Completing 3D Object Shape from One Depth Image

Jason Rock Tanmay Gupta Justin Thorsen JunYoung Gwak Daeyun Shin
Derek Hoiem
University of Illinois at Urbana-Champaign
{jjrock2, tgupta6, thorsen1, gwak2, dshin11, dhoiem} @illinois.edu
vision.cs.illinois.edu/projects/shaperecon

Abstract

Our goal is to recover a complete 3D model from a depth image of an object. Existing approaches rely on user interaction or apply to a limited class of objects, such as chairs. We aim to fully automatically reconstruct a 3D model from any category. We take an exemplar-based approach: retrieve similar objects in a database of 3D models using view-based matching and transfer the symmetries and surfaces from retrieved models. We investigate completion of 3D models in three cases: novel view (model in database); novel model (models for other objects of the same category in database); and novel category (no models from the category in database).

1. Introduction

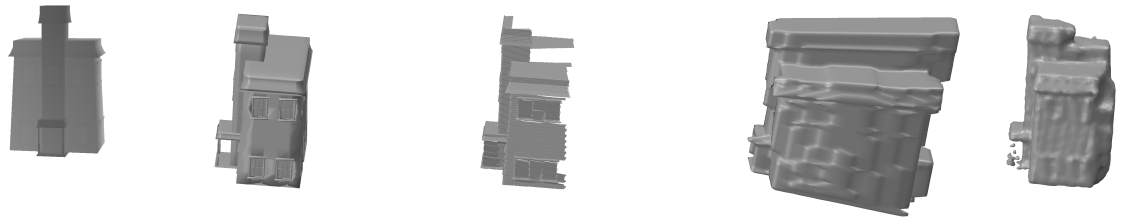
Consider the building in Figure 1. Although we only see a small portion of the object, we can accurately predict the entire shape. This ability to infer complete 3D shape from a single view is important for grasping, as we often reach around an object to grasp its unseen surfaces. Likewise, shape provides cues to category, affordance, and other properties. Recovery of 3D shape from a depth image is also useful for content creation and augmented reality applications. But how do we guess the shape of unseen surfaces? One approach is to recognize the same object or a similar object from past experience: the hidden surfaces of a favorite coffee mug can be inferred from earlier views or handling. Another is to infer missing surfaces using symmetries to duplicate and transform observed surfaces.

In this paper, we combine these strategies. Given a depth image, we find similar 3D models in similar viewpoints using view-specific matching of observed surfaces. The retrieved models are deformed to better approximate the visible portion of the query object. Deformations are constrained by smoothness and symmetries recovered from the 3D models to maintain a plausible shape. In some cases, such as when the instance or category is known, the deformed model will fit the observed surface, and hidden surfaces can be transferred from the model. In many cases, however, the deformed model does not fit the details of the observed surfaces, yet the query and retrieved object have similar symmetries. Thus, rather than transferring surfaces from the retrieved model, we instead transfer local symmetries to complete missing surfaces.

1.1. Related Work

Most previous approaches in shape completion require user interaction (e.g. [34, 33, 10, 40, 4]) or are restricted to a limited set of objects or categories (e.g., [29, 2, 23, 32], often chairs, as models are plentiful, and silhouettes informative. Our paper is unusual in its focus on quantitatively accurate reconstruction for a broad range of objects, rather than visually pleasing reconstruction for graphics content generation. We are the first to include a broad quantitative analysis of automatic 3D shape reconstruction of known objects from novel viewpoints, novel objects from known categories, and novel objects from unknown categories. Our approach includes techniques for viewpoint-based shape matching, 3D deformation, 3D mesh analysis, and 3D model synthesis that draw from or relate to a rich literature in graphics and computer vision.

Matching aims to find a complete 3D model from the library that has a similar shape to the depth-imaged object. A common approach, which we adopt, is to render each object from the library of meshes from many viewpoints and find a depth image that matches the query depth image to recover a similar 3D model from a similar viewpoint. For example, Ohbuchi et al. [28] propose an approach for retrieval based on SIFT applied to depth images, which is used by Goldfeder et al. [11] to retrieve models in order to transfer grasp points. Wohlkinger and Vincze [37] describe a variety of matching features and an algorithm for partial-view to partial-view matching. Hinterstoisser et al. [12] perform template matching to localize objects in videos of cluttered scenes. Tejani et al. [35] propose a tree based structured prediction for determining object and pose from RGB-D



(a) Query Depthmap (b) Query Mesh (Ground Truth) (c) Pointcloud Mesh (d) Reconstruct Baseline (e) Ours

Figure 1: While depth images and point clouds appear to have enough information for us to reconstruct, we are implicitly applying our intuition about what objects look like. We recognize that this building is wider than it is long, and that the chimney does not extend the whole length, even though that information is not directly captured in the depthmap. (d) shows a reconstruction using a simple reflection about the image plane at maximum depth. (e) shows our reconstruction result and conforms to our expectation of the object better.

images. Wang et al. [36] retrieve CAD models that are similar to a noisy 3D point cloud scan using local point descriptors and Regression Tree Fields. We experimented with many matching techniques but found that using Random Forests [3] to hash based on similarity and depth ordering of pixels provides a simple, fast, and effective retrieval mechanism.

Deformation aims to geometrically transform the retrieved mesh so that part of its surface aligns well with the observed query depth image. The key problems are finding local correspondences and defining a regularized deformation model that has sufficient flexibility but discourages unlikely shapes. Motivated in part by work in graphics on image-based modeling (e.g., Chen et al. [5]), we believe that symmetry-constrained deformations provide the right compromise between rigidity and arbitrary deformation. Our method for finding symmetries in the retrieved mesh is based on [26]. We use the Spin Image Descriptor [15] to find long-range correspondences for coarse, global (similarity) transformation, followed by nearest-point matching for symmetry-constrained, thin plate spline deformation. Geometrically-constrained thin-plate spline models have been applied in various fields such as 2D graphic modeling [8] and medical imaging [1]. Kurz et al. [19] explore a generic constrained optimization framework for symmetry-preserving deformation that minimizes an energy function in a grid of B-spline basis functions. Other techniques that may be applicable include Sorkine and Alexa's As-Rigid-As-Possible deformation [33] and the cuboid-based deformations of Zheng et al. [40].

Completing a 3D model from an RGB image or depth image can be attempted by fitting a parameterized model based on contours, shape priors, and detected symmetries or by deforming an exemplar mesh. Contour-based methods often have limited applicability to specific classes, such as generalized cylinders (e.g., [24, 30]) or blocky structures [40], or require substantial user interaction (e.g., [5, 14]). Exemplar-based reconstruction typically requires finding very precise correspondences to very similar exemplar 3D meshes. Automatic and semi-automatic approaches exist for chairs [2] and furniture [23, 39] that have highly informative contours and an enormous supply of available 3D models. Kholgade et al. [18] propose a general exemplar-based completion approach that requires carefully chosen user correspondences. In other recent work, Wu et al. [38] present preliminary results on automatic shape completion from depth by classifying hidden voxels with a deep network. We combine voxel prediction with shape priors from deformed exemplar meshes.

2. Method overview

From an input segmented depth image, we wish to produce a complete 3D mesh. As we show in Figure 1, this is a highly challenging problem that requires recognition or strong priors about likely shapes, and simple methods such as reflecting the depth points along the camera axis are rarely effective.

Our approach, illustrated in Figure 2 is to first find a similar depth image from a training set of meshes (Sec. 2.1). For retrieval, speed may trump clever similarity measures, because sublinear methods enable querying into a large dataset of meshes rendered from many viewpoints. We use random forests as a hashing function to produce several candidates. The candidate with minimum surface-to-surface distance based on the query and retrieved depth images is selected to produce an exemplar mesh and camera viewpoint.

The retrieved model may be of a different instance or category, and is likely to have a slightly different viewpoint. We, therefore, deform the retrieved mesh to better approx-

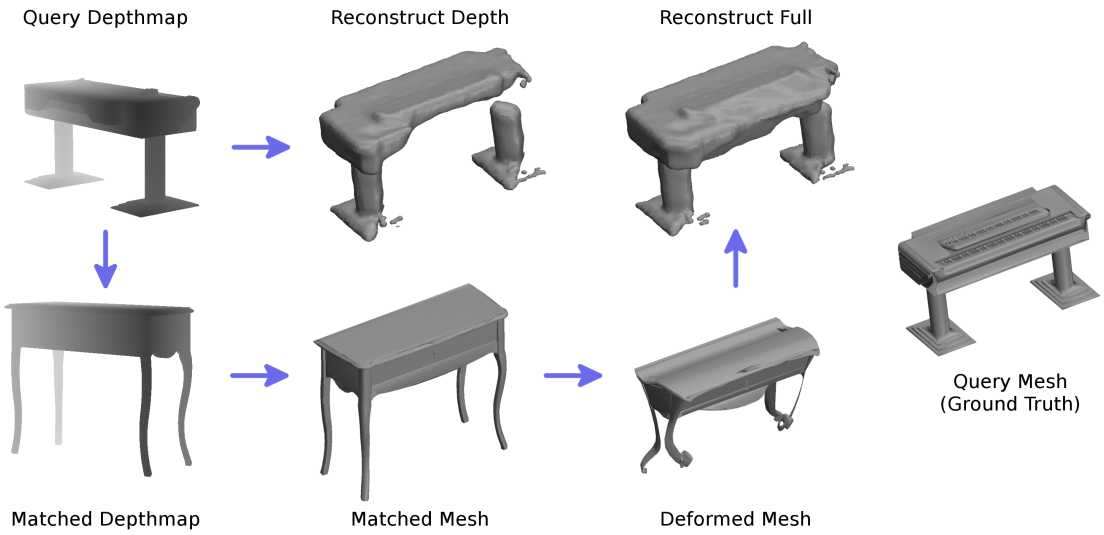


Figure 2: Pipeline for mesh reconstruction from depth image. A depth image of a piano is matched to a depth image of a table. The exemplar table is retrieved and deformed to better fit the observed depth points. Finally, a reconstructed mesh is created based on the observed depth points and deformed table exemplar. This mesh outperforms a mesh constructed from only depth points alone ("Reconstruct Depth").

imate the depth image (Sec. 2.2.3). Finding deformations to align different object models typically requires an experienced annotator to find corresponding points. We apply automatic methods to find likely correspondences between models and find symmetries within the exemplar mesh that can be used as constraints to avoid unlikely warping. We apply a similarity transformation, followed by a symmetry-preserving thin-plate-spline deformation.

Our final step is to use the deformed exemplar mesh to complete the input depth image (Sec. 2.3). If the exemplar mesh is similar to the query object, the deformed exemplar may provide a good estimate for the 3D shape of the query object. However, in many cases a similar object to the query will not exist in the dataset. For example, we may match a barrel to a building due to the similarity of their silhouettes and relative depth patterns from a particular viewpoint. Even in such cases, we show that the exemplar mesh can be useful in estimating the object's extent and for predicting symmetries of the query object which are used to reflect observed points. Then, a complete 3D mesh for the query object is constructed using graph cut to optimize a function that preserves observed and reflected depth points, conforms to the exemplar mesh, and discourages accidental alignment.

2.1. Retrieving similar exemplar 3D meshes

Given a depth image of an object, we want to find a similar 3D mesh and viewpoint in our training set. In this discussion, we consider 3D shape to be view-specific because we wish to recover the 3D surface within the camera's reference frame. We render each training mesh from many viewpoints, so the goal is to retrieve the training depth image, such that the corresponding rotated 3D mesh will align with the partially observed 3D query object. Retrieval complexity is important because the training set may contain many meshes rendered from many viewpoints.

Our approach is to use a random forest, trained to partition the training set into similar 3D shapes based on features of a depth image. Each tree of the forest acts as a hashing function, mapping the input features to a set of training examples. Random forests have been used for retrieval, e.g. by Fu and Qiu [9]. To retrieve based on shape similarity, we need to define entropy over a multi-dimensional variable (3D voxels), related to [7]. Our approach outperforms direct similarity-based matching and enables sublinear retrieval from a database of more than 22500 images.

Features. Depth maps are first cropped and resized to improve alignment of silhouettes. We use two simple features: whether a pixel at a given position is inside the silhouette and which of a pair of pixels is closer to the camera. The former encodes silhouette shape, and the latter encodes 3D shape in a manner that is invariant to depth scale.

3D Shape Similarity Entropy. Random forest training involves finding feature-based splits of the data that reduce entropy of the split groups. For each training image, we compute a 3D shape representation of 50^3 voxels for the corresponding mesh and rotation. Since it is not feasible to calculate entropy of 50^3 dimension feature, we take a

low dimensional projection of the voxels using non-negative matrix factorization (NNMF) [20] with dimension 50. For efficiency, we randomly subsample images before applying NNMF, which experiments indicate has little effect on accuracy while greatly improving speed. Then, we compute the coefficients of each new image using non-negative linear least-squares. We discretize the NNMF coefficients into 3 bins based on percentiles of non-zero values, and compute entropy for a set of images on the discretized coefficients, assuming that coefficients are independent.

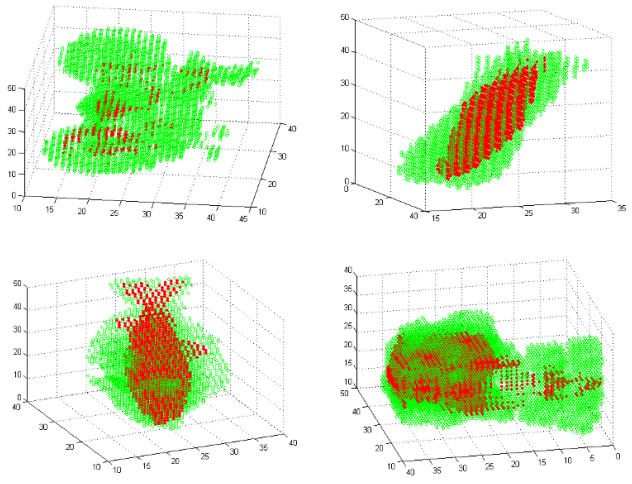


Figure 3: Voxel intersection (red) and union (green) of 3D shape examples in the same leaf node of our 3d-aware random forest. We can observe that similar objects with similar orientations are grouped into the same leaf node. For example, bikes and motorcycles in the top left, fish in the bottom left, or generic elongated shapes on the right.

Training. We train five trees, splitting until all leaf nodes contain five or fewer examples or cannot be improved with further splits. At each node, a random set of features is chosen (250 of each type). The feature that maximizes information gain and maintains a balanced split is selected. A balanced split is one which maintains at least $\max(.1S, 2)$ images in the smaller child, where S is the number of images in the node to be split. Leaf nodes tend to group similar objects, visualized in Figure 3.

Retrieval. Our random forest returns fifteen to twenty five potential matches. We select the match that has minimum surface-to-surface distance (Eq. 6) based on the query and retrieved depth images. Experimentally, we found that this selection procedure is nearly as effective as an oracle that selects the best potential match based on mesh-to-mesh distance.

2.2. Fitting retrieved exemplar to depth image

The exemplar retrieval process returns the 3D model in the training dataset with the most similar depth map to the query image. While the retrieved model is similar to the one in the query, it still needs to be deformed to fit the query point cloud in order to get a better approximation of shape. Here we describe our approach for symmetry constrained deformation for fitting a mesh to the query depth map.

2.2.1 Coarse Alignment

We back project both the query and the retrieved depth maps into the 3D world coordinates using the camera parameters of the retrieved depth map. These two point clouds are aligned through a similarity transformation using spin-images [16] based correspondences. Since we are matching 2.5D point clouds of different 3D models with similar but different orientations there are a significant number of outliers. These outliers are first pruned by Euclidean distance and then by a spectral correspondence matching technique [21] which considers geometrically conforming set of correspondence pairs as the inliers. The similarity transform is obtained using Horn's closed form solution to the least square problem [13].

2.2.2 Symmetry Detection

We preserve the symmetry of the retrieved mesh because the random forest matcher groups objects with similar global structure, for example bikes and motorcycles (Figure 3). While a motorcycle is not a good stand-in for a bike, symmetries are shared in a meaningful way. We use this symmetry as a constraint in deformation (Sec. 2.2.3) to determine the unseen parts of the object.

We find the significant symmetries in the retrieved mesh using [27]. We use a simplified version of this technique where we only consider planar symmetries and omit the symmetry basis reduction step. We also include a normal pruning step which is crucial in getting rid of the false symmetry matches. This is based on the fact that lines passing through the symmetry points should intersect at a point on the plane of symmetry (details in supplementary). This technique produces a number of symmetry proposals from which we choose the top few based on the size of the mean-shift clusters.

While this gives us the major symmetry planes in the mesh, we need to distribute these symmetries across the surface of the mesh. To do this, we sample points uniformly at random over the surface of the mesh. Then we reflect each point across every symmetry plane and match the reflection to the nearest sampled point. We retain points with a symmetric pair within a threshold of the match distance.

2.2.3 Deformation

We model the deformation using a 3D approximation thin-plate splines (TPS) model with additional symmetry constraints. Because we only see the object from a single view, correspondences for TPS deformation are only present on one side of the object. We must also deform unseen portions to preserve symmetry. We therefore choose control points for the TPS model by uniformly sampling points on the surface of the mesh. This is in contrast to the usual TPS model where the source points are generally chosen as the control points. Our choice of control points decouples the learning of weights for control points on either side of the symmetry plane leading to better deformation behavior when used with symmetry constraints. We also replace the usual radial basis function ($r^2 logr$) with r^3.

The usual approximation TPS deformation model [31] regresses the parameters of a function $\mathbf{f}$ which maps every source point u to an approximation of their target location $\mathbf{v}$ within a certain tolerance to the bending energy involved. This is different from interpolation TPS where we require $\mathbf{f}(\mathbf{u}) = \mathbf{v}$ for all specified correspondence pairs $(\mathbf{u}, \mathbf{v})$.

However, in our symmetry constrained approximation TPS model we enforce symmetric points to deform in such a way that the symmetry is preserved. Thus given N correspondences $(\mathbf{u_i}, \mathbf{v_i})$, M symmetry pairs $(\mathbf{p_j}, \mathbf{q_j})$ and their corresponding symmetry planes characterized by reflection mappings $\mathbf{R_j}$ such that $\mathbf{R_j}(\mathbf{p_j}) = \mathbf{q_j}$, we solve for a linear function $\mathbf{f}$ that minimizes the following objective

$$\sum_{i=1}^{N} \|\mathbf{v_i} - \mathbf{f}(\mathbf{u_i})\| + \lambda \mathcal{J}(\mathbf{f}) + \gamma \sum_{\mathbf{j=1}}^{\mathbf{M}} \|\mathbf{R_j}(\mathbf{f}(\mathbf{p_j})) - \mathbf{f}(\mathbf{q_j})\| \quad (1)$$

Here $\mathcal{J}$ is the bending energy involved in a given deformation and λ specifies our tolerance to this bending energy. $\lambda \to 0$ implies that don't want to enforce any smoothness constraint while larger λ would result in a smoother deformation achieved by relaxing the correspondence mapping. The parameter γ enforces the strength of symmetry constraints. Since there could be a significant difference between the number of symmetry pairs and the number of correspondences, we use the following normalization to avoid bias

$$\gamma = \alpha \times \frac{Number\ of\ correspondences}{Number\ of\ symmetry\ pairs} \quad (2)$$

Now α weighs the strength of the symmetry constraints relative to the correspondence constraints. We have used $\alpha = 2$ and $\lambda = 0.001$. If we are unable to extract any symmetries, we revert back to TPS without symmetry constraints and use $\lambda = .01$.

Finding parameters that minimize eq (1) is equivalent to finding the least square approximation to a set of linear equations. The linear equations for the usual approximation TPS model can be found in [31]. However, to minimize eq (1) we add the additional linear constraints

$$\mathbf{R_j}(\mathbf{f}(\mathbf{p_j})) - \mathbf{f}(\mathbf{q_j}) = \mathbf{0}\ \forall\ \mathbf{j} = \mathbf{1}, \cdots, \mathbf{M} \quad (3)$$

For a given symmetry plane defined by normal $\mathbf{n}$ and a point $\mathbf{m}$ on it, the reflection mapping $\mathbf{R}$ is a generalization of the Householder transformation (which defines a reflection about a plane passing through the origin) to reflection about an arbitrary plane. Specifically, $\mathbf{R}$ is defined by a matrix-vector pair $(\mathcal{A}, \mathbf{t})$ where

$$\mathcal{A} = \mathbf{I} - \mathbf{2nn^T} \quad (4)$$

$$\mathbf{t} = \mathbf{2nn^T m} \quad (5)$$

Any point p can then be reflected about the plane by using the function $\mathbf{R}(\mathbf{p}) = \mathcal{A}\mathbf{p} + \mathbf{t}$

2.3. Completing missing surfaces

Finally, we need to construct a 3D mesh for our query object based on the observed depth points and retrieved and deformed exemplar mesh. We pose this as a problem of predicting which voxels are occupied by the object. Voxels outside the depth silhouette or in front of observed depth points are observed to be empty, while voxels around observed points are known to be occupied by the object. We then predict which unobserved voxels are occupied based on six types of cues.

Features. The first two types of features are based on the observed depth map. First, voxels near observed depth points are likely to be occupied. $\mathbf{V}_{\text{dist}}(i)$ is the negative exponential of the distance of voxel i behind the depth map, in an orthographic projection. Voxels in front of the depth map have a distance of 1. Second, voxels that would require a large rotation to be exposed are more likely to be occupied (similar in concept to the accidental alignment principle). For example, a voxel behind a chair leg would have a higher accidental occlusion score than one behind the center of a wall. We compute $\mathbf{V}_{\text{ang}}(i)$ as the angle that the camera needs to move to make voxel i visible based on nearest point on the contour of the depthmap silhouette.

The third and fourth types of features are based on the retrieved and deformed mesh. We expect the query object to have similar symmetries to the matched object. We reflect points on the query depth map using symmetries of the closest points from the matched mesh. $\mathbf{V}_{\text{Sym}}(i)$ is the negative exponential distance of voxel i in front of the symmetry-generated points. The final cue is that the query mesh should tend to have the same voxels occupied as the retrieved mesh. $\mathbf{V}_{\text{mesh}}(i)$ indicates whether voxel i is occupied in the deformed exemplar mesh.

The last two features are based on the similarity of the query and match. One of them is depth distance between

query and match depthmaps. The other is the distance from the query point cloud to the deformed mesh surface.

Learning. We train a boosted decision tree (20 trees, 4 nodes per tree) based on LogitBoost [6] to predict which voxels are occupied. The boosted decision trees predicts a confidence of each voxel being occupied. Our purely depth-based reconstruction "reconstruct depth" uses only the first two feature types, our full reconstruction uses all six. We also learn a bias term on the validation set.

Voxel Prediction. The model produces per-voxel unary terms, indicating the log ratio probability that the voxel is occupied. For voxels observed to be occupied or unoccupied, the unaries have a value of 5 added or subtracted to strongly encourage labels to fit observations. We then perform min cut using a pairwise constant smoothing term of 10 in a 6-connected neighborhood in the 3D voxel space.

Reconstruction. We use two different reconstructions depending on how a mesh will be used. The first, applying marching cubes [25] directly to the predicted voxels produces meshes which are quantitatively accurate, but qualitatively unappealing due to small surface artifacts which render poorly. The second, Poisson reconstruction [17] produces smooth meshes which are visually appealing, but caused a performance decrease on validation. As such, we report quantitative results using the more direct marching cubes approach, and visualize results from Poisson reconstruction.

3. Evaluation

3.1. Datasets

We introduce a synthetic dataset built from the SHREC12 mesh classification dataset [22]. The SHREC dataset consists of twenty unaligned meshes from sixty diverse classes, including musical instruments, buildings, and vehicles. We align the meshes from each class consistently. The models are aligned such that the class of object is in the same viewpoint as the others. In Figure **??** we show aligned models. We align the models so that we can generate sample viewpoints that avoid unlikely angles (e.g., looking at the bottom of the car), but knowledge of alignment is not explicitly used in reconstruction.

We generate 50 views rejection-sampled on a unit sphere bounded within 20 degrees of the equator. We limit to this partial sphere to avoid strange or ambiguous views, for example, looking at a car from beneath.

We consider three reconstruction problems:

- *Novel View:* An object is seen from a new viewpoint, but the 3D model of the object is present in the dataset.
- *Novel Model:* The target object is from a known category, but the instance is new. For example, the input is a depth image of a car that is not present in the exemplar dataset, but models of other cars are present.
- *Novel Category:* The target object is from an unknown category. For example, the target object is an airplane, and there are no airplanes in the exemplar set.

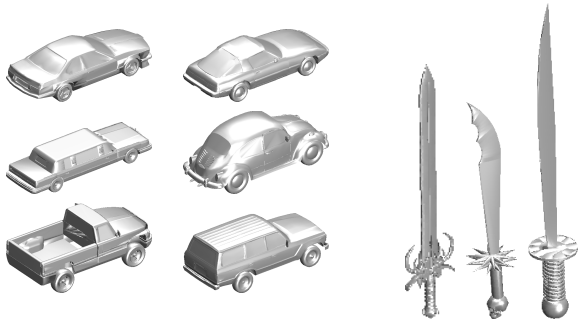


Figure 4: Manual alignment of our meshes allows us to eliminate unlikely and ambiguous views, for example, looking at a car from below or looking at the tip of a sword.

In total, our training set is 22500 images and we have 600 examples for testing each of the three experiments. It is not known to the algorithm whether the corresponding model or category of a viewed object is present in the exemplar set. Additional details of our dataset can be found in the supplementary materials.

3.2. Metrics

For evaluation we propose two metrics. The first is the intersection over union for two voxels (V_{iou}). We construct a voxel map of 200 voxels squared for both our query and match, and compute the intersection over union.

The second is a surface distance metric, V_{pcl} computed by densely sampling points on the meshes and using a normalized pointcloud distance. This gives us an estimate of the surface agreement of our meshes,

$$V_{pcl} = \text{mean} \min_m \|q - m\| + \min_q \|m - q\| \qquad (6)$$

Note that larger numbers are better for voxel I/U, and smaller numbers are better for surface distance.

3.3. Experiments

Shape retrieval. In Table 1, we compare similarity of the retrieved and query mesh before deformation. We compare retrieval using completely random forests (choosing features so that nodes are balanced) using silhouette features or silhouette and depth features, and our proposed entropy random forest retrieval. Our proposed method performs better for novel class and novel model and similarly for novel

Voxel I/U		Baseline	Reconstruct Depth	Matched Mesh	Aligned	Deformed	Reconstruct Full
Novel Class	Mean	0.164	0.425	0.224	0.243	0.265	**0.439**
	Median	0.138	0.429	0.177	0.207	0.236	**0.459**
Novel Model	Mean	0.124	0.424	0.302	0.349	0.368	**0.490**
	Median	0.107	0.408	0.249	0.289	0.322	**0.489**
Novel View	Mean	0.185	0.453	0.453	0.525	0.537	**0.565**
	Median	0.174	0.439	0.430	0.523	0.544	**0.582**
Surface Distance		Baseline	Reconstruct Depth	Matched Mesh	Aligned	Deformed	Reconstruct Full
Novel Class	Mean	0.292	**0.030**	0.057	0.065	0.057	**0.030**
	Median	0.286	**0.025**	0.053	0.058	0.048	**0.025**
Novel Model	Mean	0.264	0.028	0.039	0.042	0.037	**0.022**
	Median	0.267	0.022	0.033	0.035	0.029	**0.018**
Novel View	Mean	0.241	0.032	0.030	0.029	0.025	**0.023**
	Median	0.241	0.026	0.025	**0.019**	0.025	**0.019**

Table 2: Shape reconstruction on test. Our method achieves drastic improvements over the baseline depth reflection, and when we have a well matching mesh (Novel model and Novel View), retrieving a match allows us to make significant improvements over our depthmap based reconstruction method.

Voxel I/U		RF Sil	+Depth	Entropy
Novel Class	Mean	0.131	**0.135**	0.131
	Median	0.072	**0.096**	0.086
Novel Model	Mean	0.318	0.310	**0.333**
	Median	0.240	0.206	**0.251**
Novel View	Mean	0.301	0.291	**0.322**
	Median	0.254	0.206	**0.256**
Surface Dist		RF Sil	+Depth	Entropy
Novel Class	Mean	**0.074**	0.076	0.075
	Median	0.071	**0.070**	0.072
Novel Model	Mean	0.038	0.041	**0.037**
	Median	0.035	0.039	**0.032**
Novel View	Mean	0.040	0.041	**0.037**
	Median	0.038	0.039	**0.032**

Table 1: Shape retrieval on validation. Using additional information improves results for queries with similar models (Novel Model and Novel View) and does not hurt the matches when only dissimilar models are available (Novel Class). RF Sil uses only features based on whether a pixel is inside the object silhouette, +Depth, additionally uses which of a pair of pixels is closer to the camera, and Entropy uses an entropy measure based on mesh similarity to chose the best split with both features.

view. As expected, the similarity of the retrieved mesh is better when similar meshes are available, "novel view" and "novel model", when compared to "novel class".

Shape completion. We compare our reconstruction method with several baselines and intermediate results in Table 2. "Baseline" is computed by reflecting the observed depth across a plane parallel to the image plane at maximum depth. It performs very poorly. "Reconstructed Depth", based solely on the query depth map, uses the first two feature types to predict voxel occupancy. "Matched Mesh" shows reconstruction accuracy proposing the retrieved 3d exemplar mesh directly, "Aligned" is after similarity transform, and "Deform" after symmetric-preserving thin-plate-spline transform. These measures use only the retrieved exemplar to predict the query object shape. Finally, "Reconstructed Full" is our full proposed method that combines features from the retrieved exemplar and observed depth points to predict complete object shape.

Our deformation improves results over a simple alignment in all experiments. Matching and deformation perform best in the "novel view" case where often the retrieved mesh is the same as the query mesh. Our full method outperforms "Reconstruct Depth" substantially for the "novel model" and "novel view" cases, and performs similarly in the "novel class" case. Thus, the retrieved meshes are more helpful when examples are available of the same or similar model. See Figure 5 for examples of reconstructions.

4. Conclusion

We proposed a method to reconstruct a complete 3D model from a single depth observation: retrieve a similar 3D mesh exemplar based on depth image matching; deform the exemplar mesh to better fit the observed depth points; and predict the complete shape of the target object based on both the deformed exemplar and observed depths. We propose a new dataset to evaluate shape reconstruction based on the SHREC'12 dataset. Our experiments indicate that the quality of the reconstruction depends strongly on the similarity of available exemplar meshes, but that good re-

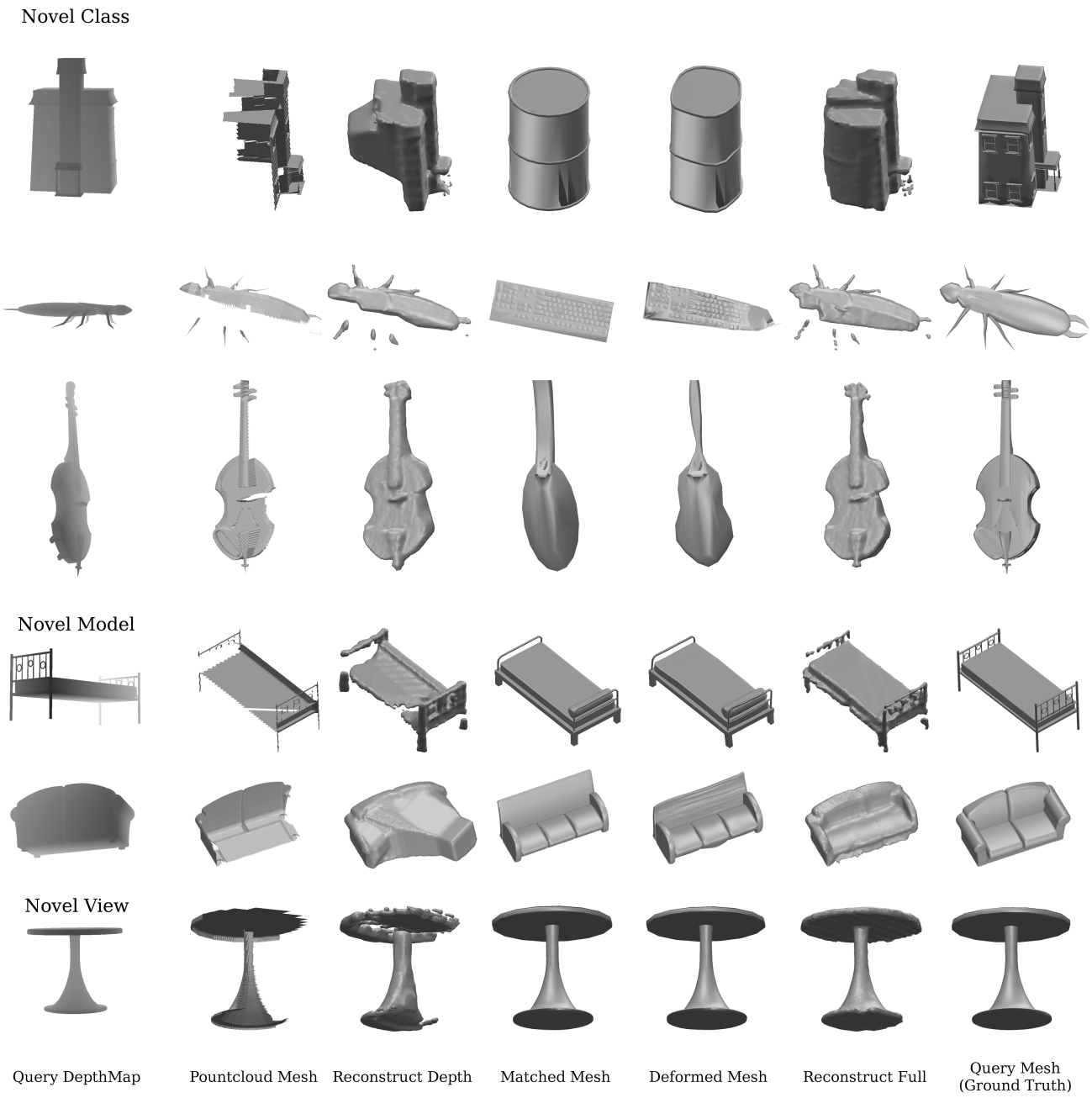


Figure 5: Examples of completed shapes. We view the objects from a view that is different from the input view so that unobserved parts of the object are visible. We wish to draw attention to the variety of deformations our model encodes. The barrel deforms to be more rectangular, the keyboard becomes tapered, and the spoon handle thins while the bowl takes on the cello's curves. Also note the information that the mesh provides in the reconstruction, the top of the bed, front of the couch, and top of the table are correctly hallucinated by our full approach.

constructions are sometimes possible, even when a similar exemplar is not available. We hope that by providing a benchmark dataset, we will encourage future research in object shape reconstruction. Future directions for this work include evaluating on sensor depth images from Kinect or other similar hardware, utilizing a similar pipeline for multiview reconstruction, and incorporating partial matches.

5. Acknowledgments

This material is based upon work supported in part by the National Science Foundation under Grants No. IIS 14-21521, IIS 09-16014, and IIS 14-21521; and in part by ONR MURI Award N00014-10-10934.

References

[1] A. A. Amini, Y. Chen, R. W. Curwen, V. Mani, and J. Sun. Coupled b-snake grids and constrained thin-plate splines for analysis of 2-d tissue deformations from tagged mri. *Medical Imaging, IEEE Transactions on*, 17(3):344–356, 1998. 2

[2] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *CVPR*, 2014. 1, 2

[3] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001. 2

[4] T. J. Cashman and A. W. Fitzgibbon. What shape are dolphins? building 3d morphable models from 2d images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(1):232–244, 2013. 1

[5] T. Chen, Z. Zhu, A. Shamir, S.-M. Hu, and D. Cohen-Or. 3-sweep: extracting editable objects from a single photo. *ACM Transactions on Graphics (TOG)*, 32(6):195, 2013. 2

[6] M. Collins, R. E. Schapire, and Y. Singer. Logistic Regression, AdaBoost and Bregman Distances. *Machine Learning*, 2002. 6

[7] P. Dollár and C. L. Zitnick. Structured forests for fast edge detection. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1841–1848. IEEE, 2013. 3

[8] M. Finch, J. Snyder, and H. Hoppe. Freeform vector graphics with controlled thin-plate splines. In *ACM Transactions on Graphics (TOG)*, volume 30, page 166. ACM, 2011. 2

[9] H. Fu and G. Qiu. Fast semantic image retrieval based on random forest. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 909–912. ACM, 2012. 3

[10] R. Gal, O. Sorkine, N. J. Mitra, and D. Cohen-Or. iwires: an analyze-and-edit approach to shape manipulation. *ACM Transactions on Graphics (TOG)*, 28(3):33, 2009. 1

[11] C. Goldfeder, M. Ciocarlie, J. Peretzman, H. Dang, and P. K. Allen. Data-driven grasping with partial sensor data. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 1278–1283. IEEE, 2009. 1

[12] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. *ICCV*, 2011. 1

[13] B. K. Horn, H. M. Hilden, and S. Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, 5(7):1127–1135, 1988. 4

[14] T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: A sketching interface for 3d freeform design. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, pages 409–416, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. 2

[15] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(5):433–449, 1999. 2

[16] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(5):433–449, 1999. 4

[17] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, 2006. 6

[18] N. Kholgade, T. Simon, A. Efros, and Y. Sheikh. 3d object manipulation in a single photograph using stock 3d models. *ACM Transactions on Computer Graphics*, 33(4), 2014. 2

[19] C. Kurz, X. Wu, M. Wand, T. Thormhlen, P. Kohli, and H.-P. Seidel. Symmetry-aware template deformation and fitting. *Computer Graphics Forum*, 33(6):205–219, 2014. 2

[20] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999. 4

[21] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1482–1489. IEEE, 2005. 4

[22] B. Li, A. Godil, M. Aono, X. Bai, T. Furuya, L. Li, R. J. López-Sastre, H. Johan, R. Ohbuchi, C. Redondo-Cabrera, A. Tatsuma, T. Yanagimachi, and S. Zhang. Shrec'12 track: Generic 3d shape retrieval. In *3DOR*, pages 119–126, 2012. 6

[23] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing ikea objects: Fine pose estimation. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2992–2999. IEEE, 2013. 1, 2

[24] J. Liu. Efficient recognition of rotationally symmetric surfaces and straight homogeneous yeneralized cylinders jane liu*, joe mundy*, david forsyth*, andrew zisserman and charlie rothwell(*) te center for research and development, 1 river rd, schenectady, ny 12345.(*) department of computer science, university of iowa, iowa city, ia 52242. 1993. 2

[25] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM siggraph computer graphics*, volume 21, pages 163–169. ACM, 1987. 6

[26] N. J. Mitra, L. J. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Transactions on Graphics (TOG)*, 25(3):560–568, 2006. 2

[27] N. J. Mitra, L. J. Guibas, and M. Pauly. Partial and approximate symmetry detection for 3d geometry. *ACM Transactions on Graphics (TOG)*, 25(3):560–568, 2006. 4

[28] R. Ohbuchi, K. Osada, T. Furuya, and T. Banno. Salient local visual features for shape-based 3d model retrieval. In *Shape Modeling and Applications, 2008. SMI 2008. IEEE International Conference on*, pages 93–102. IEEE, 2008. 1

[29] M. Pauly, N. J. Mitra, J. Giesen, M. Gross, and L. J. Guibas. Example-based 3D scan completion. In *SGP '05: Proceedings of the third Eurographics symposium on Geometry processing*. Eurographics Association, July 2005. 1

[30] M. Prasad and A. Fitzgibbon. Single view reconstruction of curved surfaces. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1345–1354. IEEE, 2006. 2

[31] K. Rohr, H. S. Stiehl, R. Sprengel, T. M. Buzug, J. Weese, and M. Kuhn. Landmark-based elastic registration using approximating thin-plate splines. *Medical Imaging, IEEE Transactions on*, 20(6):526–534, 2001. 5

[32] S. Song and J. Xiao. Sliding Shapes for 3D object detection in RGB-D images. *ECCV*, 2014. 1

[33] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, volume 4, 2007. 1, 2

[34] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 175–184. ACM, 2004. 1

[35] A. Tejani, D. Tang, R. Kouskouridas, and T. K. Kim. Latent-Class Hough Forests for 3D Object Detection and Pose Estimation. *ECCV*, 2014. 1

[36] Y. Wang, J. Feng, Z. Wu, J. Wang, and S.-F. Chang. From low-cost depth sensors to cad: Cross-domain 3d shape retrieval via regression tree fields. In *Computer Vision–ECCV 2014*, pages 489–504. Springer, 2014. 2

[37] W. Wohlkinger and M. Vincze. Shape-based depth image to 3d model matching and classification with inter-view similarity. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4865–4870. IEEE, 2011. 1

[38] Z. Wu, S. Song, A. Khosla, X. Tang, and J. Xiao. 3d shapenets for 2.5 d object recognition and next-best-view prediction. *arXiv preprint arXiv:1406.5670*, 2014. 2

[39] K. Xu, H. Zheng, H. Zhang, D. Cohen-Or, L. Liu, and Y. Xiong. Photo-inspired model-driven 3d object modeling. In *ACM Transactions on Graphics (TOG)*, volume 30, page 80. ACM, 2011. 2

[40] Y. Zheng, X. Chen, M.-M. Cheng, K. Zhou, S.-M. Hu, and N. J. Mitra. Interactive images: cuboid proxies for smart image manipulation. *ACM Trans. Graph.*, 31(4):99, 2012. 1, 2