}_\mathcal{D}$ in the 3D neighborhood around $\mathbf{s}$. The values from $\mathcal{V}_\mathcal{D}$ at these locations form the dimensions of $\mathbf{x}$. These values can come from outside the region $\mathcal{R}$, helping to give spatial context to the prediction. We do not use appearance information, instead favoring shape cues provided by $\mathcal{D}$. These features are fast to compute and capture the surface shape in the neighborhood of $\mathbf{s}$.

In contrast to other 3D volume features e.g. [48, 7], we sample offsets from a sphere centered at $\mathbf{s}$. This sphere, of radius $r_{\max}$, is aligned to the normal and world up direction at $\mathbf{s}$ (Figure 2(b)). For computational efficiency, we sample a subset of 260 offsets within the sphere as possible candidate features.

5. Predicting occupancy at test time

Each tree in our forest makes a prediction about how the volume surrounding a point in the input depth image is occupied. Our trees perform inference very efficiently, but in practice, it is unnecessary to make a prediction densely for every location in the input, because closely neighboring locations tend to yield similar predictions. We ignore locations where the normal points away from the camera, and also reject locations that point upward (as defined by the scene's 'up' direction). We then sample a set of locations throughout the input image, spanning the spectrum of depths, to ensure uniform scene coverage. We only predict occupancy for regions about these locations. For each location in the set, we simply traverse each tree to its leaf node, and return the prediction stored there (Figure 2(c)). In Figure 3, we illustrate a few voxlets and their world positions in a real scene.

5.1. Choosing the best prediction

Each leaf node in each tree in our forest stores a voxlet, i.e. the medoid of the examples that landed at that node. For a given location in the input depth image, each tree will vote for a different voxlet. We propose three strategies to combine these region predictions from the different trees:

Forest Mean: We simply take the mean of the voxlets as the forest prediction. We note that the truncation of the signed distance function helps to make this style of accumulation robust. A single incorrect estimation at a voxel can only be wrong by a maximum amount of $2d_{\max}$, where $d_{\max}$ is the level at which the distance function is truncated.

Forest Medoid: The previous approach can produce artifacts as a result of the averaging. Selecting the medoid voxlet of all of the trees (i.e. the medoid of the medoids) results in more robustness to outliers.

Observed Fit: Neither of the previous two approaches forces predicted voxlets to be consistent with the observed geometry from the input depth image $\mathcal{D}$. To achieve this consistency, we choose a single proposal, from all the trees, that is most consistent with the observed geometry according to an error measure E. To evaluate E we first compute the 3D reprojection of the points in the input depth image, and find the subset of these points $\mathcal{P}$ that fall into the current

support region $\mathcal{R}$. We then compute the error as

$$E = \frac{1}{|\mathcal{P}|} \sum_{\mathbf{p} \in \mathcal{P}} \mathbf{y}(\mathbf{p})^2, \quad (1)$$

where $\mathbf{y}(\mathbf{p})$ is the TSDF value of that tree's prediction at location $\mathbf{p}$. This measure rewards proposals which have a level-set of zero at the same location as the observed geometry. We evaluate these three strategies in Section 7.2.

For the final prediction of the output TSDF grid $\mathcal{V}$, regardless of strategy, we average the predictions of the overlapping voxlets (Figure 2(e)). We use a *weighted* average, which assigns more weight to voxlet predictions which more closely match the observed geometry. Specifically, we weight each prediction by $\exp(-\alpha E)$. When $\alpha = 0$ the averaging is equivalent to a naive averaging with no weighting. As $\alpha \to \infty$, the most consistent voxlets get a higher weighting at the expense of less highly ranked voxlets. For all our experiments we use $\alpha = 100$. If a voxel in the output grid $\mathcal{V}$ has no predictions made for it, we mark it as empty (i.e. $d_{\max}$). Finally, marching cubes [40] is used to convert the final predicted TSDF to a mesh for visualization.

5.2. Implementation details

Given the large dimensionality of output space $\mathcal{Y}$ i.e. the size of the support region $\mathcal{R}$, we perform an initial dimensionality reduction using PCA to 400 dimensions. We empirically found this to have little impact on the quality of our results, yet it provides a large speed up at training time and reduces storage requirements. We use an ensemble of 40 trees with simple axis aligned feature tests at each node, and keep splitting while there is a minimum of 5 examples at a node, up to a maximum depth of 30. When clustering the data at each node, we set the subset of random dimensions, M, for the randomized PCA to 20. At test time, we only make predictions for a subset of $N = 300$ locations in the input depth image, sampling each point with a probability proportional to its depth. The effect of varying N is analyzed in the supplementary material. We truncate the TSDF at $d_{max} = 0.02m$. When extracting features, we set the radius of the sphere $r_{\max}$ to $0.075m$ and $0.35m$ for the tabletop and NYU-Depth V2 datasets respectively.

In our experiments, to increase coverage, we predict one of two different size voxlets (requiring two different forests). The first voxlet is centered at $\mathbf{s}$ and is longer in the y-direction, being of shape $(x \times 2x \times x)$. This is the direction that is approximately parallel to the normal at $\mathbf{s}$ (Figure 2(b)). This allows the voxlet to make a larger prediction *backwards* into the scene, compared to *sideways* which typically already has observed data. The second voxlet has shape $(x \times 2x \times 2.5x)$ and its base is fixed to the ground plane. It is more suitable for making predictions for semi occluded geometry. For a given sample location in a depth image at test time, we randomly choose one of the two forests to make a prediction. For our tabletop scenes we set $x = 0.25m$, while for room-size predictions of NYUv2 we use $x = 0.5m$. In the supplementary material we explore the effect of varying this parameter.

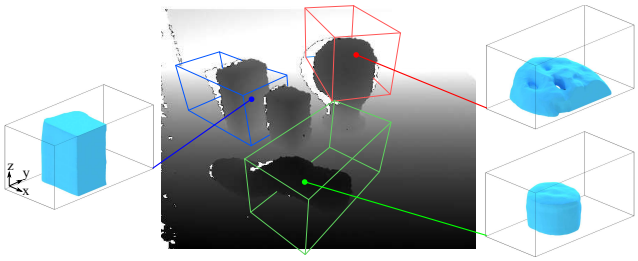


Figure 3. **Predicted voxlets.** Each tree predicts the occupancy at each sample location, in the form of a voxlet. Here we depict just three voxlets that have been meshed using marching cubes, but hundreds of predictions are made in practice.

Predicting the occupancy for a single depth image takes less than 40 seconds using our unoptimized Python implementation on a 3.60GHz processor with 4 hyperthreaded cores. Currently, the majority of the time is spent placing predicted voxlets into the output grid which could be trivially sped up with a more efficient GPU implementation.

6. Datasets

Unfortunately, existing RGBD datasets of real scenes were typically collected to evaluate semantic segmentation, object detection, or camera pose estimation. To our knowledge, no standard datasets exist that capture the full unoccluded geometry of a large number of scenes, without sections of missing data caused by occlusion. To overcome this, we introduce a new tabletop-object dataset, that we will make available to aid benchmarking of volumetric completion. Examples from this dataset are shown in Figure 5.

Our **tabletop** dataset contains the full geometry of 90 tabletop scenes, reconstructed using the KinectFusion [28] implementation of [32]. This is seven times larger than the volumetric dataset used in [61]. Each scene consists of between 2 to 6 household objects, from a set of 50, placed on a tabletop. We manually annotated the extents of the test volume for each scene. Predictions outside this domain are not used during evaluation. The dataset is split into 60 training and 30 testing scenes, captured in three different locations. The strict split ensures that no objects appear in both the training and test sets – see the supplementary material for further examples. We include the raw color and depth frames, together with the reconstructed mesh for each scene. It is worth noting that this ground truth dataset is only accurate up to the reconstruction error of [32].

The widely used **NYU-Depth V2** [50] dataset does not contain complete 3D reconstructions for each scene. However, [19] introduced a synthetic version with manually placed 3D geometry which we use for benchmarking.

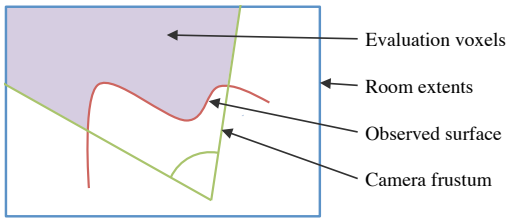


Figure 4. **Evaluation region.** For a fair comparison across all algorithms, we score on voxels that are, jointly, within the extents of the room, inside the camera frustum, and behind the surfaces visible in the input depth image.

7. Experiments

We evaluate our **Voxlets** approach using the two datasets described in the previous section. The ultimate aim of our algorithm is to accurately classify occupied vs. empty space around objects. Therefore, we report the per-voxel precision and recall over all the test data, using the sign of the accumulated TSDF as the final binary prediction of occupancy. We also report the Intersection over Union (IoU) of the predicted occupancy compared to the ground truth, which is a good measure of overall success. The evaluation region is defined as the set of voxels within the extent of the scene which are both within the camera frustum and behind the observed surface (Figure 4). The extents are tagged manually for Section 7.2, and automatically for Section 7.3. Note that we do not evaluate on the observed empty space. In both datasets the 'up' direction is extracted from the ground truth ground plane of the volume. In practice, this plane and its orientation could be detected automatically [50].

7.1. Baselines

We compare to Zheng et al. [61] as it is one of the few occupancy prediction papers that does not depend on semantics. We evaluated against their best-case idealized algorithm for reconstructing voxel occupancy, as described in Section 2 of their paper. First, the image is separated into regions using a ground-truth segmentation of the scene, and the Manhattan axes of each segment are computed using [16]. For each segment, their axis-aligned voxel search is then performed for each unobserved voxel, marking voxels as filled if more than two Manhattan directions hit a voxel directly observed by the camera. Unlike our method, their approach requires a segmentation of the scene. We use the ground truth object segmentation (only for these two baselines) to illustrate the toughest non-semantic rivals possible. We also compare to a bounding box baseline, which fits minimum volume bounding boxes to the points of each segment using Manhattan directions computed using [16].

7.2. Tabletop results

Here we perform experiments on the tabletop dataset introduced in Section 6. In Table 1 we can see that our **Voxlets** algorithm outperforms both the idealized Zheng et al. [61] method and the bounding box baseline. Qualitative results are presented in Figures 5 and 1. Despite severe occlusions and fragmentation of objects in the input depth map, we are still able to produce plausible completions. Note that we do not merge the observed geometry onto our predictions. Additional refinement to respect the observed geometry would likely improve results, but efficient inference is still an open area of research [41].

Method	IoU	Precision	Recall
Bounding Box with GT	0.445	**0.840**	0.491
Zheng et al. [61] with GT	0.528	0.773	0.630
Voxlets Observed Fit	**0.585**	0.793	**0.658**
Voxlets Forest Medoid	0.326	0.822	0.358
Voxlets Forest Mean	0.312	0.845	0.337
$\mathbf{V}_{\text{gt}}$ Ground truth voxels	0.962	0.991	0.971
$\mathbf{V}_{\text{pca}}$ GT voxels post PCA	0.908	0.977	0.927
$\mathbf{V}_{\text{nn}}$ Perfect forest	0.724	0.940	0.758
$\mathbf{V}_{\text{agg}}$ Perfect aggregation	0.701	0.897	0.766

Table 1. Quantitative results on our tabletop dataset. We also show that our final 'Observed Fit' selection strategy produces superior results compared to naive averaging and other methods e.g. [61], even when they have access to ground truth (GT) segmentation.

We compare the different strategies for selecting voxlets from our structured forest as described in Section 5.1 (see rows 3–5 of Table 1). We see that our 'Observed Fit' approach is best overall, with both better recall and IoU than other approaches. As a result of multiple conflicting overlapping predictions, 'Forest Medoid' and 'Forest Mean' tend to underpredict, resulting in higher precision but poorer recall and IoU. We favor 'Observed Fit' as it chooses the prediction that agrees most with the observed geometry at each sample point, producing better completions.

For introspection, we investigated the performance of **Voxlets** by replacing various stages with an oracle that has access to the ground truth occupancy (see Table 1):

$\mathbf{V}_{\text{gt}}$: Instead of using the structured prediction, the ground truth voxels in the local region $\mathcal{R}$ are extracted and then placed directly into the output grid. This represents what a perfectly-trained version of **Voxlets** could produce. Errors in $\mathbf{V}_{\text{gt}}$ occur for two reasons. Firstly, the proposed support regions can fail to cover some areas of the scene, hurting the recall. Secondly, quantisation effects are introduced in the extraction and re-insertion of voxel volumes.

$\mathbf{V}_{\text{pca}}$: The ground truth voxels in $\mathcal{R}$ are compressed, then decompressed, using a pre-learned PCA model. This evaluates how well PCA covers the space of voxlet shapes and shows that it does not reduce performance significantly.

$\mathbf{V}_{\text{nn}}$: We find the nearest neighbor training example that is the most similar to the ground truth voxels in $\mathcal{R}$ at each location. These are the best possible predictions, given the training set. These scores suggest that the dataset is challenging and still contains unexploited variety.

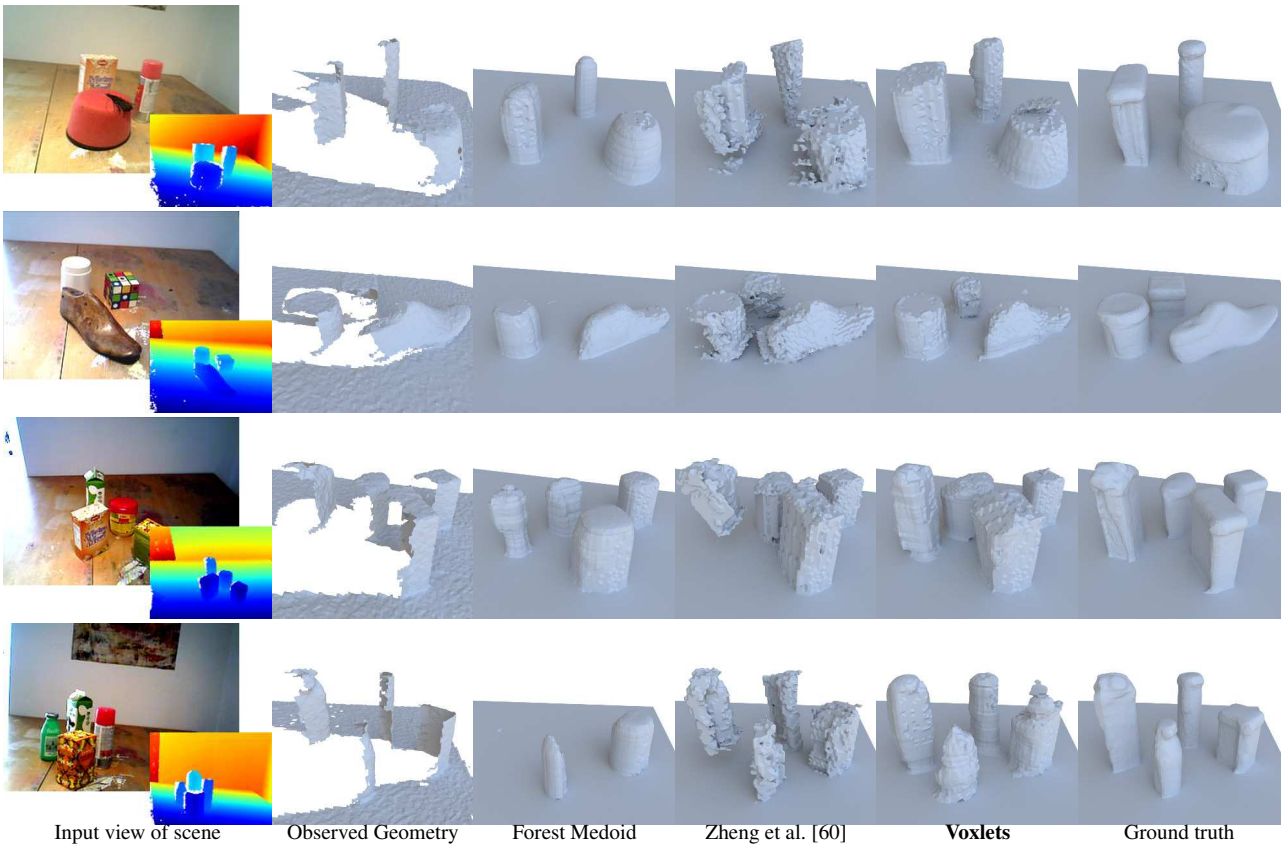


Figure 5. **Tabletop results.** Results for our tabletop dataset, where the rendered views are shown viewing in from the left of the Kinect input. For clarity, we insert a ground plane during rendering and do not superimpose the observed input geometry on top of our predictions. **Voxlets** succeeds in capturing the coarse geometry of the objects whereas Forest Medoid under predicts and Zheng et al. [61] tends to produce floating predictions. The last row shows a failure case where **Voxlets** introduces incorrect geometry on the top of the box closest to the camera as there is no observed depth due to the occlusion because of the camera baseline in the Kinect sensor.

$\mathbf{V}_{\text{agg}}$: We use our structured Random Forest with an oracle at the voxlet aggregation step. Each voxlet is greedily added to the accumulator only if its inclusion increases the score for the given scene. Results suggest that more sophisticated aggregation of voxlets could further exploit the predictions.

7.3. Synthetic NYU-Depth V2 results

In Table 2 we present results for the synthetic NYU-Depth V2 [50] dataset using the approximate but geometrically complete ground truth of [19]. We have adapted these 3D-meshed scenes to make them suitable for volumetric completion by voxelizing each one using [1], and rendering a depth image from the same viewpoint as the original Kinect camera location. We randomly assign 500 scenes from the official training set for training and 200 scenes from the test set for testing. For the method of Zheng et al. [61] we compute a local coordinate frame using the ground truth segmentation for each separate object. Despite this advantage, **Voxlets** produces superior completions. Images of these results can be seen in the supplementary material.

Method	**IoU**	**Precision**	**Recall**
Zheng et al. [61] with GT	0.346	0.601	0.467
Bounding boxes with GT	0.349	**0.693**	0.447
Voxlets	**0.508**	0.665	**0.697**

Table 2. Quantitative results for the synthetic NYU-Depth V2 dataset of [19]. All methods use single depth maps generated from the synthetic geometry as input, but the baselines use the ground truth (GT) object level segmentation to aid prediction.

7.4. Qualitative NYU-Depth V2 results

While **Voxlets** has been designed for tabletop scenes, we show here qualitative results on the challenging NYU-Depth V2 [50] dataset. We use our model trained on the synthetic dataset of [19] from the previous section. Results must be inspected by eye, because quantitatively evaluating synthetic ground truth vs. predictions made from real Kinect depth input images is not possible: the alignment between the real depth and the manually created ground truth is inaccurate. For completeness, we also compare to the methods of [39, 17] which utilize additional cues in the form of appearance and semantic classifiers. These meth-

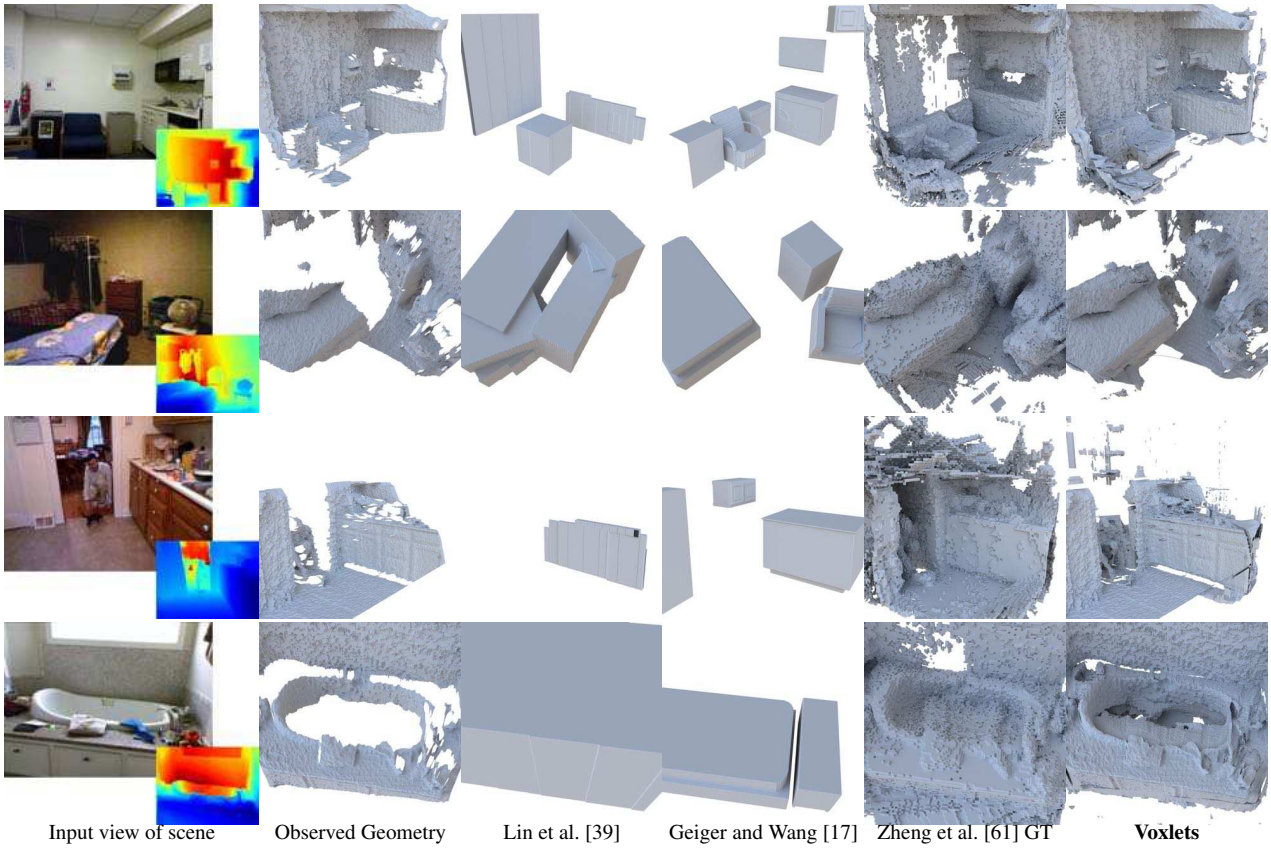


Figure 6. **NYU-Depth V2 results.** Here we qualitatively compare different occupancy predictions methods using real Kinect depth images from the NYU-Depth V2 [50] dataset. Unlike [39] and [17], our **Voxlets** algorithm does not require any appearance information. All results are rendered from the same viewpoint, and for [39] and [17] we do not show the predicted walls and floor.

ods were not designed specifically for occupancy estimation. For Zheng et al. [61], we use the ground truth object segmentation masks provided by [50].

Limitations As a supervised learning algorithm, **Voxlets** is limited by the data available at training time. Our voxlets are a fixed size, and success correlates with the test-scene having similar sized objects. Holes in the observed depth images at object boundaries can also cause problems e.g. the spiky top in the last row of Figure 5, and sharp edges can sometimes be rounded due to aggregation.

8. Conclusions and future work

We have demonstrated that **Voxlets** can successfully recover 3D geometry using only a single input depth image. Our supervised algorithm efficiently combines both selection and pose estimation of local shapes, using simple feature test evaluations to predict local geometry occupancy. We have shown that objects from distinct semantic classes share enough 3D shape components to allow plausible, though not perfect, reconstructions. Though intended for tabletop objects, our results on indoor scenes are on par with more constrained algorithms.

For some applications, the quality of our predictions may already be enough, e.g. to aid robot grasping [56, 62] or navigation. It is not guaranteed that our results are physically stable, and how to best incorporate physics-based reasoning [61, 46] is still an open problem, but enforcing this prior may improve accuracy. One interesting potential application of our method is to use the predicted completion as a prior for SLAM. As new data arrives, a next-best-view algorithm [42, 31] could leverage our predictions to guide the camera to a position which captures the geometry most likely to be informative for verifying our proposals.

Acknowledgements We wish to thank the anonymous reviewers and Neill Campbell, Peter Gehler, and the vision group at UCL for their valuable comments and suggestions. The authors are supported by EPSRC projects EP/K015664/1 and EP/I031170/1, and EU project CR-PLAY (no 611089). Michael Firman was also supported by an EPSRC Doctoral Training Grant.

References

[1] Binvox. *www.cs.princeton.edu/∼min/binvox*, Accessed Nov 2015.

[2] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 1987.

[3] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6D object pose estimation using 3D object coordinates. In *ECCV*, 2014.

[4] T. T. Cocias, F. Moldoveanu, and S. M. Grigorescu. Generic fitted primitives (GFP): Towards full object volumetric reconstruction for service robotics. In *Computer Graphics, Visualization and Computer Vision*, 2013.

[5] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *CVPR*, 2003.

[6] A. Criminisi and J. Shotton. Decision forests for computer vision and medical image analysis. Springer Science, 2013.

[7] A. Criminisi, J. Shotton, and S. Bucciarelli. Decision forests with long-range spatial context for organ localization in CT volumes. In *MICCAI*, 2009.

[8] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Computer graphics and interactive techniques*, 1996.

[9] J. Davis, S. Marschner, M. Garr, and M. Levoy. Filling holes in complex surfaces using volumetric diffusion. In *3DPVT*, 2002.

[10] S. J. Dickinson, R. Bergevin, I. Biederman, J.-O. Eklundh, R. Munck-Fairwood, A. K. Jain, and A. Pentland. Panel report: The potential of geons for generic 3-D recognition. *Image and Vision Computing*, 1997.

[11] P. Dollár and C. L. Zitnick. Fast edge detection using structured forests. *PAMI*, 2015.

[12] B. Drost and S. Ilic. 3D object detection and localization using multimodal point pair features. In *3DIMPVT*, 2012.

[13] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014.

[14] M. Firman. RGBD datasets: Past, present and future. *arXiv:1604.00999*, 2016.

[15] D. F. Fouhey, A. Gupta, and M. Hebert. Data-driven 3D primitives for single image understanding. In *ICCV*, 2013.

[16] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Manhattan-world stereo. In *CVPR*, 2009.

[17] A. Geiger and C. Wang. Joint 3D Object and Layout Inference from a single RGB-D Image. In *German Conference on Pattern Recognition*, 2015.

[18] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *ICCV*, 2011.

[19] R. Guo and D. Hoiem. Support surface prediction in indoor scenes. In *ICCV*, 2013.

[20] A. Gupta, S. Satkin, A. A. Efros, and M. Hebert. From 3D scene geometry to human workspace. In *CVPR*, 2011.

[21] S. Gupta, P. A. Arbeláez, R. B. Girshick, and J. Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *CVPR*, 2015.

[22] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 2011.

[23] G. Harary and A. Tal. Context-based coherent surface completion. *ACM Transactions on Graphics*, 2013.

[24] J. Hays and A. A. Efros. Scene completion using millions of photographs. *SIGGRAPH*, 2007.

[25] V. Hedau, D. Hoiem, and D. Forsyth. Recovering free space of indoor scenes from a single image. In *CVPR*, 2012.

[26] A. Hermans, G. Floros, and B. Leibe. Dense 3D semantic mapping of indoor scenes from RGB-D images. In *ICRA*, 2014.

[27] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In *ACCV*, 2012.

[28] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *UIST*, 2011.

[29] Z. Jia, A. Gallagher, A. Saxena, and T. Chen. 3D reasoning from blocks to stability. *PAMI*, 2015.

[30] H. Jiang and J. Xiao. A linear approach to matching cuboids in RGBD images. In *CVPR*, 2013.

[31] E. Johns, S. Leutenegger, and A. J. Davison. Pairwise decomposition of image sequences for active multi-view recognition. In *CVPR*, 2016.

[32] O. Kahler, V. A. Prisacariu, C. Y. Ren, X. Sun, P. H. S. Torr, and D. W. Murray. Very high frame rate volumetric integration of depth images on mobile devices. *ISMAR*, 2015.

[33] B. Kim, P. Kohli, and S. Savarese. 3D scene understanding by voxel-CRF. In *ICCV*, 2013.

[34] J. Kim and K. Grauman. Shape sharing for object segmentation. In *ECCV*, 2012.

[35] O. Kroemer, H. B. Amor, M. Ewerton, and J. Peters. Point cloud completion using extrusions. In *International Conference on Humanoid Robots*, 2012.

[36] K. Lai, L. Bo, and D. Fox. Unsupervised feature learning for 3D scene labeling. In *ICRA*, 2014.

[37] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *ICRA*, 2011.

[38] A. J. Law and D. G. Aliaga. Single viewpoint model completion of symmetric objects for digital inspection. *CVIU*, 2010.

[39] D. Lin, S. Fidler, and R. Urtasun. Holistic scene understanding for 3D object detection with RGBD cameras. In *ICCV*, 2013.

[40] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *SIGGRAPH*, 1987.

[41] M. R. Oswald, J. Stühmer, and D. Cremers. Generalized connectivity constraints for spatio-temporal 3D reconstruction. In *ECCV*, 2014.

[42] C. Potthast and G. S. Sukhatme. A probabilistic framework for next best view estimation in a cluttered environment. *Visual Communication and Image Representation*, 2014.

[43] V. A. Prisacariu and I. Reid. Shared shape spaces. In *ICCV*, 2011.

[44] J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin, and D. Hoiem. Completing 3D object shape from one depth image. In *CVPR*, 2015.

[45] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, 2006.

[46] T. Shao, A. Monszpart, Y. Zheng, B. Koo, W. Xu, K. Zhou, and N. J. Mitra. Imagining the unseen: Stability-based cuboid arrangement for scene understanding. In *SIGGRAPH Asia*, 2014.

[47] C.-H. Shen, H. Fu, K. Chen, and S.-M. Hu. Structure recovery by part assembly. In *SIGGRAPH Asia*, 2012.

[48] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *CVPR*, 2011.

[49] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *CVPR*, 2013.

[50] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012.

[51] N. Silberman, L. Shapira, R. Gal, and P. Kohli. A contour completion model for augmenting surface reconstructions. In *ECCV*, 2014.

[52] A. Singh, J. Sha, K. Narayan, T. Achim, and P. Abbeel. BigBIRD: A large-scale 3D database of object instances. In *ICRA*, 2014.

[53] S. Song, S. Lichtenberg, and J. Xiao. SUN RGB-D: A RGB-D scene understanding benchmark suite. In *CVPR*, 2015.

[54] S. Song and J. Xiao. Sliding shapes for 3D object detection in depth images. In *ECCV*, 2014.

[55] S. Song and J. Xiao. Deep sliding shapes for amodal 3D object detection in RGB-D images. In *CVPR*, 2016.

[56] Z. Sui, O. C. Jenkins, and K. Desingh. Axiomatic particle filtering for goal-directed robotic manipulation. In *IROS*, 2015.

[57] M. Sung, V. G. Kim, R. Angst, and L. Guibas. Data-driven structural priors for shape completion. In *SIGGRAPH Asia*, 2015.

[58] S. Thrun and B. Wegbreit. Shape from symmetry. In *ICCV*, 2005.

[59] A. Velten, T. Willwacher, O. Gupta, A. Veeraraghavan, M. G. Bawendi, and R. Raskar. Recovering three-dimensional shape around a corner using ultra-fast time-of-flight imaging. *Nature Communications*, 2012.

[60] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D ShapeNets: A deep representation for volumetric shape modeling. In *CVPR*, 2015.

[61] B. Zheng, Y. Zhao, J. C. Yu, K. Ikeuchi, and S.-C. Zhu. Beyond point clouds: Scene understanding by reasoning geometry and physics. In *CVPR*, 2013.

[62] M. Zhu, K. G. Derpanis, Y. Yang, S. Brahmbhatt, M. Zhang, C. Phillips, M. Lecce, and K. Daniilidis. Single image 3D object detection and pose estimation for grasping. In *ICRA*, 2014.