Shape Completion from a Single RGBD Image

Dongping Li, Tianjia Shao, Hongzhi Wu, and Kun Zhou, *Fellow, IEEE*

Abstract—We present a novel approach for constructing a complete 3D model for an object from a single RGBD image. Given an image of an object segmented from the background, a collection of 3D models of the same category are non-rigidly aligned with the input depth, to compute a rough initial result. A volumetric-patch-based optimization algorithm is then performed to refine the initial result to generate a 3D model that not only is globally consistent with the overall shape expected from the input image but also possesses geometric details similar to those in the input image. The optimization with a set of high-level constraints, such as visibility, surface confidence and symmetry, can achieve more robust and accurate completion over state-of-the art techniques. We demonstrate the efficiency and robustness of our approach with multiple categories of objects with various geometries and details, including busts, chairs, bikes, toys, vases and tables.

Index Terms—RGBD camera, shape completion, single RGBD image

1 Introduction

As commercial RGBD cameras are becoming widely available, the process of 3D geometry acquisition has been considerably simplified. The user can construct a complete geometric model for an object, by walking around and pointing the RGBD camera towards the object, using existing techniques such as KinectFusion [1]. However, this capturing process still takes considerable time and effort, and remains cumbersome compared to using cameras for taking photographs: the user just needs to press one button and then the photograph is acquired instantly.

The goal of our paper is to make geometry acquisition as simple as taking a photograph. We aim to construct a complete 3D model for an object from a single RGBD image. The constructed model should look almost the same as the actual object from the captured view. From a novel view, the model should be visually plausible: 1) it should be globally consistent with the overall shape expected from the input image; 2) it locally possesses geometric details similar to the input image. Please see the bust in Figure 1 for an example.

This problem is known as shape completion from incomplete scans. Various approaches have been proposed, which can be classified as context-based (e.g., [2], [3]) or template-based approaches (e.g., [4], [5]). Context-based methods complete shape with details, but are limited to filling small holes. On the other hand, template-based techniques are able to fill large holes, but cannot guarantee that the resulting models have similar details as observed in the input image. Furthermore, they require that the input image should be topologically close to the deformed template, which is a strong condition that may not be easily satisfied.

We propose a new approach that combines the advantages of both context-based and template-base methods to complete shape from a single RGBD image. First, a collection of 3D template models of the same class as the image object are non-rigidly deformed and aligned with the input data, so as to compute an initialization for subsequent optimization. Next, a context-based global optimization framework is proposed to complete a final shape with consistent contexts based on the initialization.

Completing the shape from the geometry information in a single RGBD image is very challenging, mainly due to the uncertainty in the invisible parts. While deformed template models matching the input data can give some good hints of the invisible parts, it only provides a rough initialization for the completion. Existing context-based synthesis techniques do not work well in these situations (see comparisons in Figure 10, Figure 12 and Figure 14). To address this challenge, we introduce a volumetric-patch-based global optimization algorithm. We formulate an energy function to minimize the coherence error among the *local* details on the visible and invisible regions. We further introduce a set of high-level constraints, including visibility, surface confidence and symmetry, into the energy function to achieve robust completion that conforms to the *global* geometry and topology of the input image. The optimization is efficiently solved with an iterative algorithm.

To test the performance of the proposed approach, we use Microsoft Kinect to capture single RGBD images for multiple categories of objects with various geometries and details, including busts, chairs, bikes, toys, vases and tables. Experimental results show that the proposed approach recovers the unseen shapes that are visually plausible, and is more accurate than a simple combination of the state-of-the-art context-based and template-based techniques.

Contributions. The key novelties in this paper are:

- We show that by combining context-based and template-based techniques, complete 3D models can be constructed from single RGBD images for many categories of objects. The reconstructed models look almost the same to the input image under the capturing view and are plausible under novel views;
- We introduce a volumetric-patch-based optimization algorithm to shape completion, which includes a set of high-level constraints to achieve more robust completion over

- *D. Li, T. Shao, H. Wu and K. Zhou are with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, China.*
  E-mail: {lidongping83,tianjiashao}@gmail.com, {hwu,kunzhou}@acm.org
- *Corresponding authors: Tianjia Shao, Kun Zhou*

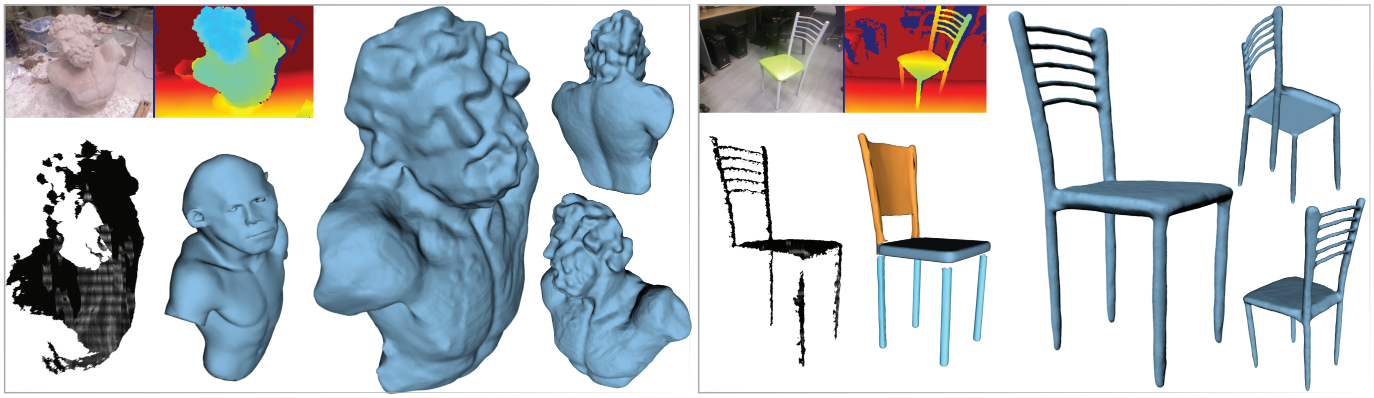


Fig. 1. Starting from a single view RGBD image (upper left), our volumetric-patch based optimization can faithfully complete the invisible geometry (right) from a rough initial solution (lower left), which is either a deformed template model (bust) or a composition of deformed parts (chair, with parts visualized in different colors) with different details and topologies .

state-of-the-art techniques.

2 Related Work

3D shape completion. An excellent, comprehensive survey on hole filling can be found in [6]. Related techniques may be categorized as smoothing and remeshing methods [7], scattered data fitting approaches [8], style transfer approaches [9], context-based and template-based methods. We review in details the latter two categories, which are closely related to our work.

Inspired by texture synthesis techniques [10], context-based methods exploit self-similarity priors to complete input object with rich textures or repetitive elements. Sharf et al. [2] first propose a context-based surface completion method to recover missing geometric features from existing regions. They use a greedy strategy to replace points using an octree structure. Inspired by patch-based image completion [11], [12], Harary et al. [3] introduce a context-based coherent completion algorithm with a global optimization on surface meshes. Such methods are mainly used to fill small holes and cannot be directly applied to a single RGBD image, where about half the geometry is missing. For urban scenes, repetitions are exploited to consolidate the imperfect data [13].

Template-based methods align template models with the input object and fill holes with the matched regions from the deformed template. Kraevoy and Sheffer [4] introduce a completion algorithm using a mapping between the incomplete mesh and a template model. Pauly et al. [5] retrieve suitable reference models from a database, warp the retrieved models to conform with the input object, and consistently blend the warped models to obtain the final 3D shape. These methods can fill large holes, but fail to recover details when the template does not have similar ones as input object.

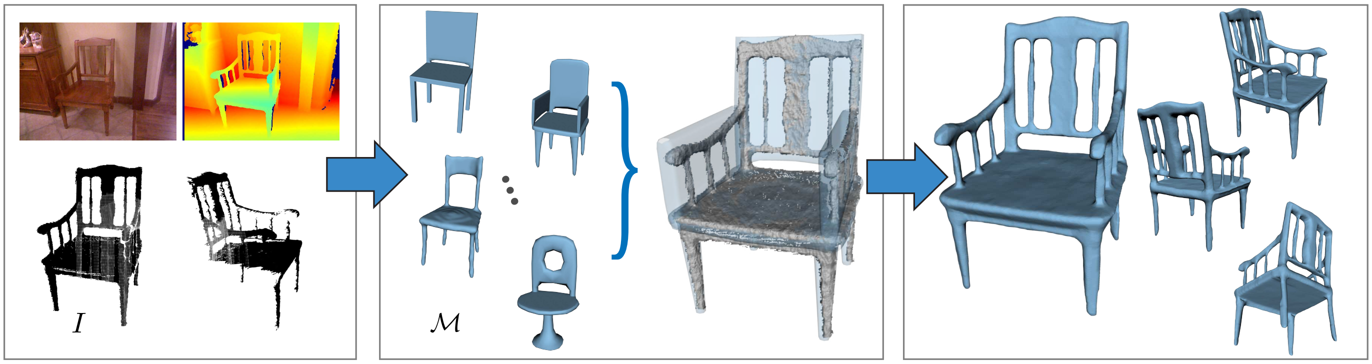
For special types of objects with a well-defined parametric model, such as human faces [14], [15] and bodies [16], [17], [18], template-based methods work well. In comparison, our goal is to handle more challenging and general objects.

3D modeling from RGBD images. Much effort has been devoted to obtaining high-quality geometry information from a single RGBD image [19], [20]. However, the goal of most related work is not to get a complete model. Shen et al. [21] introduce a structure-based approach to extract suitable model parts from a database, and compose them to form high-quality models from one RGBD image. This method, however, cannot guarantee that the modeling result contains the geometric details observed in the input image. Shao et al. [22] recover unseen structures from a single RGBD image using cuboids, but the geometry is not completed.

In terms of recovering a complete 3D model from a single depth map, the most similar work to ours is [23]. It introduces a learning-based algorithm to predict the unseen shape from a synthetic depth image. The method retrieves similar 3D models using view-based matching and transfers the symmetries and surfaces from retrieved models using boosted decision trees. It handles a broad range of objects without categorical knowledge. But the geometric details are usually lost. In comparison, categorical knowledge is required in our approach, but no training process is needed. In addition, we handle real RGBD images captured by a Kinect camera.

Another learning-based shape completion technique is described in [24]. The method uses a deep network to automatically estimate object categories and global structures/shapes, but it is difficult to guarantee the coherence between visible and invisible parts (i.e., the surface details may be lost). To get a complete model, 3D modeling techniques from multiple RGBD images (e.g., [1], [25], [26]) can be applied. In comparison, our method aims to save the time and labor of capturing multi-view data, by computing a complete shape from a single RGBD image.

PatchMatch algorithms. Our patch-based shape completion algorithm with global optimization on volumetric voxels is largely inspired by the PatchMatch approaches for image editing applications [12], [27], [28]. The core PatchMatch algorithm quickly finds correspondences between patches of an image by computing a nearest-neighbor field, which records the coordinate offsets of corresponding patches. The algorithm consists of three main components. First, the nearest-neighbor field is initialized with random offsets. Next, an iterative process is performed to propagate good patch offsets to adjacent pixels. Finally, random search is applied in the neighborhood of the best offsets calculated so



(a) input object I (b) initial solution from the model collection $\mathcal{M}$ (c) our patch-based optimization result

Fig. 2. The algorithm pipeline. We complete the invisible geometry from one RGBD image by combining context-based and template-based techniques. Given the input object I in a single RGBD image, a collection of 3D models of the same category are non-rigidly aligned with the depth data. The best aligned model (or composed parts) is selected as the initial solution (parts are used in this chair example). Then our global volumetric-patch-based algorithm is applied to optimize the shape under the constraints of surface confidence, visibility and symmetry (optional).

far to further improve the nearest-neighbor field.

3 Overview

Similar to state-of-the-art single-image-based modeling techniques [21], [29], our shape completion approach takes a single RGBD image of an object I segmented from the background, and a collection of 3D template models $\mathcal{M} = \{M_1, M_2, ..., M_n\}$ of a user-specified category, which the input image belongs to. Then the algorithm reconstructs a complete 3D shape from I with *global* geometry and topology consistent with the overall shape expected from I, and with the *local* details similar to those in I.

As illustrated in Figure 2, we first compute an initial shape guess for I, based on the model collection $\mathcal{M}$ (Section 4.1). The task is formulated as a non-rigid alignment problem from $\mathcal{M} = \{M_1, M_2, ..., M_n\}$ to I. Global non-rigid alignment [29] often produces a good initial solution, which is computed from the deformed template model M_i^* that aligns best with I. Moreover, if the models in $\mathcal{M}$ are already segmented to parts [30], we further refine the alignment in a global-to-local manner [21], and compose the best aligned parts together to calculate the initial guess.

As the model collection typically does not contain a shape or a part that is exactly the same as the object to be reconstructed, the initial completed shape from the non-rigid alignment is typically rough – with different geometry, topology and details (as shown in Figure 1 and Figure 8). Thus we further optimize the initial guess to obtain the final complete shape. We formulate this task as a constrained volumetric-patch-based optimization (Section 4.2). The local details can be consistently recovered within the patch-based optimization framework, while the constraints, including visibility, surface confidence and symmetry, help recover the globally similar geometry and topology. Finally, a patch-based denoising algorithm (Section 4.3) is applied to the optimized shape, to remove high-frequency noise.

Database. Our 3D model database includes 6 model collections of different categories (bust, chair, bike, toy, vase and table). Please see the supplemental material for all the database models.

For the chair and table, we download the pre-segmented 3D models from public datasets [30], which can be aligned in a global-to-local manner [21] to improve the accuracy of alignment. We also segment models in the toy category for the same purpose. For the bust, bike and vase, we find that global non-rigid alignment [29] is sufficient and do not perform part segmentation.

To support symmetry constraints, we perform symmetry analysis [31] to detect reflective and rotational symmetries in each model. We also refine the upright directions of the models to facilitate the initial alignment, using the method in [32]. We construct the embedded deformation model [29] to parameterize a non-rigid deformation. Please refer to [33] for details.

Volumetric representation. We express the surface using a volumetric representation, similar to previous work such as [1], [34]. To initialize this representation, the input depth image is back-projected into a global coordinate space, assuming that the camera is at the origin of the global coordinate. Then an axis-aligned 3D volume grid is constructed with a voxel size of $\max(l_x, l_y, l_z)/w$, where $w = 512$ is the default volume resolution and the bounding box of the point cloud corresponding to the input depth has a size of $[l_x, l_y, l_z]$. Finally, the point cloud is embedded into voxels using a variant of signed distance functions (SDFs) [35], which record the relative distances to the actual surfaces. We also only store a truncated region around the actual surface, similar to [1] (see Figure 3).

We define volumetric patches as groups of neighboring voxels with the size of $r \times r \times r$ ($r = 5$ in our implementation). The patches are densely sampled on every voxel. By using volumetric patches, we avoid the complex operations used in surface-patch-based methods [3], such as iterative closest point (ICP) for alignment and remeshing for patch updating, so that the completion process is more robust to geometric errors. Patch dissimilarity is natively measured using the concatenation of distance function values in voxels, hence avoiding the high computational cost of feature descriptors (e.g., the heat kernel feature (HKS) used in [3]).

4 Shape Completion

We solve a global optimization on voxels to recover the invisible geometry of the object I from an RGBD image,

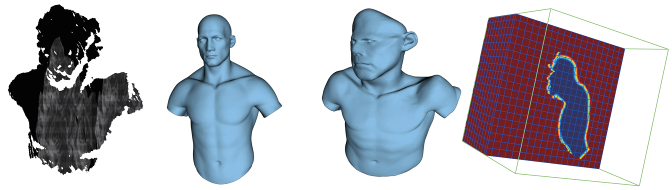


Fig. 3. Initial solution for our optimization. Starting from the input object I (left), one template model from the model collection is rigidly aligned with the depth (middle-left). Then non-rigid deformation is applied to improve the alignment (middle-right). The initial solution is the truncated signed distance functions of input depth and best aligned model stored in a volume.

guided by an initial shape guess computed from the shape collection $\mathcal{M} = \{M_1, M_2, ..., M_n\}$ of the same category in the database. We begin by showing how to align the models to the input object I to obtain the rough initialization (Section 4.1). Then we introduce the patch-based global optimization framework with various constraints, including surface confidence, visibility and symmetry, to get the final solution in Section 4.2. We also present a patch-based denoising algorithm (Section 4.3) for the optimized shape, to remove high-frequency noise.

4.1 Initial Solution

There are a large number of degrees of freedom in the invisible geometry. If we directly apply the patch-based optimization with an initial solution obtained using a traditional method (e.g., Poisson surface reconstruction [36]), the completion result may differ a lot from the overall expected shape (see Figure 4). Thus proper initialization for optimization is needed to generate visually plausible result. In our pipeline, we test over all models in the collection $\mathcal{M} = \{M_1, M_2, ..., M_n\}$ of the same category, and compute the initial solution either from the best aligned model or from the composition of the best aligned parts from different models.

Our alignment is computed in two stages: global rigid alignment and non-rigid alignment. An illustration is shown in Figure 3. We may further refine the global alignment into part-based alignment, if the model collection $\mathcal{M}$ is pre-segmented.

Rigid alignment. We adopt the same strategy as in [21] to estimate the rigid alignment. The upright orientations of the template model and the input depth are first aligned (the former is predefined in the database and the latter is determined by detecting the dominant supporting plane in the scene using a RANSAC approach [37]). Then the template is translated and scaled to fit into the bounding box of the scan. Finally, we exhaustively search for the orientation of the template around the upright axis, which minimizes the sum of distances between corresponding points on the template and the depth image.

Non-Rigid alignment. After the rigid alignment, the embedded deformation model [33] is applied to perform non-rigid alignment. The embedded deformation algorithm samples a set of control points $\mathbf{p}_\alpha$ over the template mesh (200 in our implementation), and defines the associated weighting functions $B_\alpha(.)$ for the control points. Given a vertex $\mathbf{v} \in \mathbb{R}^3$, its deformed position $g(\mathbf{v})$ is a weighted sum of its position after application of the affine transformations of the control points:

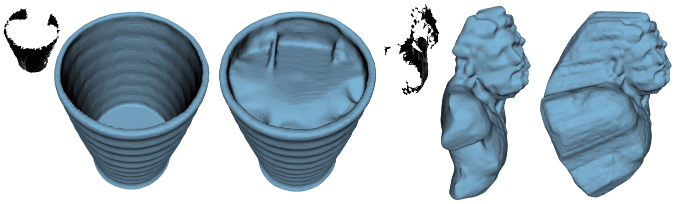


Fig. 4. Effect of initialization from the template model. For a vase/bust, the left/right shows the optimization results with/without the template. The input depth map is shown on the upper left.

$$g(\mathbf{v}) = \sum_{\mathbf{p}_\alpha} B_\alpha(v)[\mathbf{R}_\alpha(\mathbf{v} - \mathbf{p}_\alpha) + \mathbf{p}_\alpha + \mathbf{t}_\alpha], \tag{1}$$

where $\mathbf{R}_\alpha$ and $\mathbf{t}_\alpha$ represent the rotation matrix and translation vector of control point $\mathbf{p}_\alpha$ during deformation. The deformation energy function includes the matching error, the shape smoothness cost, and the rigidity cost.

To preserve symmetry during deformation, we add a symmetry constraint term into the deformation energy as in Rock et al. [23]. Specifically, for each estimated symmetric relation, we uniformly sample symmetric point pairs on the surface mesh. Suppose we have in total N point pairs $\{(t_i, s_i), \mathcal{R}_i\}$ ($\mathcal{R}_i$ is the corresponding symmetric transformation). We preserve symmetries during the deformation by adding the following energy into the embedded deformation model:

$$\frac{1}{N}\sum_{i=1}^{N} ||\mathcal{R}_i(g(t_i)) - g(s_i)||^2. \tag{2}$$

We perform the alignment process for every template model M_i in $\mathcal{M}$. The model with the highest score is selected for the initialization.

Optionally, the global deformation may be further refined in a global-to-local manner, if the model collection $\mathcal{M}$ is pre-segmented [21]. To do this, we perform part-level non-rigid matching, similar to [21], to further improve the alignment. For the pre-segmented models, each part is deformed separately, following the above rigid-to-nonrigid way within a local space. Finally we compose the parts from different models with optimized alignment to obtain the initial model.

Integrating aligned model to the volume. After the selected template model is aligned with the depth image, we convert the aligned model to the volumetric representation (see Figure 3). We first use the method of [38] to generate a signed distance field for the template, which uses piecewise quadratic functions to capture the local shape of the surface and a set of weighting functions to blend together these local shape functions. Then we truncate the distance field to the same range as the distance field of depth data. After that, we integrate the truncated distance field values to the volumetric representation of the depth, keeping the values of those voxels corresponding to the depth data unchanged.

4.2 Patch-based Optimization

The initial solution is usually rough because of the inevitable dissimilarity (geometrically and topologically) between the

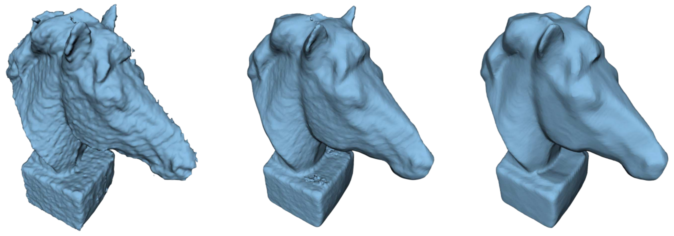


Fig. 5. Comparison between optimizing only invisible regions (left) and optimizing both visible and invisible regions (middle). After optimization, patch-based denoising filters the remaining high frequency noise (right).

input depth and the deformed model. So we would like to optimize the rough shape to obtain a final higher-quality shape with the details consistent with the visible ones in the input image. Essential constraints are added to make the completed shape similar in geometry and topology as the input data.

Our volumetric-patch-based optimization algorithm is inspired by the patch-based 2D image completion. But in order to obtain a reasonable optimization result from the rough initialization, we cannot simply apply the original formulation from [12], [28]. This is due to several key differences between 2D images and RGBD images. The first difference is that the visible regions in the RGBD image cannot be 100% trusted due to the high noise level. Thus we cannot directly replace the geometry of invisible regions using that from visible regions. The second difference is that the visibility constraint must be enforced during the completion process. The completed shape should not occlude other visible regions (this information is provided by the segmentation mask from the input image). The third difference is that template models may impose other global constraints (e.g., symmetry), which the optimized shape should satisfy.

To handle the above differences, we propose an optimization process which is performed on *both visible and invisible* regions, rather than only updating the unknown voxels with the known voxels unchanged. The basic patch-based optimization formulation is modified to handle the visibility and symmetry constraints. Besides, we introduce a surface confidence constraint, so as to prevent the visible regions from being optimized too far from the initial values. We would like to make sure the completed shape looks almost the same as the initial object from the captured view.

In the following, we first define related variables. Then we introduce the optimization function along with the constraints. Finally we show how to solve the optimization problem iteratively, using a patch matching step and a voting step in each iteration.

4.2.1 Variable Definitions

As aforementioned, we cannot fully trust the visible regions, as they already carry high level of noise from the RGBD image. If we simply follow the image completion algorithm to optimize the invisible regions from visible regions, the noise in the visible depth data could spread across the completed shape (see Figure 5, left).

Therefore, we define variables in a way different from image completion algorithms (or the state-of-the-art shape completion algorithms). Let S denote the *source* regions, which refer to both the known regions S_k of the input depth data and all the regions S_d of the template. The target T refers to the *target* regions to be optimized, consisting of both the known T_k and unknown T_u regions of the input depth data. During patch matching, the visible patches from T_k can only be matched with the patches from S_d belonging to the template. The invisible patches from T_u can only be matched with the visible ones from S_k to ensure consistency. The benefit of adding the template surface patch to the source is that this allows the visible noisy regions to be refined using the clean regions. Based on this definition, the final completed shape with our optimization method is much less noisy (see Figure 5, middle).

Algorithm 1 Volumetric shape completion

```
Initialize the completion with the model collection $\mathcal{M}$.
for each volume scale from coarse to fine do
    for iter $= 1$ to iter$_{\max}$ do
        // patch matching
        Randomly initialize a source patch $P$ and a transformation $f$ for each target patch $Q$.
        for k $= 1$ to k$_{\max}$ do
            for each target patch $Q$ in scan-line order do
                Perform propagation via Eq. (11)
                Perform random search via Eq. (12)
            end for
        end for
        // voting
        update $T$ via Eq. (13)
    end for
    Upsample to the next finer scale.
end for
```

4.2.2 Optimization Formulation

In patch-based optimization, the goal is to optimize the voxel values in T to minimize the matching error between patches in T and S, while satisfying the constraints of surface confidence, visibility and symmetry. The energy function is defined as:

$$\underset{T}{\operatorname{argmin}}\, \mathcal{E}_m(T,S) + \lambda_c \mathcal{E}_c(T,\tilde{T}) + \lambda_s \mathcal{E}_s(T) \text{ s.t. } \mathbf{M} \cdot \mathbf{d}(T) > 0. \tag{3}$$

Here $\mathcal{E}_m(T,S)$ measures the matching error, $\mathcal{E}_c(T,\tilde{T})$ is the confidence energy term, $\mathcal{E}_s(T)$ is the optional symmetry energy term, and $\mathbf{M} \cdot \mathbf{d}(T) > 0$ is the visibility constraint, which will be detailed later in this subsection. $\lambda_c = \lambda_s = 1$ are used in all experiments. We describe all terms in detail as follows.

Matching term. The matching energy term is similar to the basic formulation in image completion algorithm. It finds the optimal fill of the *target* regions in which every local neighborhood (patch) appears similar to a local neighborhood within the *source*, under a restricted set of transformations. The term is defined as:

$$\mathcal{E}_m(T,S) = \sum_{q \subset T} \min_{p \subset S} \mathcal{D}(Q,P), \tag{4}$$

where $Q = \mathcal{N}(q)$ is a $r \times r \times r$ *target* patch centered at a target voxel q, and $P = f(\mathcal{N}(p))$ is a $r \times r \times r$ *source* patch that is a result of a transformation (rotation/reflection) f applied on a patch $\mathcal{N}(p)$ centered at a source voxel p. The new voxel values after transformation are obtained by bilinear resampling. $\mathcal{D}(Q, P)$ gives the scalar-valued dissimilarity between Q and P, which is defined as:

$$\mathcal{D}(Q, P) = \|\mathbf{d}(Q) - \mathbf{d}(P)\|^2, \tag{5}$$

with

$$\mathbf{d}(\mathcal{N}(q)) = \{d(q^0), d(q^1), ..., d(q^n))\}, \tag{6}$$

Here $d(q^i)$ is the signed distance stored in a voxel q^i, and $n = r \times r \times r$.

Confidence constraint. In order to make sure that the completed shape looks almost the same as the initial object from the captured view, the target voxels in visible regions should be constrained by the values from the input depth image (or the initial solution). Thus we introduce a confidence term to measure how much a target voxel can be modified:

$$\mathcal{E}_c(T, \tilde{T}) = \sum_{q \subset T} \|w_q(d(q) - \tilde{d}(q))\|^2. \tag{7}$$

Here $\tilde{T}$ is the initial *target* regions, computed from the initial solution. $\tilde{d}(q)$ is the initial distance field value of the target voxel q, and w_q is a weight measuring the reliability for $\tilde{d}(q)$. The invisible voxel q_u^i satisfies $w(q_u^i) = 0$, as it is totally unknown. The reliability of the visible voxels is determined by: 1) input noise level and 2) inaccuracy of the distance field calculated from a single depth image (a single view may results in inaccurate distances due to missing data). Thus for each visible q_k^i, $w(q_k^i)$ is defined as:

$$w(q_k^i) = \exp\left(-\frac{\theta^2}{\sigma^2}\right) \exp\left(-\frac{l^2}{\sigma_l^2}\right). \tag{8}$$

The first term measures the distance field, where θ is the function value stored in a voxel, and σ^2 is the variance of the values in all visible voxels. If a voxel is closer to the visible surface (the value is near 0), the calculation of distance field is less affected by missing data, so the first term gives higher confidence. The second term measures the noise level. l is the voxel's noise level estimated with [39], and $\sigma_l = 0.05$ is the mean noise level of Kinect data (estimated from 100 depth images).

Visibility constraint. The initial solution usually has visibility conflicts. For example, in Figure 8(d), the back of the aligned chair fills the empty space between the railings of the scanned chair. To avoid such occlusion, we have to keep the visible voxels being always visible during the optimization. Since we have the segmentation mask of the input image, we can easily classify visible and invisible voxels. Note that the segmentation mask is generated from the RGB image, which does not suffer from noise/holes of the depth image. To identify the visibility for each voxel, we project the voxel onto the 2D image and mark its type following the simple rules: 1) if the projection falls outside the mask, the voxel is empty (visible), and we fill in the largest positive value (1 in our implementation); 2) if the projection falls inside the mask and the depth value of the corresponding pixel is greater than the projected depth of the voxel, the voxel is marked as empty; otherwise (i.e., the corresponding pixel has no depth or its depth value is less than the projected depth of the voxel), we mark this voxel as unknown (invisible).

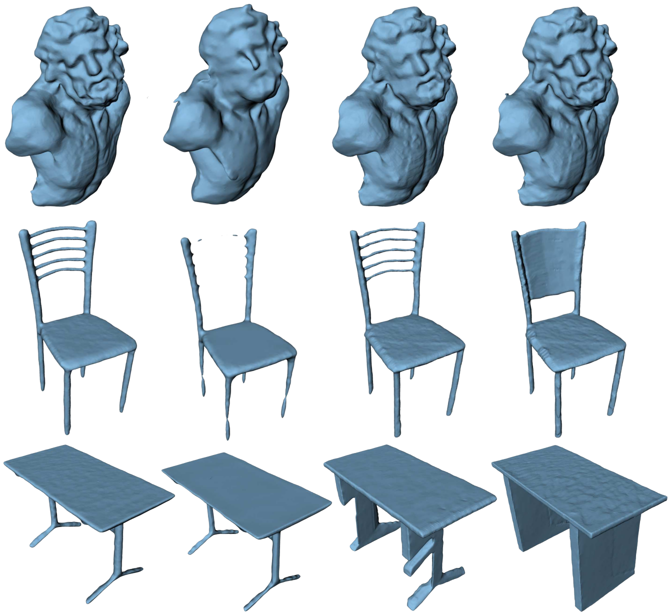


Fig. 6. Effects of constraints. From left to right are the optimization results: with all constraints, without surface confidence, without symmetry and without visibility constraint. The symmetry for the bust is manually added for comparisons only.

During the optimization, a hard constraint is added, to make sure that the visible voxels stay visible:

$$\mathbf{M} \cdot \mathbf{d}(T) > 0. \tag{9}$$

Here $\mathbf{M}$ is a diagonal matrix whose size is the number of target voxels N_T, with $\mathbf{M}_{ii} = 1$ indicating that the corresponding voxel q^i is visible from the camera, and $\mathbf{M}_{ii} = 0$ otherwise. $\mathbf{d}(T) = \{d(q^0), d(q^1), ..., d(q^{N_T}))\}$ is a column vector. Each element $d(q^i)$ is the distance field value for voxel q^i, and $d(q^i)$ should be larger than 0 during the optimization for visible voxels.

Symmetry constraint. We impose the symmetry constraint, whenever there are reflective or rotational symmetries in the aligned reference model. Note that symmetries of the models in the database are softly preserved during the deformation (Eq. (2)). We refit the plane or axis with the associated point-pairs recorded in the database, considering the impacts of small distortions during non-rigid alignment. Without loss of generality, we use the plane symmetry to describe the symmetry energy term. Given a target voxel q and a corresponding symmetry plane $\mathcal{P}_q$, the symmetric voxel is obtained by the reflection mapping $q^\star = \mathcal{R}(q, \mathcal{P}_q)$. Then the symmetry energy penalizes the differences between all q and $q^\star$:

$$\mathcal{E}_s(T) = \sum_{q \subset T} \|d(q) - d(q^\star)\|^2. \tag{10}$$

4.2.3 Iterative Solver

The objective function can be solved iteratively, following the algorithm proposed by Wexler et al. [40], which optimizes the function by alternating between two steps – patch matching step and voting step. Each step is guaranteed to decrease the energy function. In the patch matching step, the most similar patches in source regions are retrieved for all overlapping target patches. After patch matching, every target voxel $q \subset T$ has a set of best matched voxels $\Omega(q)$ from matched patches overlapping at q. Thus Eq. (3) reduces to a linear least square problem with an inequality constraint in the voting step. After solving Eq. (3), the target voxels are modified acoordinngly. The iterations continue until the voxel values converge, and are repeated across scales in a coarse-to-fine fashion. The pseudo-code of the iterative solver is listed in Algorithm 1.

Patch matching. To find the most similar patch P for Q in Eq. (4), a brute-force method is to compare Q with all source patches. However, the computation cost is prohibitive. Barnes et al. [27] proposed the Generalized PatchMatch algorithm, which efficiently finds dense approximate nearest source patches for all target image patches, with rotations and scales of source patches considered. We extend the algorithm from 2D to 3D, using rotations and reflections as the transformation f for source patches. We do not consider scales, as our target and source regions are prealigned and have similar scales.

Here we briefly describe the process of patch-match in our case. To initialize, for each voxel $q \in T$, whose corresponding patch $Q = \mathcal{N}(q)$, a random transformation function f and a random source patch $P = f(\mathcal{N}(p))$ are sampled. After the initialization, *propagation* and *random search* are performed for several iterations.

The purpose of propagation is to refine the current matching using neighboring patch information. As shown in Figure 7, given a target patch Q and its left neighbor Q_{x-}, the corresponding source patches are P and P_{x-}. During propagation, we match Q with P and the right neighbor of P_{x-}, denoted by P^+_{x-}. The better one is updated as the current matching for Q (similar for P^+_{y-}, P^+_{z-}):

$$P^* \leftarrow \underset{P_b \in \{P, P^+_{x-}, P^+_{y-}, P^+_{z-}\}}{\operatorname{argmin}} D(Q, P_b), \tag{11}$$

To further improve the matching results, random search is then applied, where a sequence of random candidate patches $\{P_i\}$ are selected to improve P:

$$P^* \leftarrow \underset{P_b \in \{P\} \cup \{P_i\}}{\operatorname{argmin}} D(Q, P_b). \tag{12}$$

Here $P_i = f_i(\mathcal{N}(p + w \cdot 0.5^i \cdot \mathbf{r}_i))$; w is the volume size, f_i is a random-sampled transformation, $\mathbf{r}_i = [-1, 1] \times [-1, 1] \times [-1, 1]$ and $i = 0, 1, 2, ...$ until the current search radius $w \cdot 0.5^i$ is below 1 voxel.

The propagation and random search are performed in scan-line order, voxel by voxel. In our implementation, we perform 4 iterations. In each odd iteration, we scan forward and in each even iteration backward. We only sample a discrete set of transformations f: the combinations of rotations $\{-\frac{3}{4}\pi, -\frac{2}{4}\pi \, ..., \frac{3}{4}\pi, \pi\}$ and reflections on all axes.

Voting. After patch matching, P in Eq. (4) is fixed, so every target voxel q has a set of best matched voxels from the matched patches overlapping at q. Now the minimization formulation in Eq. (3) reduces to linear least squares, with an inequality constraint. We first relax the problem to an unconstrained optimization problem. The unconstrained energy function $\mathcal{E}_m(T, S) + \lambda_c \mathcal{E}_c(T, \tilde{T}) + \lambda_s \mathcal{E}_s(T)$ is minimized when:

$$\nabla \mathcal{E}_m(T, S) + \lambda_c \nabla \mathcal{E}_c(T, \tilde{T}) + \lambda_s \nabla \mathcal{E}_s(T) = 0. \tag{13}$$

Given a target voxel q, we denote its best matched voxels by $\Omega(q)$. Then we have:

$$\nabla \mathcal{E}_m(T, S) = 2 \sum_k (\mathbf{d}(\mathbf{q}) - \mathbf{d}(\mathbf{q'}_k)), \tag{14}$$

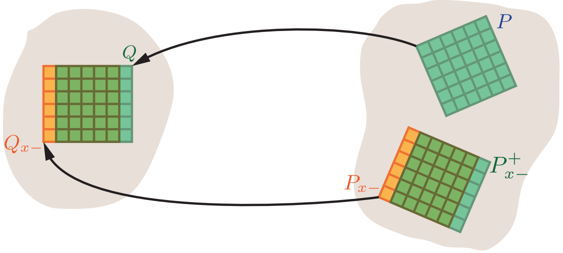


Fig. 7. An illustration of the propagation step in patch matching.

$$\nabla \mathcal{E}_c(T, \tilde{T}) = 2 \mathbf{w}_{\mathbf{q}} (\mathbf{d}(\mathbf{q}) - \tilde{\mathbf{d}}(\mathbf{q})), \tag{15}$$

$$\nabla \mathcal{E}_s(T) = 2(\mathbf{d}(\mathbf{q}) - \mathbf{d}(\mathbf{q}^\star)). \tag{16}$$

Here $\mathbf{q}$ is the vector concatenating all $q \in T$, and $\mathbf{q'}_k$ is the vector concatenating all the k_{th} element in $\Omega(q)$. Substituting Eq. (14), (15) and (16) back to Eq. (13), the only unknown in the equation is $d(\mathbf{q})$, which can be solved in closed-form. After obtaining the optimal solution for the unconstrained function, we project the solution back to the feasible domain by checking whether the visible voxels satisfy the constraint. If the value is smaller than or equal to 0, we simply assign a small positive value ϵ to the voxel ($\epsilon = 1e^{-5}$ in our implementation). Such strategy is applied to all the results in this paper, and empirically the process converges quickly. See Figure 11 for the convergence curves. An example is also shown in Figure 9.

4.3 Denoising: Patch-based Fusing

With the above optimization pipeline, a completed shape can be obtained, and the low frequency noise from the input depth is reduced. However, high frequency noise is still visible (see Figure 5, middle). This is because the optimization based on patch matching minimizes the coherence error among the target patches, which results in reduced low frequency noise (like pattern noise), while leaving high frequency noise unchanged.

To further improve the quality of the final shape, we propose a patch-based denoising algorithm on the input depth image. Our algorithm is inspired by the KinectFusion algorithm [1], which filters high-frequency noise by integrating multiframes of the depth data to a volume. Instead of using multiple depth images, we fuse similar patches from database models into the corresponding voxels occupied by the completed shape. Given a patch Q^o_i around a voxel belonging to the optimized shape, our algorithm finds top K (10 in our implementation) similar patches P^m_i from the database models. Note that these patches are noise-free, so high frequency noise is filtered by fusing these patches together. We follow the patch candidate selection process in the patch matching step. One denoising result is shown in Figure 5, right.

5 Experimental Results

Our algorithm is implemented in C++ and ran on a laptop with a 3.40Ghz quad-core processor and 16GB of memory. The experimental statistics including running time are listed in Table 1. Our optimization converges quickly in all experiments. We plot the objective function curves for selected objects (see Figure 11).

Results. We process various Kinect RGBD images in order to evaluate the proposed completion algorithm. We take single-view RGBD images on 11 objects of 6 categories: chair, table, bust/sculpture, bike, vase and toy. Corresponding database models are listed in the supplemental material. Figure 8 shows the input image I, the initial shape guess from the best aligned model/composed parts, and completion result, for each experimental object. We also prepare a synthetic 3D model to provide ground truth for algorithmic comparisons.

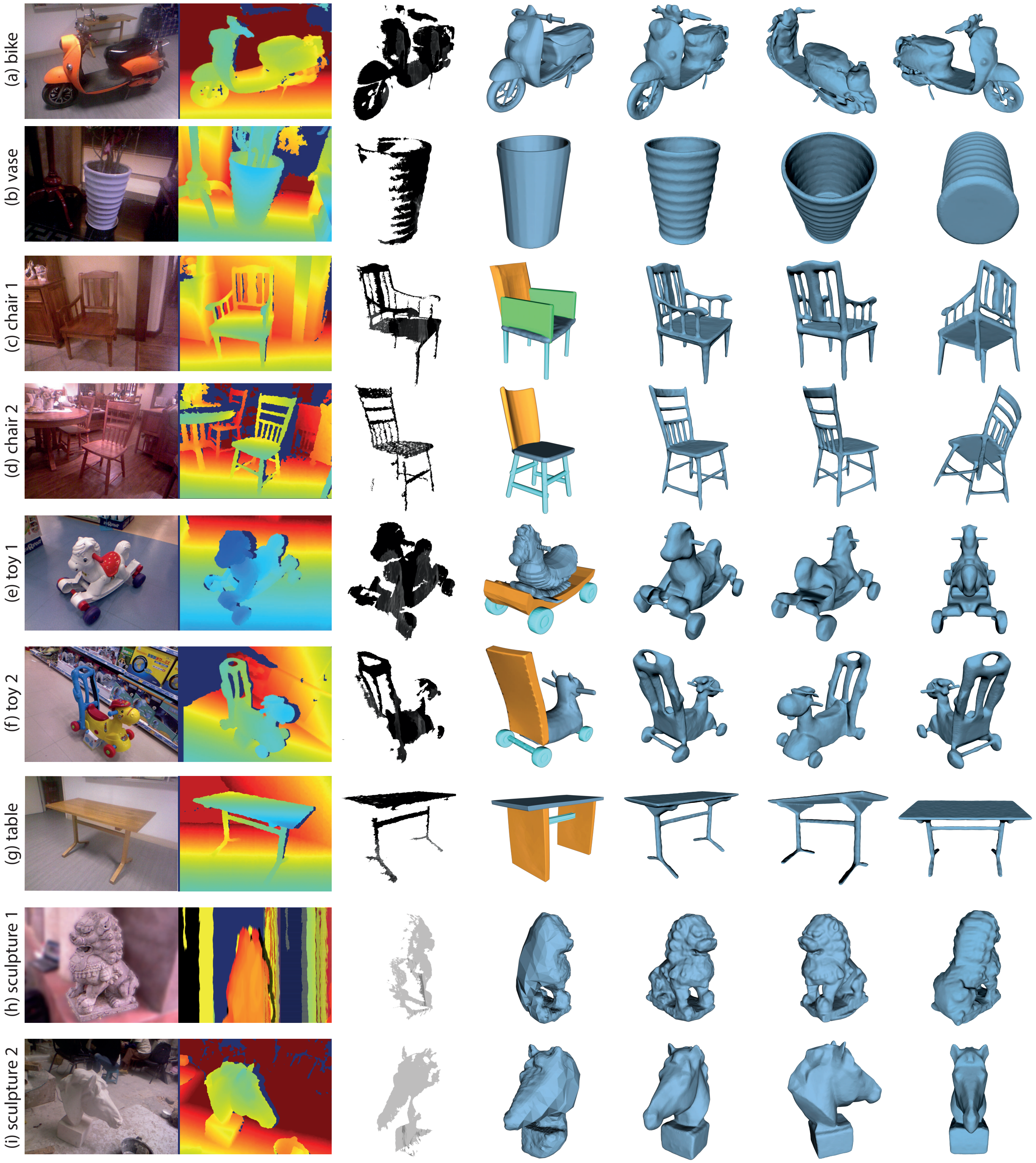


Fig. 8. Experimental results. From left to right: the input RGBD image, segmented object I, initial solution from best aligned model/composed parts (parts are shown with different colors), and completed shape (3 views). Note we use Poisson surface reconstruction to initialize the last two examples; each initialization is shown in a different view from the input.

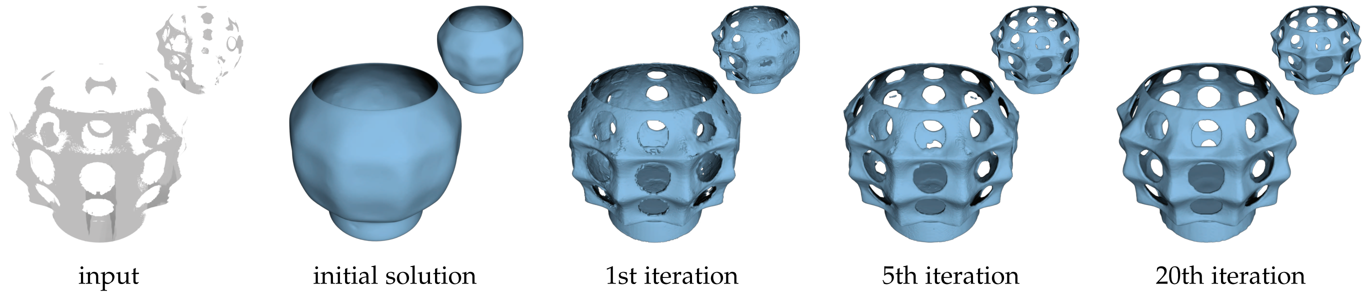


Fig. 9. Optimization progress on the vase example. Note that even though the topologies of the match model and the input object are quite different, our method produces a good result within a few iterations.

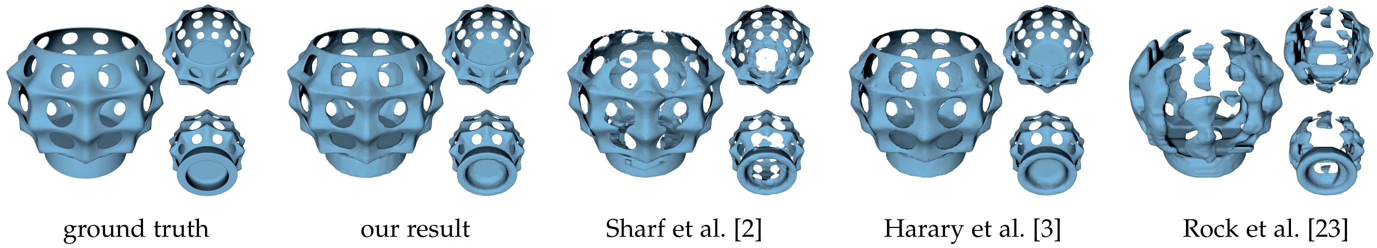


Fig. 10. Comparison results with a synthetic vase. All results are calculated from the same initial solution. For fair comparisons, we modify the methods of [2] and [3] to use initialization from template and add the constraints of visibility and symmetry. The method of [23] can be directly extended to incorporate the initialization, visibility and symmetry constraints.

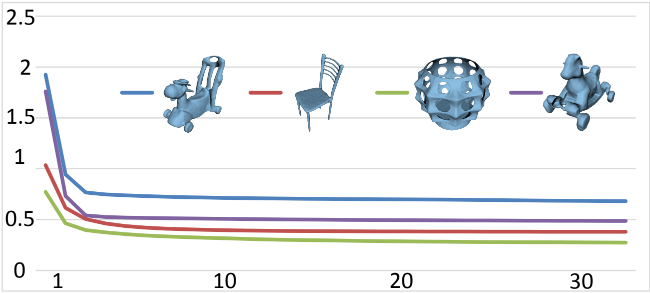


Fig. 11. Selected convergence curves during the optimization.

TABLE 1
Statistics for the experiment results.

Model	Template	Align.	Symm.	Opti. time
bike	single	-	-	269s
vase	single	-	-	275s
synthetic	single	-	-	620s
chair1	composed	-	-	321s
chair2	composed	-	-	205s
toy1	composed	-	-	331s
toy2	composed	-	-	307s
table	composed	-	-	478s
chair (teaser)	composed	-	-	202s
bust (teaser)	single	add point	-	418s
sculpture1	-	-	adjust	366s
sculpture2	-	-	adjust	196s

Our completion results are shown in Figure 1 and 8. Plausible shapes and consistent details are recovered from single-view RGBD images. Noise level in the original depth image is also reduced by our algorithm. The optimization successfully recovers a good shape with correct topology and consistent details from a rough initialization, even if the initialization differs a lot from the input object in geometry and topology.

For example, in Figure 8(c), the template chair has a considerably different topology from the physical model. The back and armrest of the template chair is solid, while there are large holes in the corresponding regions of the input depth map. Our optimization successfully recovers a plausible chair with a correct topology. Although one leg is missing in the depth map, we obtain a reasonable initial estimate from the template model, and recover the complete shape with an added leg consistent with visible ones. For the vase example in Figure 8(b), we start with a smooth vase as the initial estimation and obtain consistent details via rotational symmetry. For toy2 in Figure 8(f), our method completes the largely missing data using all constraints in the optimization. The symmetry constraint is important here to obtain the final result. For the table case, the triangular supporting structures below the surface are constructed from the visibility constraint, which actually do not exist in the physical object. For the bust example in Figure 1, in spite of the differences between the input depth and the initial template (a smooth head with no hair) as well as the imprecise alignment, our algorithm is able to complete the invisible half face and consistent hairs of the bust, as shown in the left part of Figure 1. No symmetry information is used in the optimization of this example. For other results, all energy terms and the visibility constraint are used. No template models are used in Figure 8(h) and 8(i), where objects have good symmetry and the visible regions almost occupy one symmetric side. Initialization from Poisson surface reconstruction [36] plus the symmetry constraint are sufficient to obtain good results.

Comparisons. We compare our algorithm with the state-of-the-art methods [2], [3], [23] on a synthetic example with ground truth (Figure 10) as well as real data (Figure 12). For fair comparisons, we make several important modifications for [2], [3], as they are originally designed to fill small holes. We first filter noise on the depth images for all the three methods, as their framework does not have surface confidence term. For [2], to satisfy the visibility constraint, we remove the points that are in visibility conflict after the point completion stage and then apply Poisson reconstruction. We restrict the search area for an octree cell to a local neighborhood of its symmetric cells to enforce the symmetry constraint (the initial solution is obtained by reflecting the visible part about the symmetry plane/axis). It is difficult to apply [3] to shape completion from a single RGBD image, because their algorithm requires a surface mesh for the computation. In order to calculate such a mesh, we apply the marching-cube algorithm [41] to our initial solution.

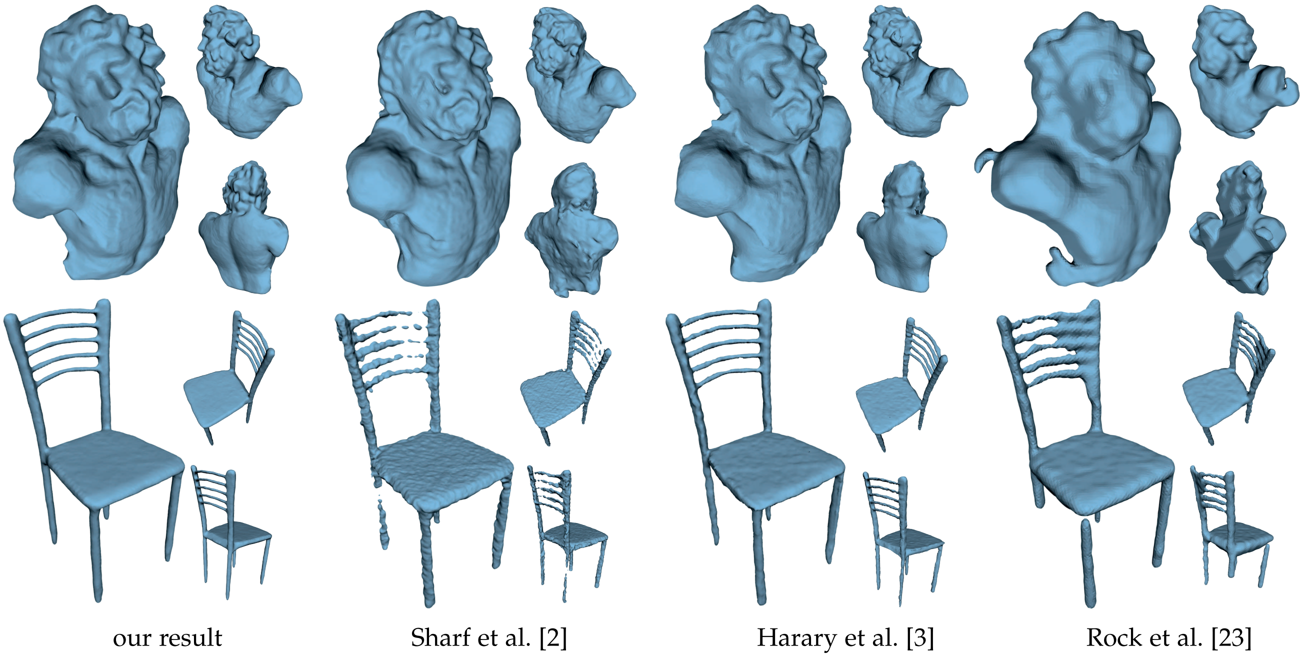


Fig. 12. Comparison results on real data. All results are calculated from the same initial solution. The chair inherits symmetry information from its template. There is no symmetry information for the bust. Note that as in Figure 10, we modify the methods of [2] and [3] to use initialization from template and add the constraints of visibility and symmetry. The method of [23] is directly adopted to incorporate the initialization, visibility and symmetry constraints.

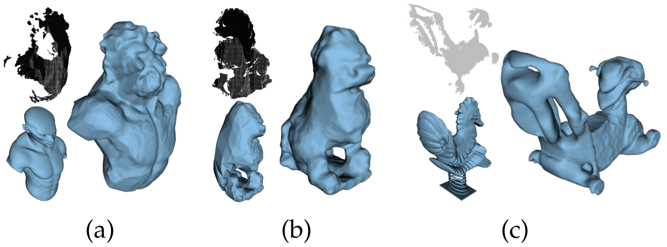


Fig. 13. Results from the default pipeline. (a) Bust: without user interaction in model alignment. (b) Sculpture 1: without user specified symmetry. (c) Toy 2: replacing composed parts with a single template for initialization.

The visibility is checked based on the segmentation mask to correct values of voxels that cause occlusions before calling the marching cube algorithm. Symmetry is also computed by reflecting the cells about the symmetry plane/axis. We directly adopt the algorithm of [23], since it considers visibility features and respects symmetry in the learning process. For fair voxel predictions, we add our database models to their training data and then learn the boosted decision trees.

The comparison results demonstrate that our algorithm out-performs the state-of-the-art techniques. In the synthetic example, our result approximates the ground truth better than other techniques. The non-conforming problem of [2] is more severe when large areas are missing (see the back of the bust in Figure 12). The ability of [3] to recover correct details from rough initialization is weaker than our algorithm (e.g., the recovered half face and back of head are smoother than ours, as the initialization shape is a smooth head without hairs). This is because their optimization is done on surface meshes and the extracted HKS features are calculated once from the initialization shape, which may result in wrong candidate selection. Also the noise level is higher than ours. The focus of [23] is to learn a general prediction model to recover a complete model from a depth image of any category, so details are usually missing in the results.

Default pipeline versus additional options. The default pipeline of our algorithm is first searching a best single template model to generate an initial solution and then performing patch-based optimization to obtain a final complete shape. If no template models of the same category are available, we simply use Poisson surface reconstruction to get an initial solution. We also adopt additional options in order to handle very challenging examples. These additional options include 1) user interaction in model alignment and symmetry adjustment and 2) generating the initial solution by composing the pre-segmented template parts. The different options for the experimental results are listed in Table 1.

We find that a small amount of user interaction in model alignment and symmetry adjustment can significantly improve the completion results for very challenging cases. As shown in Table 1, only the bust, sculpture1 and sculpture2 need user interactions. In the bust case, the automatic non-rigid alignment between the template model and the input object I is difficult to get a satisfying initialization, as shown in Figure 13(a). In such case, our system allows the user to click corresponding point pairs separately on the model and the RGBD image to improve the alignment. The completion result is considerably improved (Figure 1). The symmetry adjustment can be applied, when the non-rigid alignment causes severe distortion of the original symmetry plane/axis, or there is no symmetry information as shown in Figure 13(b). In this case, the user is allowed to add and adjust the symmetry plane/axis. Figure 8(h) shows the improved result after the symmetry adjustment. The interaction takes less than 4 minutes in our experiments.

As aforementioned in Section 4.1, if template models in $\mathcal{M}$ are pre-segmented, the initial solution may be improved. This option is useful for the input object that has very different global shape from models in $\mathcal{M}$ but contains similar local parts. An example is shown in Figure 13(c), where the topologies of the input toy and the best single template model are very different, resulting in an unsatisfactory completion result. After refining the initial solution from the composition of pre-segmented parts, the completion result is largely improved as shown in Figure 8(f).

TABLE 2
Quantitative Evaluation on the *Novel Model* and *Novel View* dataset from [23].

Voxel I/U		No Template	No Confidence	No Visibility	No Symmetry	Full	[2]	[3]	[23]
Novel Model	Mean	0.458	0.358	0.460	0.559	**0.605**	0.443	0.467	0.486
	Median	0.452	0.373	0.482	0.607	**0.658**	0.430	0.454	0.476
Novel View	Mean	0.445	0.421	0.541	0.616	**0.682**	0.492	0.535	0.578
	Median	0.453	0.436	0.580	0.633	**0.719**	0.486	0.543	0.595
Surface Distance		No Template	No Confidence	No Visibility	No Symmetry	Full	[2]	[3]	[23]
Novel Model	Mean	0.031	0.042	0.040	0.020	**0.017**	0.036	0.020	0.020
	Median	0.029	0.041	0.026	0.014	**0.011**	0.018	0.019	0.016
Novel View	Mean	0.031	0.035	0.025	0.015	**0.011**	0.030	0.016	0.018
	Median	0.030	0.034	0.019	0.013	**0.008**	0.017	0.015	0.015

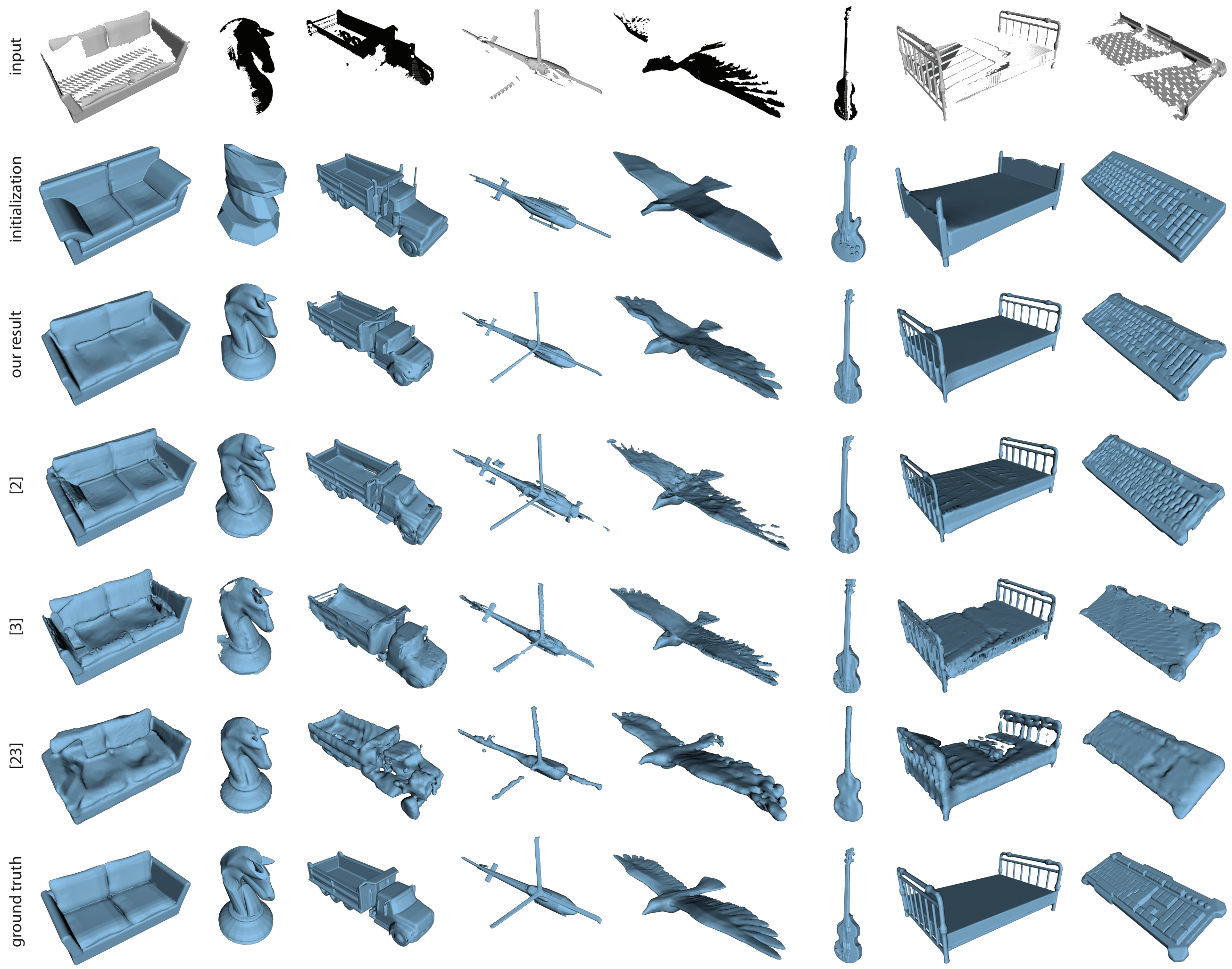


Fig. 14. Results of our algorithm on the dataset of [23]. From top to bottom: the input depth rendered as a point cloud from a novel view, the best-matched mesh as the initialization, our result, result of [2], [3], [23] and the ground truth.

Quantitative evaluation. To test the generalization of our method, we perform quantitative evaluations on the *Novel Model* and *Novel View* datasets provided in [23], which are composed of a broader range of objects, such as beds, keyboards, birds, guitars and trucks. Both *Novel Model* and *Novel View* contain 24 categories of objects. For *Novel Model*, each category contains 20 models, among which 5 models are selected to generate depth images as input from a set of viewing directions and the remaining 15 models are used as template models. For *Novel View*, each category contains 15 models, from which 5 models are selected to generate depth images. Unlike *Novel Model*, in *Novel View* all 15 models of every category are used as template models. Note that the original method of [23] retrieves the most similar object from the whole database. To make a fair comparison, we restrict the retrieval to be within the same category as the input depth image, just as our algorithm does.

The quantitative evaluation as well as the comparison with [2], [3], and [23] is listed in Table 2. We use the same metric as in [23] to measure the quality of the completion result with respect to the ground truth: voxel I/U and surface distance. The voxel I/U metric is computed as the ratio of intersection over union of the two volumes. The surface distance metric is computed by densely sampling points on two surfaces, and using a normalized point-cloud distance – the sum of distances between nearest points. A better completion result has a larger voxel I/U value and smaller surface distance value. We perform our algorithm as well as the algorithms of [2], [3], and [23] on all the generated depth images, and use the mean and median scores for evaluation. The effects of our algorithmic options are demonstrated quantitatively, including no template for initialization (using Poisson Surface Reconstruction instead), no surface confidence constraint, no visibility constraint and no symmetry constraint. As shown in the table, all the options are crucial to obtain a good completion result. Relatively, the surface confidence constraint is most important, as it keeps the visible regions staying similar to the input. The symmetry constraint is less important, as not all objects have symmetric property. The last four columns of Table 2 show that our algorithm consistently outperforms [2], [3], and [23] in terms of both the voxel I/U and surface distance metrics.

We also demonstrate some qualitative results as well as the comparisons with [2], [3], and [23] on the dataset in Figure 14. Qualitatively, our method is able to produce visually plausible results even if the initialization is different from the ground truth in geometry and topology (e.g., the truck initialization with irrelevant parts and the helicopter initialization with missing parts).

6 Discussions

Initialization from model collection. We find that the initialization from the model collection is crucial for shape completion. A collection of 3D models of the same category as the input object can provide a rough guess of the global shape, which is close to the expectation from the input image. To demonstrate the importance of initialization from template models, we compare the optimization results starting from the template model initialization and the traditional initialization of Poisson reconstruction. Figure 4 demonstrates that the template initialization can give better guidance for unseen regions.

Although our results depend on the initialization of template models from the model collection $\mathcal{M}$, we show that our method is robust to different initializations using the following control experiment: for each input image, we repeatedly remove the current best aligned model from the dataset and rerun our algorithm based on the remaining models (thus the initial shape becomes more different from the target). Our completion result remains reasonable with different initializations. Figure 15 shows some experimental results on real captured data (chair) and synthetic data from [23] (truck and bird). For each object, we give results from top-1, top-5, top-10 and least similar (chair: top-20; truck and bird: top-15) templates. The robustness comes from two facts: 1) the global constraints (visibility, symmetry and surface confidence) effectively prune incorrect initializations; 2) patch-based optimization helps preserve self-similarity.

Constraints. The confidence term, symmetry term and visibility term together constrain the completion result to have similar topology and geometry as the input data, and reduce the noise pattern in the input depth at the same time. We show the effect of each term on a set of examples (Figure 6), by comparing the results including/excluding the term. We observe that the confidence term effectively reduces the noise pattern; visibility constraint eliminates the topology error; and the symmetry energy strengthens high-level global consistency (e.g., making the filled-in part similar to a face in the bust example) and reduces the artifacts caused by visibility checking.

Volume resolution. As a volumetric method, our results depend on the resolution of the volume: higher resolution implicates finer surface details but requires more computational cost. Figure 16 shows a result of different resolutions. We use the default resolution 512^3 for all the results.

Part blending after alignment. Another benefit of the volumetric representation is that it is robust to the part blending problem after alignment. As shown in Figure 17, parts from different template models do not match well after alignment: both self-intersections and disconnections exist. Nevertheless the method [38] of generating distance functions can handle self-intersections and non-distant disconnections. Hence our algorithm naturally supports the blending of the aligned parts.

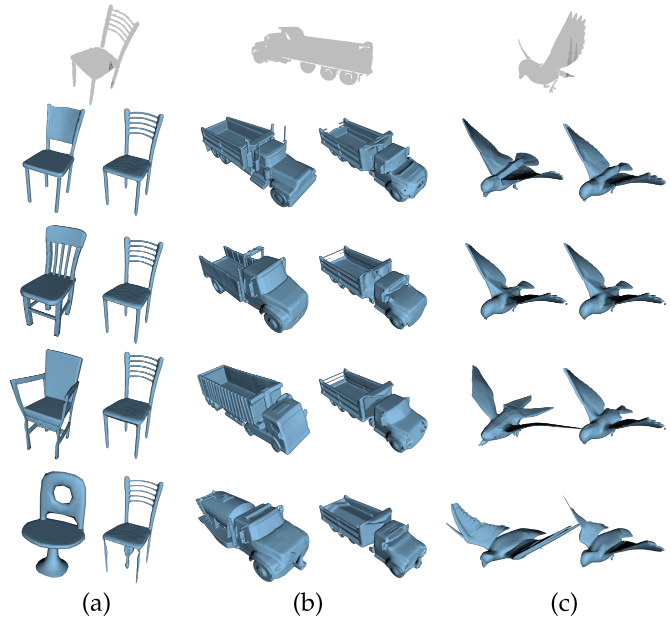


Fig. 15. Completion results with different initializations. For each input depth (top), the initialization (left) is computed from different templates/composed parts (from top to bottom: most similar to most different).Our completion results (right) remain plausible, until when the initialization is too different (bottom).

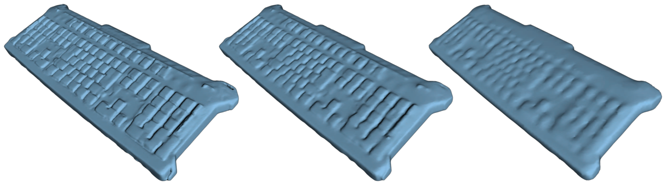


Fig. 16. Results from different volume resolutions. From left to right: 512^3, 256^3, 128^3.

Failure cases. Our algorithm may not work well for very complex objects. As shown in Figure 18, the tree and the piano contain delicate geometric features, which are difficult to capture from the initialization. The optimized results are thus unsatisfactory.

7 Limitations and Future Work

Our approach is subject to a number of limitations. First, as stated, our approach requires a segmented image and prior knowledge of the object category. These are classic problems studied in computer vision and many algorithms can be combined within our framework. Second, the user may add some corresponding points to ensure the accuracy of non-rigid alignment for very challenging cases. More robust shape alignment algorithm is worth exploring. Currently we only consider extrinsic symmetry, but intrinsic symmetry [42] may need to be considered to better handle large deformation. Another main limitation is that it is not guaranteed that our completed shape is the same as the fully scanned data by KinectFusion. For the chair example in Figure 19, our result is comparable with the KinectFusion result. But for the bust example, even a person cannot determine what the back should be like from the capture view, let alone the machine. The completed back using our

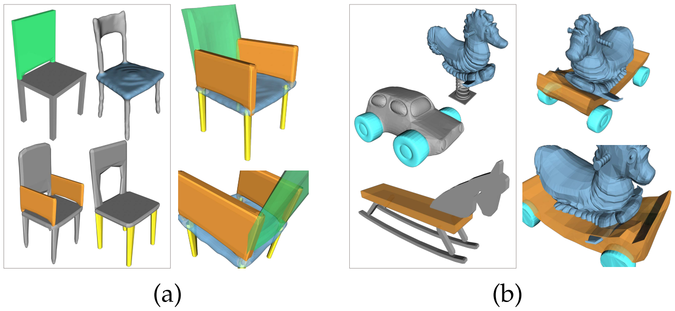


Fig. 17. Our volumetric representation can naturally handle the blending problem of the aligned parts for initialization. The chair (Figure 2) and the toy (Figure 8(e)) are successfully completed from parts that do not blend well. The original models from which the parts are taken are shown on the left.

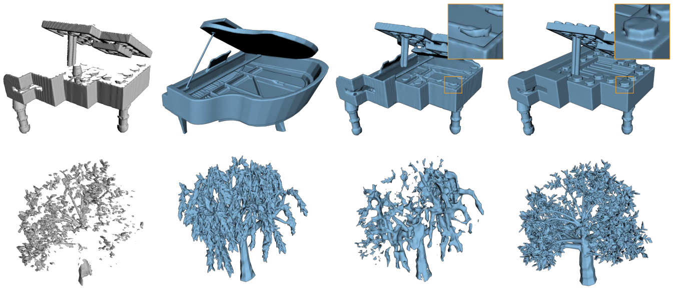


Fig. 18. Failure cases. From left to right: input depth, best matched template, our result and ground truth.

algorithm is largely affected by the deformed template, which is different from the fully scanned data.

A natural extension of this work is to complete the geometry of an entire scene. Finally, the result of our algorithm can be combined with smart image manipulation techniques [43] to achieve interesting effects.

Acknowledgments

This work is partially supported by the NSF of China (No. 61272305, No. 61303135 and No. 61402402), the National High-tech R&D Program of China (No. 2012AA010903), the National Program for Special Support of Eminent Professionals of China, and Lenovo's Program for Young Scientists.

References

[1] 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 *Proceedings of UIST*. New York, NY, USA: ACM, 2011, pp. 559–568.

[2] A. Sharf, M. Alexa, and D. Cohen-Or, "Context-based surface completion," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 878–887, Aug. 2004.

[3] G. Harary, A. Tal, and E. Grinspun, "Context-based coherent surface completion," *ACM Trans. Graph.*, vol. 33, no. 1, pp. 5:1–5:12, Feb. 2014.

[4] V. Kraevoy and A. Sheffer, "Template-Based Mesh Completion," in *Proceedings of SGP*, 2005.

[5] M. Pauly, N. J. Mitra, J. Giesen, M. Gross, and L. J. Guibas, "Example-Based 3D Scan Completion," in *Proceedings of SGP*, 2005.

[6] T. Ju, "Fixing geometric errors on polygonal models: A survey," *Journal of Computer Science and Technology*, vol. 24, no. 1, pp. 19–29, 2009.

[7] S. Bischoff, D. Pavic, and L. Kobbelt, "Automatic restoration of polygon models," *ACM Trans. Graph.*, vol. 24, no. 4, pp. 1332–1352, Oct. 2005.

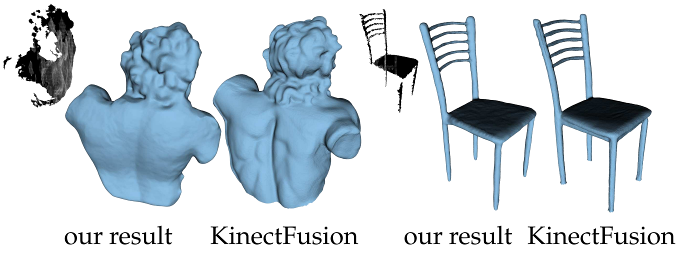


Fig. 19. Results from our algorithm / KinectFusion. Starting from single view depth data, our algorithm produces plausible shapes, but cannot guarantee that the completed shape is the same as the scanning result by KinectFusion.

[8] S. Shalom, A. Shamir, H. Zhang, and D. Cohen-Or, "Cone carving for surface reconstruction," *ACM Trans. Graph.*, vol. 29, no. 5, Dec 2010.

[9] C. Ma, H. Huang, A. Sheffer, E. Kalogerakis, and R. Wang, "Analogy-driven 3D style transfer," *Computer Graphics Forum*, vol. 33, no. 2, pp. 175–184, 2014.

[10] L.-Y. Wei, S. Lefebvre, V. Kwatra, and G. Turk, "State of the art in example-based texture synthesis," in *Eurographics, State of the Art Report, EG-STAR*. Eurographics Association, 2009.

[11] A. A. Efros and W. T. Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of SIGGRAPH*. New York, NY, USA: ACM, 2001, pp. 341–346.

[12] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "PatchMatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, Aug. 2009.

[13] Q. Zheng, A. Sharf, G. Wan, Y. Li, N. J. Mitra, D. Cohen-Or, and B. Chen, "Non-local scan consolidation for 3d urban scenes," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 94:1–94:9, 2010.

[14] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," in *Proceedings of SIGGRAPH*. New York, NY, USA: ACM, 1999, pp. 187–194.

[15] T. Weise, S. Bouaziz, H. Li, and M. Pauly, "Realtime performance-based facial animation," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 77:1–77:10, Aug. 2011.

[16] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "Scape: Shape completion and animation of people," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 408–416, Jul. 2005.

[17] A. Weiss, D. Hirshberg, and M. Black, "Home 3d body scans from noisy image and range data," in *Proceedings of ICCV*, Nov 2011, pp. 1951–1958.

[18] J. Tong, J. Zhou, L. Liu, Z. Pan, and H. Yan, "Scanning 3d full human bodies using kinects," *IEEE TVCG*, vol. 18, no. 4, pp. 643–650, Apr. 2012.

[19] H. Yu, "Edge-preserving photometric stereo via depth fusion," in *Proceedings ICCV*. Washington, DC, USA: IEEE, 2012, pp. 2472–2479.

[20] Y. Han, J.-Y. Lee, and I. S. Kweon, "High quality shape from a single rgb-d image under uncalibrated natural illumination," in *Proceedings of ICCV*. Washington, DC, USA: IEEE, 2013, pp. 1617–1624.

[21] C.-H. Shen, H. Fu, K. Chen, and S.-M. Hu, "Structure recovery by part assembly," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 180:1–180:11, Nov. 2012.

[22] T. Shao, A. Monszpart, Y. Zheng, B. Koo, W. Xu, K. Zhou, and N. J. Mitra, "Imagining the unseen: Stability-based cuboid arrangements for scene understanding," *ACM Trans. Graph.*, vol. 33, no. 6, pp. 209:1–209:11, Nov. 2014.

[23] J. Rock, T. Gupta, J. Thorsen, J. Gwak, D. Shin, and D. Hoiem, "Completing 3d object shape from one depth image," in *Proceedings CVPR*, 2015, pp. 1810–1817.

[24] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of CVPR*, 2015, pp. 1912–1920.

[25] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, and B. Guo, "An interactive approach to semantic modeling of indoor scenes with an rgbd camera," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 136:1–136:11, Nov. 2012.

[26] Q.-Y. Zhou and V. Koltun, "Color map optimization for 3d reconstruction with consumer depth cameras," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 155:1–155:10, Jul. 2014.

[27] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein, "The generalized patchmatch correspondence algorithm," in *Proceedings of ECCV*. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 29–43.

[28] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen, "Image melding: Combining inconsistent images using patch-based synthesis," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 82:1–82:10, Jul. 2012.

[29] H. Su, Q. Huang, N. J. Mitra, Y. Li, and L. Guibas, "Estimating image depth using shape collections," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 37:1–37:11, Jul. 2014.

[30] Y. Wang, S. Asafi, O. van Kaick, H. Zhang, D. Cohen-Or, and B. Chen, "Active co-analysis of a set of shapes," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 165:1–165:10, Nov. 2012.

[31] N. J. Mitra, L. J. Guibas, and M. Pauly, "Partial and approximate symmetry detection for 3d geometry," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 560–568, Jul. 2006.

[32] H. Fu, D. Cohen-Or, G. Dror, and A. Sheffer, "Upright orientation of man-made objects," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 42:1–42:7, Aug. 2008.

[33] R. W. Sumner, J. Schmid, and M. Pauly, "Embedded deformation for shape manipulation," *ACM Trans. Graph.*, vol. 26, no. 3, pp. 80–85, Aug. 2007.

[34] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proceedings of SIGGRAPH*. New York, NY, USA: ACM, 1996, pp. 303–312.

[35] S. Osher and R. Fedkiw, *Level Set Methods and Dynamic Implicit Surfaces*, ser. Applied Mathematical Sciences. Springer, 2003.

[36] M. Kazhdan and H. Hoppe, "Screened poisson surface reconstruction," *ACM Trans. Graph.*, vol. 32, no. 3, pp. 29:1–29:13, Jul. 2013.

[37] R. Schnabel, R. Wahl, and R. Klein, "Efficient ransac for point-cloud shape detection," *CGF*, vol. 26, no. 2, pp. 214–226, 2007.

[38] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel, "Multi-level partition of unity implicits," in *Proceedings of SIGGRAPH*. New York, NY, USA: ACM, 2003, pp. 463–470.

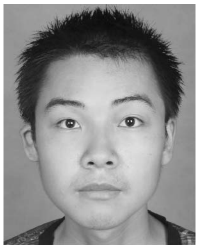
[39] S. Pyatykh, J. Hesser, and L. Zheng, "Image noise level estimation by principal component analysis," *IEEE Trans. Image Processing*, vol. 22, no. 2, pp. 687–699, Feb. 2013.

[40] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *IEEE Trans. PAMI*, vol. 29, no. 3, pp. 463–476, March 2007.

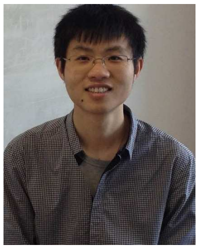
[41] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," in *Proceedings of SIGGRAPH*. New York, NY, USA: ACM, 1987, pp. 163–169.

[42] K. Xu, H. Zhang, W. Jiang, R. Dyer, Z. Cheng, L. Liu, and B. Chen, "Multi-scale partial intrinsic symmetry detection," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 181:1–181:10, 2012.

[43] 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.*, vol. 31, no. 4, pp. 99:1–99:11, Jul. 2012.



Dongping Li received the bachelors degree in mathematics from Zhejiang University, Hangzhou, China, in 2011. Currently, he is working toward the PhD degree at Graphics and Parallel Systems Lab, Zhejiang University. His research interests include RGBD-image reconstruction, image manipulation, and mesh deformation.



Tianjia Shao is currently an Assistant Researcher in the State Key Lab of CAD&CG, Zhejiang University. He received his PhD in Computer Science from Institute for Advanced Study, and his B.S. from the Department of Automation, both in Tsinghua University. His research interests include RGBD image processing, indoor scene modeling, structure analysis and 3D model retrieval.

Hongzhi Wu is an assistant professor in State Key Lab of CAD & CG, Zhejiang University. He received B.Sc. in computer science from Fudan University in 2006, and Ph.D. in computer science from Yale University in 2012. His research interests include appearance modeling, design and rendering. He has served on the program committees of PG, EGSR and SCCG.

Kun Zhou is a Cheung Kong Professor in the Computer Science Department of Zhejiang University, and the Director of the State Key Lab of CAD&CG. Prior to joining Zhejiang University in 2008, Dr. Zhou was a Leader Researcher of the Internet Graphics Group at Microsoft Research Asia. He received his B.S. degree and Ph.D. degree in computer science from Zhejiang University in 1997 and 2002, respectively. His research interests are in visual computing, parallel computing, human computer interaction, and virtual reality. He currently serves on the editorial/advisory boards of ACM Transactions on Graphics and IEEE Spectrum. He is a Fellow of IEEE.