Pacific Graphics 2018
H. Fu, A. Ghosh, and J. Kopf
(Guest Editors)

Volume 37 (2018), Number 7

Online Global Non-rigid Registration for 3D Object Reconstruction Using Consumer-level Depth Cameras

Jiamin Xu[1] Weiwei Xu[1†] Yin Yang[2] Zhigang Deng[3] Hujun Bao[1]

[1]State Key Lab of CAD&CG, Zhejiang University [2]University of New Mexico [3]University of Houston

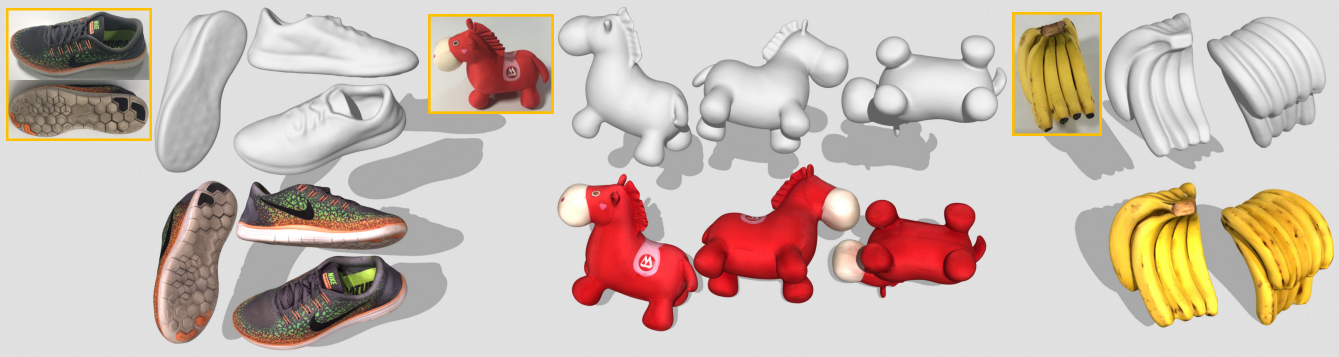

Figure 1: Selected 360-degree reconstruction results using our algorithm. The resulting reconstructions are rendered using the stitched texture as diffusion maps. Please refer to Table 1 for detailed statistics. The real-world photos of the target objects are shown in yellow boxes.

Abstract

We investigate how to obtain high-quality 360-degree 3D reconstructions of small objects using consumer-level depth cameras. For many homeware objects such as shoes and toys with dimensions around $0.06 - 0.4$ *meters, their whole projections, in the hand-held scanning process, occupy fewer than* 20% *pixels of the camera's image. We observe that existing 3D reconstruction algorithms like KinectFusion and other similar methods often fail in such cases even under the close-range depth setting. To achieve high-quality 3D object reconstruction results at this scale, our algorithm relies on an online global non-rigid registration, where embedded deformation graph is employed to handle the drifting of camera tracking and the possible nonlinear distortion in the captured depth data. We perform an automatic target object extraction from RGBD frames to remove the unrelated depth data so that the registration algorithm can focus on minimizing the geometric and photogrammetric distances of the RGBD data of target objects. Our algorithm is implemented using CUDA for a fast non-rigid registration. The experimental results show that the proposed method can reconstruct high-quality 3D shapes of various small objects with textures.*

CCS Concepts

•***Computing methodologies*** → *Shape modeling;*

1. Introduction

3D reconstruction is one of the core methodologies of digital content creation for a wide range of graphics applications, especially when digitized 3D objects or environments are of primary concerns [BRU10, RZS07]. Despite the proliferation of 3D reconstruction methods, such as structure-light based 3D scanning and image-based modeling, 3D reconstruction often requires expensive hardware and dedicated scanning environments. Thus, many research efforts have been devoted to 3D reconstruction using low-cost, consumer-level depth cameras [BF15, NIH*11, WZB14].

Although there are well-established 3D-scene reconstruction algorithms [NIH*11, NZIS13, WLS*15], we observe that existing methods frequently fail to achieve high-quality 3D object reconstruction results (see Fig. 2). The main reasons are twofold. First, many common homeware objects, such as shoes and toys of sizes

† Corresponding author: Weiwei Xu (xww@cad.zju.edu.cn)

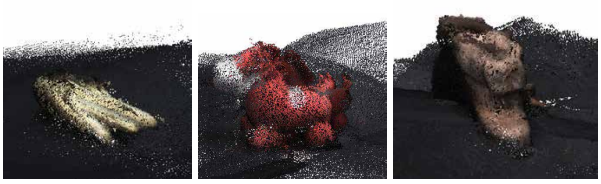

Figure 2: Existing 3D reconstruction techniques like KinectFusion [NIH*11] often fall short of high-quality reconstructions for small-scale objects.

roughly between $0.1-0.4$ meters, are small-scale w.r.t the depth range ($0.5-8.0$ meters) of depth cameras. Empirically, during the hand-held scanning, their shapes will only occupy a small portion of the depth image, usually fewer than 20% of the pixels of the image. This implies that conventional full-image based camera registration algorithms cannot concentrate on the target object, and irrelevant depth readings are likely to downgrade the quality of the reconstructed object. Second, even though one can move the depth camera near the target object to increase its resolution, the close-range data from the depth camera are often too noisy for a high-quality reconstruction of the 3D object.

As an echo to the above limitations, we propose a novel 3D reconstruction pipeline for commodity objects using consumer-level depth cameras, which integrates automatic object segmentation to improve the accuracy of the depth registration algorithm significantly and achieves high-quality, 360-degree 3D object reconstruction results regarding both geometry and texture. The technical components of our reconstruction pipeline include:

- *Automated object extraction from RGBD images*. To reduce or even eliminate the unwanted influence of the depth data irrelevant to a target object before 3D registration, our algorithm automatically segments out the object frame-by-frame and only keeps the depths of the object in the registration. To locate the target object in the scene at the beginning of scanning, the user only needs to point the center of the depth camera image to the target object for a short period, which eases the burden of manual intervention (e.g., manually put a 3D bounding box on the object at the beginning of scanning) often needed in current practices.
- *Online global non-rigid registration*. Unlike previous algorithms that only perform global registration after all the data are acquired or loop closures are detected, our algorithm performs online global non-rigid registration repeatedly during the scanning process. Specifically, our pipeline first registers each captured depth image rigidly frame-by-frame and organizes the rigidly-registered consecutive frames into fragments [ZMK13]. The global non-rigid registration is performed online whenever a new fragment is formed, which improves the accuracy of the estimated camera poses and removes the unnecessary influence from the non-linear distortion in the data. At this step, we adopt the embedded deformation (ED) graph algorithm [SSP07] with an additional photometric energy term. Due to its parallel implementation on GPU and the reduced computational cost at fragment level, the non-rigid registration step takes fewer than 70 milliseconds, which can efficiently support the interactive response of the pipeline.
- *Pause-and-restart for 360-degree reconstruction*. This scheme allows the user to pause the process after finishing the scanning of visible parts, change the object pose to uncover the occluded surface, and then restart the scanning. In this way, we are able to achieve high quality, 360-degree reconstruction results. At this step, we choose view-independent features (e.g., Combine ORB features [RRKB11] with FPFH features [RBB09]) to register the frames.

We have tested our reconstruction algorithm on a variety of 3D objects. As shown in Figs. 1 and 16, our algorithm can achieve high-quality textured 3D reconstruction results.

2. Related Work

Existing 3D reconstruction techniques include structure-light based 3D scanning and image-based multi-view 3D reconstruction, which have been extensively investigated in computer vision and computer graphics communities [HZ06, Wik]. Our work follows the fast development of 3D reconstruction technique using consumer-level RGB-D cameras, and we briefly review recent research efforts most related to our work in this section.

Rigid registration for depth fusion: Since the captured depth images are only partial scans of real-world object(s), it is critical to register them to a unified model to form a final 3D reconstruction result. At the early stage, researchers proposed to perform an iterative closest point (ICP) based rigid registration algorithm to register each depth image during 3D scanning [BM92, CM91], which has been developed to the widely-known *KinectFusion* technique [NIH*11]. To make the rigid registration procedure more robust, Kerl et al. [KSC13b, KSC13a] proposed to consider the color consistency during the registration, which can help to reduce the issue of geometry drifting. Endres et al. [EHE*12] use both geometric metrics and local image descriptors (SIFT [Low04], SURF [BETG08], or ORB [RRKB11]) to register RGB-D frames. The same idea was also explored in [DNZ*17, HKH*10]. In addition, Fast Point Feature Histogram (FPFH) [RBB09] and depth edges have also been adopted to enhance the registration accuracy [ZK15, ZPK16]. There also exist research works to improve the reconstruction quality via selecting sparse frames [YYFX18] and emphasizing salient objects in the fusion [SKM*17].

The registered frames can be fused into a volumetric representation which stores truncated sign distance function (TSDF) values at each voxel. Octree and spatial voxel hashing techniques are explored to reduce the memory footprint required in uniform volumetric representation in the reconstruction of large-scale scenes [WKF*12, NZIS13, KPM16, ZZZL12]. Roth et al. [RV12] developed a volume shifting and re-mapping technique to synchronize the volume with the movement of the depth camera. Along with the widely-used volumetric representation, the surface element technique *surfel*, a combination of point clouds and attached features, has been also proposed [WWLG09, WLS*15].

For 3D reconstruction tasks, loop closure is often desired to obtain good results. Many recent works use global optimization to correct the issue of accumulated drifting during scanning. Choi et al. [CZK15] proposed a robust off-line global optimization method on a graph consisting of all the fragment pairs. Based on a similar idea, Prisacariu et al. [KPM16] developed an online real-time

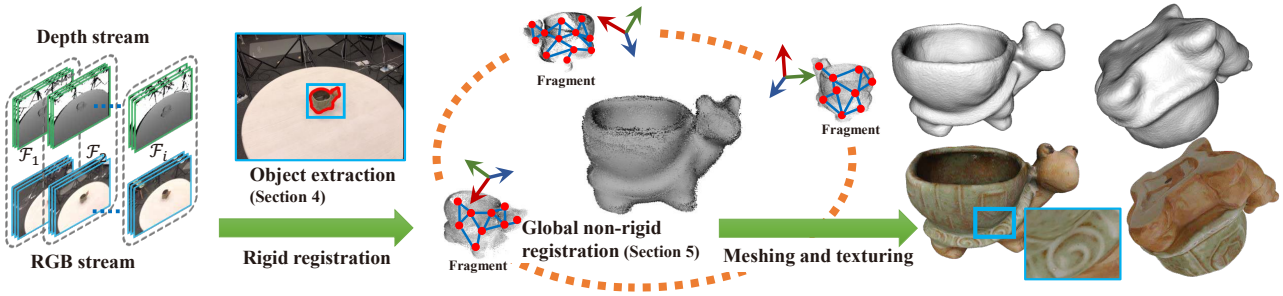

Figure 3: A schematic overview of our reconstruction pipeline.

system by dynamically creating sub-maps and performing the optimization of detected loop closures. Instead of correcting the drifted camera poses, Whelan et al. [WLS*15] register the drifted point clouds using embedded deformation graph [SSP07] on each detected local loop closure and global loop closure. A more recent work [DNZ*17], on the other hand, maintains the relation at frame level and performs global optimization in real time. It performs local pose optimization for each new frame and global pose optimization after a chunk of frames. Therefore, it can correct the drifting issue at the frame level.

Non-rigid registration for 3D reconstruction: The non-rigid registration is employed in 3D reconstruction to handle deformable objects and non-linear distortions in the captured depth data. Zollhöfer et al. [ZNI*14] use the as-rigid-as-possible surface modeling [SA07] method to do non-rigid surface fitting. The embedded deformation graph method developed in [SSP07] is widely used to track the motion of deformable objects [NFS15, DKD*16]. Guo et al. [GXW*18] proposed to use both L2 and L0 based motion regularizations to further stabilize the tracking. Our work is also related to the recent contributions that apply non-rigid deformation techniques to register the distorted point clouds to achieve the global geometry consistency [BR07, BR04, LPC*00]. In [WWLG09, WLS*15], global registration to reduce the gap at loop closures is achieved by the embedded deformation graph method.

Although the state-of-the-art non-rigid registration methods, such as dynamic fusion in [NFS15], can be combined with "Pause-and-restart" scheme to achieve 360-degree object reconstruction, we found our system improves the reconstruction quality greatly through online global non-rigid registration (see Fig.10 for a comparison).

Texture generation: To obtain high-quality object reconstruction, the texture consistency along with the geometry consistency is another critical aspect. Although the colors and depth sequences are registered before reconstruction, the texture inconsistency may exist between different views and between texture-geometry paring. Some previous efforts [CDG*13, CDPS09] designed for structure from motion (SFM) were proposed to obtain accurate camera poses according to the image-to-image or image-to-geometry consistency, and then build the final texture by blending the color images of all the views. For online scanning pipelines, Bornik et al. [BKB*01] use edge information to register the color images in different views, while Zhou et al. [ZK14] chose to directly optimize RGB camera poses to handle the possible inconsistency between depth and color images. They also proposed to use a 2D non-rigid deformation model to eliminate color image distortion. The recent work by Bi et al. [BKR17] uses a patch-based synthesis method to synthesize the texture for each scan with a content constraint and the inter-view consistency constraint, which can correct large misalignments in challenging cases. Other methods [LI07, WMG14] formulate the texturing problem as a labeling problem for each triangle using the Markov random field (MRF) model. After global color adjustments between the labeled patches, the final texture is packed into texture atlases. As a result, these two methods are suitable for high-resolution texture generation.

3. Overview

Given a captured RGBD image stream, our system first segments out the target object selected by the user and performs interleaved rigid registrations and online global non-rigid registration to reconstruct the 3D model. A *pause-and-restart* scheme can be used by the user to uncover the occluded parts of the target object to achieve its 360-degree reconstruction. Finally, we convert the reconstructed 3D model into a triangular mesh and compute its texture information to be the output of our pipeline. Fig. 3 illustrates the overall pipeline.

The 3D model during scanning is represented by *surfels*, where each surfel represents a tiny portion of surface that can be well approximated by a plane. It is similar to point cloud data but each surfel has five attributes, i.e. {position, normal, color, radius, and confidence}, which are updated by fusing overlapped 3D points from different depth maps. Please refer to [WWLG09] for the details of surfel fusion. We employ the surfel-based representation since it allows us to directly define the correspondences between 3D points in both rigid and global non-rigid registrations.

The global non-rigid registration is performed at fragment level to reduce the computational cost, inspired by the elastic fragments used in [ZMK13]. For a new captured RGBD image, our pipeline first performs an ICP-based rigid registration to align it with the previous frame and fuses the 3D points from the new image into surfels. A *fragment* is essentially 50 consecutive frames that have been rigidly registered. Each fragment can then be deformed using the embedded deformation graph method during the non-rigid registration process. To ease the non-rigid registration and the subsequent texture generation, a fragment in our system has the following attributes: 1) the surfels belonging to the fragment, 2) an embedded deformation graph [SSP07], 3) a keyframe, and 4) the camera pose corresponding to the keyframe. The keyframe for a

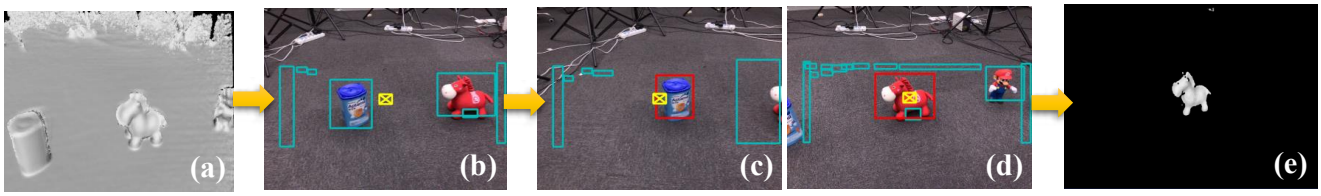

Figure 4: Object segmentation. (a) The rendered depth image from the surfel fusion. (b) The cyan bounding boxes computed from connected depth edges. (c)-(d) After the user moves the depth camera such that the yellow rectangle is on one of the objects, the overlapped bounding box will be drawn in red to inform the user. (e) The segmented object to inform the user to start the scanning.

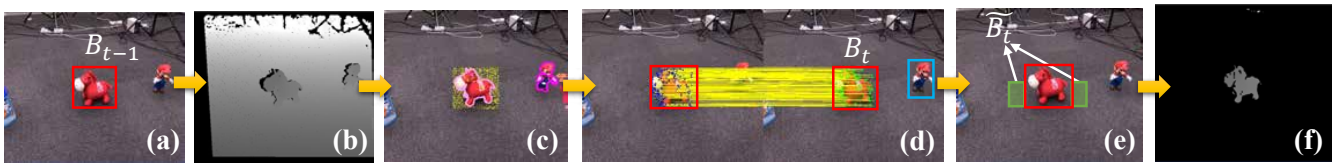

Figure 5: Object tracking. (a) The bounding box $\mathcal{B}_{t-1}$ and the computed ORB feature points indicated by the yellow dots. (b) The depth image of the next frame t. (c) The depth edge pixels and the ORB feature points. (d) The correspondences between the ORB feature points using the GMS method in [BLM*17]. (e) The extended window $\widetilde{\mathcal{B}}_t$ shown as green rectangles used to compute the supporting plane. (f) The final extracted object computed from the new bounding box $\mathcal{B}_t$.

fragment is selected to be the frame with minimal blurriness score proposed in [CDLN07]. A fragment graph, as shown in Fig. 6, is constructed to connect adjacent fragments, which are used to build the correspondences in global registration.

The triangular mesh in the final output is created using Poisson surface reconstruction [KBH06], where the maximum Octree depth is set to 9 to keep the details of the reconstructed model. The texture is generated by merging RGB images at the keyframes of fragments using the graph-cut method in [WMG14].

4. Automatic Object Extraction Using RGBD Images

We assume that objects are put on a plane (e.g., desk, ground) and do not contact with each other. At the beginning of the scanning, the user needs to point the camera center to the target object for a short period (e.g., 8 RGBD frames in our experiments) to select the target object. Afterward, during the scanning process, our system will automatically track the target object and extract its depth data frame-by-frame. Eliminating the irrelevant depth information through object extraction effectively improves the robustness and quality of the registration.

Initial segmentation: Initial segmentation is used to facilitate the user to select the target object to start the scanning. To this end, we compute a 2D bounding box for each object in the scene using the depth edge information. The pixels whose depth gradient magnitude is larger than a threshold (default 20 in our experiments) are deemed to be edge pixels, where the depth gradients are computed using the 3×3 Sobel operator. The edge pixels are grouped into connected components based on 8-nearest neighbors, and 2D bounding boxes are extracted for each connected component (Fig. 4 (b)) as the representation of objects. Those very small bounding boxes (with fewer than 10 pixel rows or columns) are removed. After the user points the center of the camera to the same bounding box for at least 8 frames, the object in the bounding box is selected to be the target object and segmented out using its contours formed by the depth edges in the box (Fig. 4 (e)).

To reduce the influence of the depth noise, we choose to use the fused depth data for the computation of depth edges. In addition, since our system requires the target object located at the center of the current view, we use a center window $\mathcal{W}^c$ of size 324×242 centered at the current view to calculate the depth range in which the target object should be contained. Specifically, we discard edge pixels in the depth image D whose depth values are not in the interval $[d_{min}, d_{max}]$, where d_{min} and d_{max} are respectively defined as:

$$\begin{aligned} d_{min} &= \min D(\mathbf{p}), \quad \mathbf{p} \in \mathcal{W}^c \\ d_{max} &= 2d_{avg} - d_{min}, \end{aligned} \tag{1}$$

where $\mathbf{p} \in \mathbb{R}^2$ is a pixel in D, and $d_{avg} = \sum D(\mathbf{p})/|\mathcal{W}|, \mathbf{p} \in \mathcal{W}$ is the average depth within $\mathcal{W}$.

Object tracking: It is used to transfer the bounding box $\mathcal{B}_{t-1}$ at frame $t-1$ to the current frame t, and then segment out the target object at frame t. We first compute ORB feature points $\mathcal{O}_{t-1}$ at the dilated object bounding box $\vec{\mathcal{B}}_{t-1}$ according to the RGB frame I_{t-1} and its associated depth frame D_{t-1}. At frame t, we also use connected depth edges to obtain candidate bounding boxes and then select the boxes based on the correspondences computed between $\mathcal{O}_{t-1}$ and $\mathcal{O}_t$ [BLM*17]. Those selected boxes are merged into one bounding box $\mathcal{B}_t$ as the new one. Only those candidate bounding boxes that receive more than five correspondences are selected to be merged into the new bounding box. Similarly, to improve the tracking performance, we only detect edge pixels and ORB feature points at frame t for depth pixels in the dilated bounding box $\vec{\mathcal{B}}_{t-1}$. Fig. 5 illustrates the procedure of ORB feature point tracking and the computation of a new bounding box.

However, depth data inside the bounding box could still contain the depths associated with the supporting plane, which should

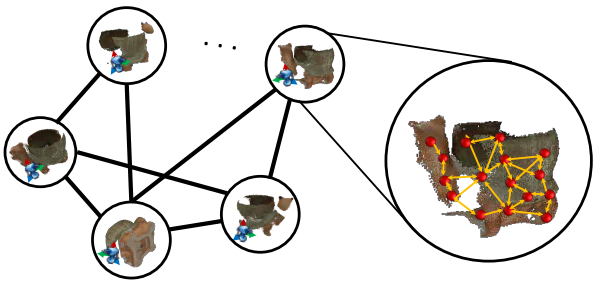

Figure 6: (Left) Fragments are organized into a fragment graph. Two fragments are connected in the graph if their camera orientations are similar. (Right) Each fragment has an embedded deformation graph, which parameterizes the underlying deformation of the fragment. The red spheres indicate the deformation graph nodes selected from the surfels of the fragment.

be removed to facilitate the reconstruction and registration. It is done by detecting the plane according to the tracked bounding box. Specifically, given a tight bounding box $\mathcal{B}_t$, we extend the bottom half of this region laterally by 25% at both vertical sides of the original bounding box, yielding an extended window $\widetilde{\mathcal{B}}_t$ indicated by two green boxes in Fig. 5(e). Presumably, $\widetilde{\mathcal{B}}_t$ provides the high-confidence information of the current supporting plane if the view does not parallel to the supporting plane. For a pixel $\mathbf{p}$ in the depth image D, we can obtain its 3D vertex $\mathbf{v} \in \mathbb{R}^3$ by $\mathbf{v}(\mathbf{p}) = D(\mathbf{p})\mathbf{K}^{-1}[\mathbf{p}^\top, 1]^\top$, where $\mathbf{K}$ is the camera calibration matrix. The supporting plane should cross the set of vertices $\mathbf{v}$ in $\widetilde{\mathcal{B}}_t$ and its normal $\mathbf{n}$ can be estimated using Principal Component Analysis (PCA). After the supporting plane for the current frame is obtained, the vertices in the bounding box $\mathcal{B}_t$ which are above the supporting plane is extracted as the vertices of the object as shown in Fig. 5(f).

5. Online Global Non-rigid Registration

To obtain high-quality reconstruction results, we adopt the ED method [SSP07], an efficient shape deformation framework, to capture the geometry distortion of the object induced by camera drifting. Each non-rigid registration seeks for the optimal deformation across all the RGBD images received. In our implementation, we detect the overlapping area among all the fragments and build correspondences accordingly so that their spatial distance and photometric difference can be minimized. Loop closure is thus naturally achieved by registering the overlapping fragment pairs.

5.1. Fragment Graph

The fragment graph $\mathcal{G}\langle\mathcal{N},\mathcal{E}\rangle$ determines the correspondences needed in the global non-rigid registration, where the note set $\mathcal{N}$ is a set of fragments and the edge set $\mathcal{E}$ stands for their connections, as shown in Fig. 6. If two nodes (fragments) are not connected via an edge in $\mathcal{G}$, the two fragments do not have their correspondences. The connectivity between two fragments is determined by the camera orientations of the keyframes from the two fragments. Specifically, let $\mathbf{T}_i, \mathbf{T}_j \in \mathbb{SE}(3)$ be the keyframe camera poses for fragments i and j, respectively. They are connected in $\mathcal{G}$ if the following condition holds:

$$\arccos\left([\mathbf{T}_i]_3 \cdot [\mathbf{T}_j]_3\right) < \gamma. \tag{2}$$

$[\mathbf{A}]_i$ stands for the i^{th} column vector of matrix $\mathbf{A}$, and $[\mathbf{T}_i]_3$ is the local optical axis of poses i. The threshold angle γ is set as 120° in our experiments and it is not critical for our system as more connections are also acceptable. We found in our experiments that using such a simple inner product based metric suffices to quantify the similarity between different camera poses. This is because during the scanning process, the camera generally faces to the object and the object is relatively small (Figure 7). For each pair of adjacent fragments $\mathcal{F}_i$ and $\mathcal{F}_j$, we maintain a set of matching vertices $\mathcal{M}_{i,j}$, which is computed by projecting the depth of one fragment to the other (from new to old in our implementation). The matched vertices are recorded as the overlapping vertices of $\mathcal{F}_i$ and $\mathcal{F}_j$ and need to be re-calculated whenever either fragment is updated.

5.2. ED-based Optimization

Once a fragment $\mathcal{F}_i$ is received, we construct an embedded deformation graph [SSP07] containing a set of vertices $\mathcal{S}_i$ as the nodes for the deformation graph, as indicated by the red spheres in the right of Fig. 6. We set $|\mathcal{S}_i| = 16$ in our implementation, which is selected using systematic sampling. Each non-node vertex in $\mathcal{F}_i$ is connected with four nearest node vertices in $\mathcal{S}_i$, and each node vertex is connected with eight nearest node vertices. Such a dense connection setup effectively avoids the singularity issue during the follow-up optimization.

Figure 7: The camera's trajectory for the upper half of the shoe.

A node vertex possesses its original position $\mathbf{g}_i \in \mathbb{R}^3$, a local rotation $\mathbf{R}_i \in \mathbb{SO}(3)$, and a local translation $\mathbf{t}_i \in \mathbb{R}^3$. $\mathbf{R}_i$ is parameterized using exponential map that maps a vector $\boldsymbol{\theta}_i \in \mathbb{R}^3$ to a rotation matrix, namely, $\mathbf{R}_i = \exp(\boldsymbol{\theta}_i)$. The deformed position and normal of a non-node vertex in $\mathbf{v}_i^k$ are estimated through weighted blending of the transformations of its four associated node vertices $\mathbf{v}_i^l$, $l \in \{1,2,3,4\}$ as follows:

$$\tilde{\mathbf{v}}_i^k = \sum_l w_i^l(\mathbf{v}_i^k)\left[\mathbf{R}_i^l(\mathbf{v}_i^k - \mathbf{g}_i^l) + \mathbf{g}_i^l + \mathbf{t}_i^l\right], \tag{3}$$

and

$$\tilde{\mathbf{n}}_i^k = \sum_l w_i^l(\mathbf{v}_i^k)\mathbf{R}_i^l\mathbf{n}_i^l. \tag{4}$$

$w_i^l(\mathbf{v}_i^k)$ is the interpolation weight computed as:

$$w_i^l(\mathbf{v}_i^k) = \frac{\left(1 - \left\|\mathbf{v}_i^k - \mathbf{g}_i^l\right\| / d_{max}\right)^2}{\sum_{n=1}^{4}\left(1 - \left\|\mathbf{v}_i^k - \mathbf{g}_i^n\right\| / d_{max}\right)^2}, \tag{5}$$

where d_{max} is the distance between $\tilde{\mathbf{v}}_i^k$ and its nearest non-node vertex. Here, we use the superscript $[\cdot]^k$ to denote the vertex index and the subscript $[\cdot]_i$ is the fragment index. The nonlinear registration seeks for the optimal rotation and translation at each node vertex

so that the geometric and photometric inconsistencies across all the fragment pairs, $\{(\mathcal{F}_i,\mathcal{F}_j)|\langle i,j\rangle \in \mathcal{E}\}$, are minimized. Our objective energy function consists of the following three components:

$$E = \lambda_{geo}E_{geo} + \lambda_{photo}E_{photo} + \lambda_{smooth}E_{smooth}. \tag{6}$$

Weights λ_{geo}, λ_{photo}, λ_{smooth} are primarily for scaling each energy term to the same range, and they are set as 10, 0.001 and 100 in our implementation, respectively.

The **geometric energy** (E_{geo}) encodes the geometric inconsistency based on the point-to-plane distance as:

$$E_{geo} = \sum_{\langle i,j\rangle\in\mathcal{E}} \sum_{k_1,k_2\in\mathcal{M}_{i,j}} \left[\tilde{\mathbf{n}}_i^{k_1}\cdot(\tilde{\mathbf{v}}_j^{k_2}-\tilde{\mathbf{v}}_i^{k_1})\right]^2 + \left[\tilde{\mathbf{n}}_j^{k_2}\cdot(\tilde{\mathbf{v}}_i^{k_1}-\tilde{\mathbf{v}}_j^{k_2})\right]^2, \tag{7}$$

assuming $\tilde{\mathbf{v}}_i^{k_1} \in \mathcal{F}_i$ and $\tilde{\mathbf{v}}_j^{k_2} \in \mathcal{F}_j$ are two matched vertices from the associated matching vertices set $\mathcal{M}_{i,j}$.

The **photometric energy** (E_{photo}) is designed to describe the mutual projection difference in the RGB color space between two connected fragments. Mathematically, it is formulated as:

$$E_{photo} = \sum_{\langle i,j\rangle\in\mathcal{E}} \sum_{k_1,k_2\in\mathcal{M}_{i,j}} \left\| I_j\big(\pi(\mathbf{T}_j^{-1}\tilde{\mathbf{v}}_i^{k_1})\big) - I_i\big(\pi(\mathbf{T}_i^{-1}\tilde{\mathbf{v}}_j^{k_2})\big)\right\|^2. \tag{8}$$

Similar to Eq. (7), $\tilde{\mathbf{v}}_i^{k_1} \in \mathcal{F}_i$ and $\tilde{\mathbf{v}}_j^{k_2} \in \mathcal{F}_j$ are two matched vertices from $\mathcal{M}_{i,j}$. First, $\tilde{\mathbf{v}}_i^{k_1}$ is transformed to the camera space, determined by the key frame camera pose of $\mathcal{F}_j$ as $\mathbf{T}_j^{-1}\tilde{\mathbf{v}}_i^{k_1}$. π is the camera projection operation such that:

$$\pi([x,y,z]^\top) = [\frac{xf_x}{z}+c_x, \frac{yf_y}{z}+c_y]^\top, \tag{9}$$

where f_x, f_y are the camera's focal lengths and c_x, c_y are the optical center coordinates. Next, we project the transformed $\tilde{\mathbf{v}}_i^{k_1}$ onto I_j, which is the key RGB frame of $\mathcal{F}_j$. The color is computed using a bi-linear interpolation based on the projected image coordinates. Similarly, the color value of $\tilde{\mathbf{v}}_j^{k_2}$ is also evaluated in I_i and the squared color differences are summed.

The **smooth energy** (E_{smooth}) is a regularization term as suggested in [SSP07]. E_{smooth} penalizes sharp high-frequency deformations, which can be calculated as:

$$E_{smooth} = \sum_i \sum_k \left\| \sum_l \mathbf{R}_i^l(\mathbf{g}_i^k - \mathbf{g}_i^l) + \mathbf{g}_i^l + \mathbf{t}_i^l - (\mathbf{g}_i^k + \mathbf{t}_i^k)\right\|^2, \tag{10}$$

where the first summation iterates all the fragments. For a node vertex k on the embedded deformation graph of $\mathcal{F}_i$, Eq. (10) sums the squared displacement as it is plugged into the transformation field at its neighboring node l.

5.3. Camera Poses Update

The online global non-rigid registration can reduce scan drifting by both registering the fragments and rectifying the current camera pose. Therefore, an important step in our system is to update the camera poses for all the fragments' keyframes according to the non-rigid registration result during scanning, since it is not directly optimized in the embedded deformation algorithm. To this end, we consider the camera as a "super vertex" deformed by all the nodes in the embedded graph. Precisely, its new pose, denoted by $\tilde{\mathbf{T}}_i^{cam} = \{\tilde{\mathbf{R}}_i^{cam}, \tilde{\mathbf{t}}_i^{cam}\}$, is computed as the average of all the node vertices on the embedded graph:

$$\tilde{\mathbf{R}}_i^{cam} = \exp\left(\sum_k \log\left(\mathbf{R}_i^k \mathbf{R}_i^{cam}\right)/16\right), \tag{11}$$

where the camera's rotation $\tilde{\mathbf{R}}_i^{cam}$ is updated by averaging the local rotation $\mathbf{R}_i^k$ of all of the 16 node vertices based on the initial camera pose $\mathbf{R}_i^{cam}$. Likewise, the camera translation is updated by averaging all the $\mathbf{t}_i^k$:

$$\tilde{\mathbf{t}}_i^{cam} = \sum_k \left[\mathbf{R}^k(\mathbf{t}_i^{cam} - \mathbf{g}_i^k) + \mathbf{g}_i^k + \mathbf{t}_i^k\right]/16. \tag{12}$$

With the updated camera poses for keyframes, the current camera pose is updated accordingly by calculating a rigid transformation related to its corresponding keyframe camera pose.

5.4. Optimization

During the optimization each loop updates the matching vertices set $\mathcal{M}_{i,j}$ for each connected fragment pair $\mathcal{F}_i$ and $\mathcal{F}_j$, $\langle i,j\rangle \in \mathcal{E}$, computes an incremental correction towards current deformation parameters, updates the fragments' geometry using Eqs. (3) and (4), and adjusts the camera pose for each fragment.

The Gauss-Newton method is employed in our implementation and the following linear system is solved at each iteration:

$$\left(\mathbf{J}^\top\mathbf{J}\right)\delta\mathbf{x} = -\mathbf{J}^\top\mathbf{r}, \tag{13}$$

where the unknown vector $\mathbf{x}$ stacks deformation parameters on the embedded deformation graph for all the fragments. As we have 6 parameters per node vertex and 16 nodes per embedded graph, $\mathbf{x}$ is a $(6\cdot16\cdot|\mathcal{F}|)$-dimensional column vector. During scanning $|\mathcal{F}|$ will increase to typically between 30 to 50 in our experiments. $\mathbf{r}$ is the residual that is updated after each iteration. The Jacobi matrix $\mathbf{J}$ is $2\cdot\sum|\mathcal{M}_{i,j}| + 3\cdot16\cdot|\mathcal{F}|$ by $6\cdot16\cdot|\mathcal{F}|$ and is sparse because of the local property of ED graph. To reduce the computational workload, we down-sample each fragment to 512 vertices when assembling $\mathcal{M}_{i,j}$ to reduce the cardinality of $\mathcal{M}_{i,j}$ and speed up the optimization.

$\tilde{\mathbf{n}}_i^{k_1}$ and $\tilde{\mathbf{n}}_j^{k_2}$ in Eq. (7), and $\mathbf{T}_i$ and $\mathbf{T}_j$ in Eq. (8) are treated as constants at a given iteration, so that E_{geo} becomes standard quadratic forms of $\mathbf{x}$, and the corresponding matrix sub-block in $\mathbf{J}$ can be computed trivially. For E_{photo}, the partial derivative of I with respect to $\mathbf{x}$ is computed via the chain rule:

$$\frac{\partial I}{\partial \mathbf{x}} = \frac{\partial I}{\partial \pi}\cdot\frac{\partial \pi}{\partial [x,y,z]^\top}\cdot\frac{\partial [x,y,z]^\top}{\partial \mathbf{x}}. \tag{14}$$

The last two partial derivatives are straightforward. The derivative $\partial I/\partial\pi$ is calculated by applying a normalized Scharr kernel over I. We convert the gray values of the image to floats and apply a Gaussian filter to smooth the variation of the gray values on I to avoid local gradient vanishing.

Eq. (13) is solved using the Jacobi-preconditioned conjugate gradient (PCG) method. The PCG loop terminates when the residual is sufficiently small (e.g., below a threshold), which takes typically

50-100 iterations, and the deformable registration needs fewer than five non-linear iterations to converge.

GPU implementation: We implement the PCG solver in parallel on GPU. Since the dimension of the Jacobi matrix $\mathbf{J}$ in Eq. (13) could become very large with the increase of the number of edges in the fragment graph, calculating $\mathbf{J}^\top\mathbf{J}$ will be time-consuming after $\mathbf{J}$ is obtained. Instead, we choose to directly calculate the $\mathbf{J}^\top\mathbf{J}$, similar to Fusion4D [DKD*16]. In the ED graph, a non-node vertex is only affected by its 4 neighboring nodes in the data term, and a node is only affected by its 8 neighboring nodes in the smooth term. With such characteristics, the resulting $\mathbf{J}^\top\mathbf{J}$ is a sparse matrix composed of 6×6 dense sub-blocks, each of which is only affected by a limited number of vertex-to-node and node-to-node energy derivatives. In our implementation, we launch one CUDA block for each dense sub-block and use the threads in each CUDA block to calculate the related vertex-to-node and node-to-node energy derivatives concurrently. Each CUDA block will fill its sub-block after multiplying these energy derivatives with each other and reduce them together. This parallel algorithm to directly calculate $\mathbf{J}^\top\mathbf{J}$ can significantly reduce the computational time while reducing the GPU memory usage. Similarly, $\mathbf{J}^\top\mathbf{r}$ can also be calculated directly.

6. Pause-and-restart for 360-degree Reconstruction

However, to achieve pause-and-restart, using the frame-to-model ICP based rigid registration method is insufficient because both the camera pose and the object pose have been changed. For these cases, we choose to use view-independent features to register the depth frame "after restart" to the model "before pause".

After the user pauses then restarts the scanning, our system first extracts the ORB feature points [RRKB11] and the FPFH feature points [RBB09] separately on the current color and depth frames. Then, the correspondences are calculated between these features and the same features extracted from the projected color map and depth map at the camera pose "before pause". As shown in Fig. 8, the correspondences are calculated using the HAMMING distance for the ORB features and the Euclidean distance for the FPFH features, and we only choose the nearest neighbor in our system. By combining the ORB features with the FPFH features, the registration process is robust even for objects without rich texture.

After the extraction of the feature correspondences, candidate matching points between the depth image and the model are obtained. Then we implement a random sample consensus (RANSAC) method to register the depth image to the current model by employing the following scheme:

1. Select 6 sample points randomly while ensuring that their pairwise distances are greater than a user-defined minimum distance (4 pixels in our implementation).
2. Compute the rigid transformation defined by the sample points and their correspondences.
3. Compute the re-projection error between the point cloud and the depth image.

The RANSAC method with GPU implementation stops after $1,000$ iterations, and it takes less than $1ms$ for each iteration. The rigid transformation with minimal re-projection error is used to register the depth frame "after restart" to the model "before pause".

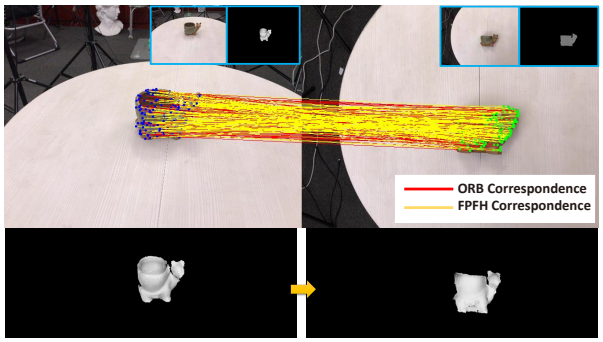

Figure 8: Based on ORB feature and FPFH feature correspondences and the RANSAC method, the model "before pause" and depth image "after restart" can be registered correctly.

7. Experimental Results

We implemented our approach on a desktop computer equipped with an `Intel i-7 7700K` 4.2 GHz CPU and an `NVIDIA GTX 1080Ti` GPU with 11G GDDR5 video memory. The depth cameras used in our experiments are Occipital structure sensor and Xiton Pro, and their depth resolution is set to be 640×480. The structure sensor is installed on an iPad Pro equipped with a color sensor with 2592×1936 resolution. We have extensively tested our system as various objects as reported in Table 1.

Comparison with KinectFusion and ElasticFusion: The comparison results are shown in Fig. 9. It can be seen that the ICP-based registration in Kinectfusion is not capable of handling nonlinear shape distortions, and errors get accumulated quickly. Such a problem will be exaggerated when the target is a small object. Similarly, ElasticFusion also focuses on 3D reconstruction of large-scale scenes. While ElasticFusion allows deformable registration operations, they are handled locally or incrementally in most cases to save the computational cost. Therefore, we still can observe accumulated misalignments in the final results by them. As aforementioned, our method is specially tuned for 3D object reconstruction. We allow a more aggressive registration strategy that performs a global deformable registration frequently (every 50 frames when a fragment is formed). Each ED-based registration not only fuses a batch of new depth readings into the model but also aims to correct the current residual error and seek for a better global configuration.

With and without object extraction: One of the main advantages of our method is the automated object extraction step that can locate the target object with minimal user intervention and isolate the depth pixels around the object. The advantage of this process is also reported in Fig. 9. No matter one chooses to use locally rigid registration (KinectFusion) [NIH*11], local-global deformable registration (ElasticFusion) [WLS*15], or global deformable registration (our method), removing irrelevant depth readings will always improve the reconstruction quality noticeably.

Comparison with DynamicFusion and ElasticFragment: Although the dynamic fusion method in [NFS15] also employs the embedded deformation graph method to register depth images, it lacks an explicit global registration step to handle loop closures as the global non-rigid registration of fragments in our method.

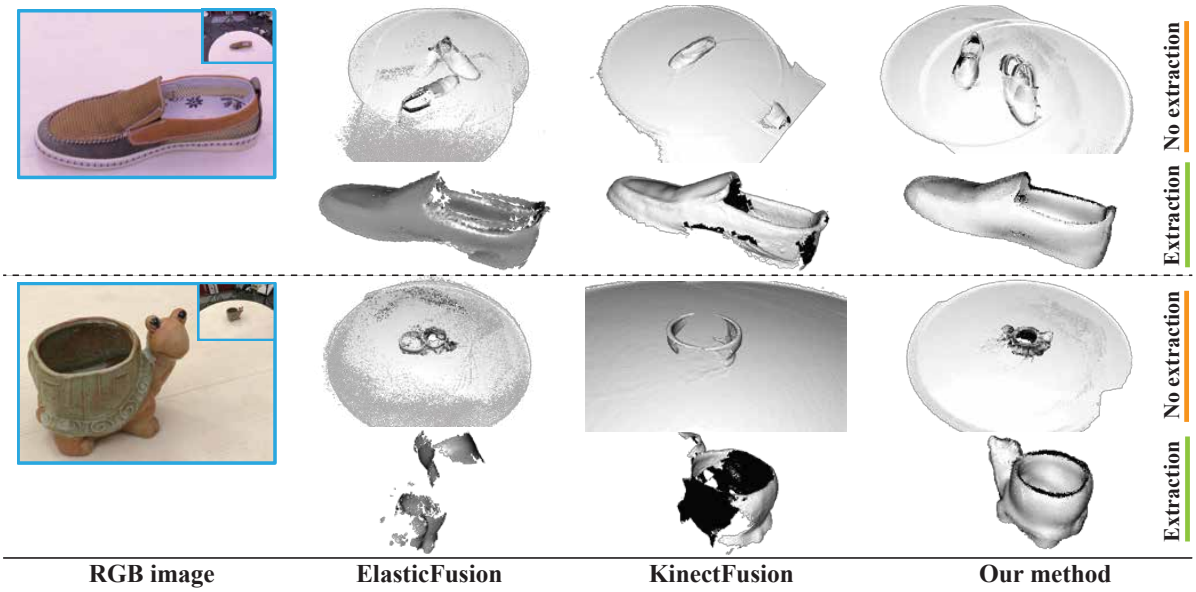

Figure 9: We compare the reconstruction results using KinectFusion [NIH*11], ElasticFusion [WLS*15] and our method. Clearly, performing objection extraction effectively improves the resulting 3D models.

Model	# of frames (fragments)	Size (cm)	Object extraction	Fusion & rigid registration	Global non-rigid registration	Meshing & texturing
Formular canister (Fig. 16)	2400 (48)	$13 \times 11 \times 20$	13.21s/5.5ms	48.97s/20.4ms	1.94s/0.8ms	37.84s
Detergent bottle (Fig. 16)	2400 (48)	$15 \times 8 \times 19$	13.33s/5.6ms	49.32s/20.6ms	1.97s/0.8ms	38.65s
Toy dinosaur (Fig. 16)	2100 (42)	$41 \times 11 \times 15$	11.67s/5.6ms	42.45s/20.2ms	1.56s/0.9ms	30.05s
Casual shoe (Fig. 16)	1600 (32)	$28 \times 11 \times 10$	8.64s/5.4ms	32.87s/20.5ms	1.03s/0.6ms	34.47s
Sports shoe (Fig. 1)	1450 (29)	$29 \times 12 \times 11$	8.05s/5.6ms	29.69s/20.5ms	0.88s/0.6ms	38.69s
Boot (Figure 16)	2200 (44)	$25 \times 9 \times 17$	12.34s/5.6ms	44.01s/20.0ms	1.68s/0.8ms	29.07s
Pot turtle (Fig. 3)	2350 (47)	$15 \times 10 \times 9$	12.93s/5.5ms	47.50s/20.2ms	1.86s/0.8ms	33.34s
Toy Mario (Fig. 16)	2500 (50)	$13 \times 8 \times 23$	13.45s/5.4ms	49.94s/20.0ms	2.05s/0.8ms	34.38s
Toy monkey (Fig. 14)	1300 (26)	$17 \times 13 \times 27$	6.87s/5.3ms	26.79s/20.6ms	0.76s/0.6ms	32.56s
Banana (Fig. 1)	1900 (38)	$25 \times 20 \times 6$	10.33s/5.4ms	35.12s/18.5ms	1.33s/0.7ms	35.77s
Toy horse (Fig. 1)	1700 (34)	$25 \times 12 \times 23$	9.66s/5.7ms	34.35s/20.2ms	1.13s/0.7ms	37.15s

Table 1: Statistics of reported 3D reconstruction results. **# of frames (fragments)** indicates the number of frames and the number of formed fragments in the captured RGB-D sequences for all the tested objects. Their dimensions in centimeter is recorded in columns, **Size**. The 4th to 6th columns are the total time in seconds and time per frame in millisecond for three steps: 1) object extraction, 2) surfel fusion and rigid registration, 3) global non-rigid registration. The last column is the total time in seconds for meshing and texturing.

As illustrates in Fig. 10, our pipeline achieves better 3D reconstruction results of the same 3D objects in Fig. 9. The results of the dynamic fusion method are also obtained after the objects are segmented out from the depth images. Our online, fragment-based non-rigid registration can be viewed as a GPU implementation of the offline elastic fragment method in [ZMK13]. However, the advantage of our method is that the frame-by-frame registration can be corrected by the global registration online in the reconstruction. Fig. 11 shows a comparison of the online and offline registration results. Both methods produce high-quality registration results in the experiment, and the result of the elastic fragment method is obtained using the implementation at `https://github.com/qianyizh/ElasticReconstruction`.

Global rigid registration vs global non-rigid registration: Next, we show that non-rigid registration is also an important recipe of a high-quality reconstruction even for still objects. For this purpose, we replace our deformable registration subroutine with a standard rigid registration. While global rigid registration appears to yield a plausible result at the first sight, a zoom-in view tells us otherwise. As reported in Fig. 12, many vertices are ill-registered to the object's surface due to the nonlinear deformation during the scan. Such mismatched vertices visually blur the model and they could lead to dangling triangles on the final mesh. On the other hand, our ED-based non-rigid registration greatly improves the result. As we can see from Fig. 12, most vertices are tightly aligned to the object surface.

With and without photographic metric: Another important ingredient is the photometric energy E_{photo} term in the objective

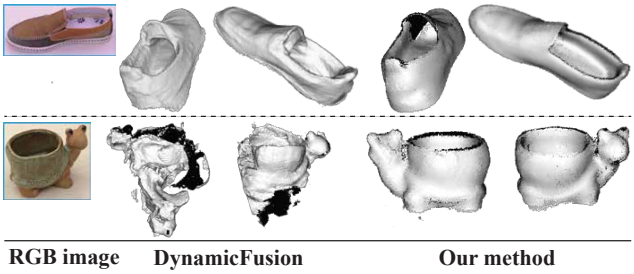

Figure 10: Comparison with DynamicFusion [NFS15].

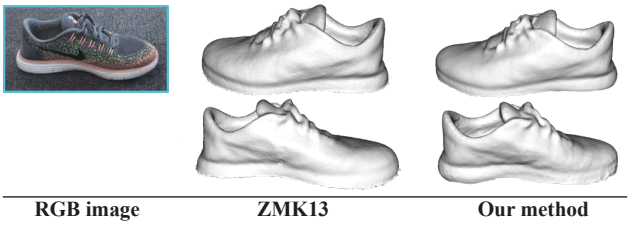

Figure 11: Comparison with ElasticFragment [ZMK13].

function defined in Eq. (8). The geometry metric only captures the local shape variation between fragments. When a fragment does not have strong geometric features, for instance, a flat thin board would be an extreme case, it is possible that vertices are actually misregistered even if the corresponding geometric energy is low. An example can be seen in Fig. 13. Without employing E_{photo}, the fragments are misaligned by a translational displacement even with a low E_{geo} value. After plugging E_{photo}, the registration re-position the fragment to fit the consistency in the RGB color space. The quality of the result, especially for texturing the model, is improved.

System performance: Our implementation makes significant use of GPU computing. The surfel fusion was implemented using the `OpenGL Shading Language`, while the object extraction, camera pose tracking, and ED-based global non-rigid registration were implemented using `CUDA`. For a typical small object with 30 to 50 fragments, the total GPU memory footprint is around 1.5 gigabytes. The time used for each module during scanning is reported in table 1. With the GPU implementation of the GMS method [BLM*17], the automatic object extraction can be done within 5-6 milliseconds per frame. The ICP based rigid registration between 2 frames can finish within 13-15 milliseconds, and the surfel fusion took 4-5 milliseconds. The computational time for the global non-rigid registration depends on the number of fragments in the input data. At the beginning, it is around 20 milliseconds for 10 fragments and will increase to 60-70 milliseconds for 50 fragments. As the global non-rigid registration is only called when a new fragment arrives, our approach can still run at 30fps using two threads: a data thread responsible for recording RGBD frames and a registration thread for object extraction and non-rigid registration. For "pause and restart" during scanning, typically it took about 0.8-1.0 seconds for the RANSAC method, and our system discarded frames captured in this period.

More results: Besides the examples shown in Fig. 1, more textured reconstruction results can be found in Fig. 16. It can be seen

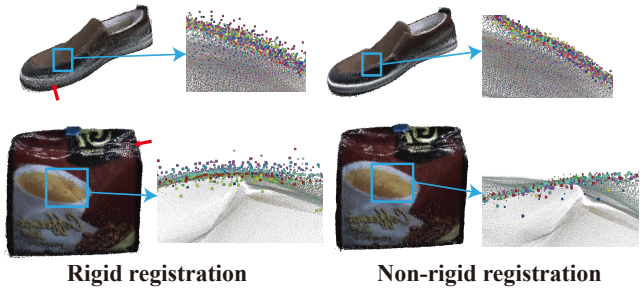

Figure 12: Even for still objects, non-rigid registration is important. Using only global rigid registration produces many "diffusive" and ill-aligned vertices due to the nonlinear distortion. Our ED-based deformable registration yields better results

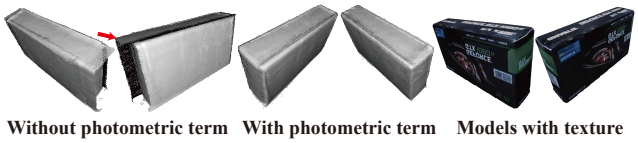

Figure 13: The photometric energy term resolves the registration ambiguity during the reconstruction especially for object's fragment without sharp geometric features.

that our system can produce high-quality 3D reconstruction for small objects using Occipital structure sensor. The photometric metric further improves the texture quality. Even the ingredient information on the formula canister is legible. The RGB images and depth map captured by the structure sensor installed on the iPad Pro is sent back to the desktop PC for segmentation and registration via WiFi network. Moreover, we also tested our algorithm using Xtion Pro depth camera. As shown in Fig. 14, the reconstructed objects are also of high quality. We also measured the geometric reconstruction error of our method using a synthetic chair model. The chair model is normalized into a bounding box of 1-meter side length and rendered into depth images using virtual camera trajectories. As shown in Fig. 15, the maximal Hausdorff distance between the original chair model and the reconstructed model is 0.0037m (the relative reconstruction error is thus 0.37%), and the points with large errors mostly appear along the sharp edges of the model, where the surfel fusion procedure might smooth their positions.

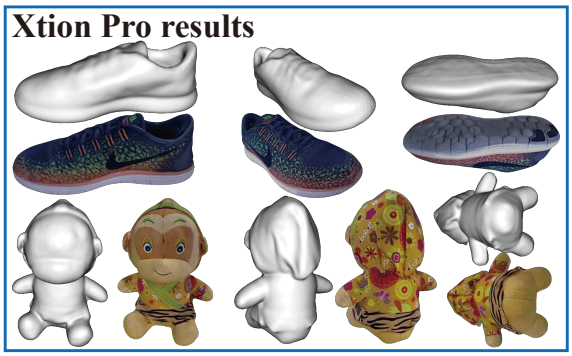

Figure 14: Using Xtion Pro depth camera, our system can also produce high-quality 3D reconstruction.

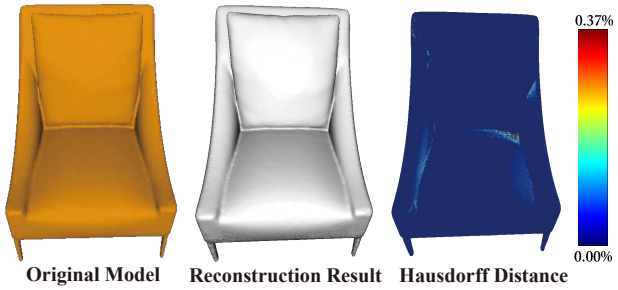

Figure 15: Relative reconstruction error measured using a synthetic chair model.

8. Discussion and Conclusion

In this paper, we present an efficient pipeline specially crafted for 3D object reconstruction. Since we focus on 3D object reconstruction, the proposed pipeline isolates the target object's depth information. Targeting small-scale objects also allows us to choose more aggressive registration strategies. Our system couples an ED-based deformable registration with a rigid registration. The deformable registration is performed over all the received fragments to correct the residual error, and it well captures the nonlinear distortion of the object, and the entire optimization routine is implemented on the GPU. All of these technical features converge to a stable, efficient, and high-quality 3D reconstruction system.

Limitations and future work: Nevertheless, our current system still has many limitations that enlight us several exciting future directions to explore. We currently assume the object is placed on an open area without occlusions from other objects. How to extend our system to deal with multiple objects with potential occlusions is an interesting and challenging future direction we would like to pursue. There exist opportunities to minimize the data transfer between CPU and GPU, which is the speed bottleneck of our algorithm. We also want to study better nonlinear optimization procedures on GPU as PCG heavily relies on vector inner product, which is not superior on GPU.

Acknowledgement

We would like to thank the anonymous reviewers for their constructive comments. Weiwei Xu is partially supported by National Key R&D Program of China (2017YFB1002600), NSFC (No. 61732016), and fundamental research fund for the central universities (2017XZZX009-03). Yin Yang is supported in part by NSF 1717972 and AFRL A18-0460. Zhigang Deng is partially supported by NSF IIS-1524782.

References

[BETG08] BAY H., ESS A., TUYTELAARS T., GOOL L. J. V.: Speeded-up robust features (SURF). *Computer Vision and Image Understanding, CVIU, 110*, 3 (2008), 346–359. 2

[BF15] BECK S., FRÖHLICH B.: Volumetric calibration and registration of multiple rgbd-sensors into a joint coordinate system. In *IEEE Symposium on 3D User Interfaces, 3DUI ,* (2015), pp. 89–96. 1

[BKB*01] BORNIK A., KARNER K. F., BAUER J., LEBERL F., MAYER H.: High-quality texture reconstruction from multiple views. *Journal of Visualization and Computer Animation, JVCA, 12*, 5 (2001), 263–276. 3

[BKR17] BI S., KALANTARI N. K., RAMAMOORTHI R.: Patch-based optimization for image-based texture mapping. *ACM Transactions on Graphics, TOG, 36*, 4 (2017), 106:1–106:11. 3

[BLM*17] BIAN J., LIN W., MATSUSHITA Y., YEUNG S., NGUYEN T., CHENG M.: GMS: grid-based motion statistics for fast, ultra-robust feature correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR,* (2017), pp. 2828–2837. 4, 9

[BM92] BESL P. J., MCKAY N. D.: A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence, TPAMI, 14*, 2 (1992), 239–256. 2

[BR04] BROWN B. J., RUSINKIEWICZ S.: Non-rigid range-scan alignment using thin-plate splines. In *International Symposium on 3D Data Processing, Visualization and Transmission, 3DPVT,* (Sept 2004), pp. 759–765. 3

[BR07] BROWN B. J., RUSINKIEWICZ S.: Global non-rigid alignment of 3-d scans. *ACM Transactions on Graphics, TOG, 26*, 3 (2007), 21. 3

[BRU10] From 3d reconstruction to virtual reality: A complete methodology for digital archaeological exhibition. *Journal of Cultural Heritage 11*, 1 (2010), 42 – 49. 1

[CDG*13] CORSINI M., DELLEPIANE M., GANOVELLI F., GHERARDI R., FUSIELLO A., SCOPIGNO R.: Fully automatic registration of image sets on approximate geometry. *International Journal of Computer Vision, IJCV, 102*, 1-3 (2013), 91–111. 3

[CDLN07] CRETE F., DOLMIERE T., LADRET P., NICOLAS M.: The blur effect: perception and estimation with a new no-reference perceptual blur metric. 64920I. 3

[CDPS09] CORSINI M., DELLEPIANE M., PONCHIO F., SCOPIGNO R.: Image-to-geometry registration: a mutual information method exploiting illumination-related geometric properties. *Computer Graphics Forum, CGF, 28*, 7 (2009), 1755–1764. 3

[CM91] CHEN Y., MEDIONI G.: Object modeling by registration of multiple range images. In *IEEE International Conference on Robotics and Automation, ICRA,* (1991), pp. 2724–2729 vol.3. 2

[CZK15] CHOI S., ZHOU Q. Y., KOLTUN V.: Robust reconstruction of indoor scenes. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR,* (June 2015), pp. 5556–5565. 2

[DKD*16] DOU M., KHAMIS S., DEGTYAREV Y., DAVIDSON P. L., FANELLO S. R., KOWDLE A., ORTS-ESCOLANO S., RHEMANN C., KIM D., TAYLOR J., KOHLI P., TANKOVICH V., IZADI S.: Fusion4d: real-time performance capture of challenging scenes. *ACM Transactions on Graphics, TOG, 35*, 4 (2016), 114:1–114:13. 3, 6

[DNZ*17] DAI A., NIESSNER M., ZOLLHÖFER M., IZADI S., THEOBALT C.: Bundlefusion: real-time globally consistent 3d reconstruction using on-the-fly surface re-integration. *ACM Transactions on Graphics, TOG, 36*, 4 (2017). 2, 3

[EHE*12] ENDRES F., HESS J., ENGELHARD N., STURM J., CREMERS D., BURGARD W.: An evaluation of the RGB-D SLAM system. In *International Conference on Robotics and Automation, ICRA,* (2012), pp. 1691–1696. 2

[GXW*18] GUO K., XU F., WANG Y., LIU Y., DAI Q.: Robust non-rigid motion tracking and surface reconstruction using l_0 regularization. *IEEE Transactions on Visualization and Computer Graphics, TVCG, 24*, 5 (2018), 1770–1783. 3

[HKH*10] HENRY P., KRAININ M., HERBST E., REN X., FOX D.: RGB-D mapping: Using depth cameras for dense 3d modeling of indoor environments. In *International Symposium on Experimental Robotics, ISER* (2010), pp. 477–491. 2

[HZ06] HARLTEY A., ZISSERMAN A.: *Multiple view geometry in computer vision (2. ed.)*. Cambridge University Press, 2006. 2

[KBH06] KAZHDAN M. M., BOLITHO M., HOPPE H.: Poisson surface reconstruction. In *Eurographics Symposium on Geometry Processing, SGP,* (2006), pp. 61–70. 3

Figure 16: Our system can produce high-quality 3D reconstruction for various objects using Occipital structure sensor. It is robust for geometric-complex items like the toy dinosaur and various shoes. The photometric metric further improves the texture quality. Even the ingredient information on the formula canister is legible.

[KPM16] KÄHLER O., PRISACARIU V. A., MURRAY D. W.: Real-time large-scale dense 3d reconstruction with loop closure. In *European Conference on Computer Vision, ECCV,* (2016), pp. 500–516. 2

[KSC13a] KERL C., STURM J., CREMERS D.: Dense visual slam for rgb-d cameras. In *International Conference on Intelligent Robots and Systems, IROS,* (Nov 2013), pp. 2100–2106. 2

[KSC13b] KERL C., STURM J., CREMERS D.: Robust odometry estimation for rgb-d cameras. In *IEEE International Conference on Robotics and Automation, ICRA,* (May 2013), pp. 3748–3754. 2

[LI07] LEMPITSKY V., IVANOV D.: Seamless mosaicing of image-based texture maps. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR,* (June 2007), pp. 1–6. 3

[Low04] LOWE D. G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision, IJCV, 60*, 2 (2004), 91–110. 2

[LPC*00] LEVOY M., PULLI K., CURLESS B., RUSINKIEWICZ S., KOLLER D., PEREIRA L., GINZTON M., ANDERSON S., DAVIS J., GINSBERG J., SHADE J., FULK D.: The digital michelangelo project: 3d scanning of large statues. In *ACM SIG International Conference on Computer Graphics and Interactive Techniques, SIGGRAPH,* (2000), pp. 131–144. 3

[NFS15] NEWCOMBE R. A., FOX D., SEITZ S. M.: Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR,* (2015), pp. 343–352. 3, 7, 9

[NIH*11] NEWCOMBE R. A., IZADI S., HILLIGES O., MOLYNEAUX D., KIM D., DAVISON A. J., KOHI P., SHOTTON J., HODGES S., FITZGIBBON A.: Kinectfusion: Real-time dense surface mapping and tracking. In *International Symposium on Mixed and Augmented Reality, ISMAR,* (2011), pp. 127–136. 1, 2, 7, 8

[NZIS13] NIESSNER M., ZOLLHÖFER M., IZADI S., STAMMINGER M.: Real-time 3d reconstruction at scale using voxel hashing. *ACM Transactions on Graphics, TOG, 32*, 6 (2013), 169:1–169:11. 1, 2

[RBB09] RUSU R. B., BLODOW N., BEETZ M.: Fast point feature histograms (FPFH) for 3d registration. In *International Conference on Robotics and Automation, ICRA,* (2009), pp. 3212–3217. 2, 7

[RRKB11] RUBLEE E., RABAUD V., KONOLIGE K., BRADSKI G. R.: ORB: an efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision, ICCV,* (2011), pp. 2564–2571. 2, 7

[RV12] ROTH H., VONA M.: Moving volume kinectfusion. In *British Machine Vision Conference, BMVC,* (2012), pp. 1–11. 2

[RZS07] REITINGER B., ZACH C., SCHMALSTIEG D.: Augmented reality scouting for interactive 3d reconstruction. In *IEEE Virtual Reality, VR,* (2007), pp. 219–222. 1

[SA07] SORKINE O., ALEXA M.: As-rigid-as-possible surface modeling. In *Eurographics Symposium on Geometry Processing,* (2007), p-p. 109–116. 3

[SKM*17] SHENG Y., KANG C., MINGHUA L., HONGBO F., SHIÂĂŘMIN H.: Salienc-aware real-time volumetric fusion for object reconstruction. *Computer Graphics Forum 36*, 7 (2017), 167–174. 2

[SSP07] SUMNER R. W., SCHMID J., PAULY M.: Embedded deformation for shape manipulation. *ACM Transactions on Graphics, TOG, 26*, 3 (2007), 80. 2, 3, 5, 6

[Wik] WIKIPEDIA: Structured-light 3d scanner. https://en.wikipedia.org/wiki/Structured-light_3D_scanner. 2

[WKF*12] WHELAN T., KAESS M., FALLON M., JOHANNSSON H., LEONARD J., MCDONALD J.: Kintinuous: Spatially extended kinectfusion. p. 7. 2

[WLS*15] WHELAN T., LEUTENEGGER S., SALAS-MORENO R. F., GLOCKER B., DAVISON A. J.: Elasticfusion: Dense SLAM without A pose graph. In *Robotics: Science and Systems XI, Sapienza University of Rome,* (2015). 1, 2, 3, 7, 8

[WMG14] WAECHTER M., MOEHRLE N., GOESELE M.: Let there be color! large-scale texturing of 3d reconstructions. In *European Conference on Computer Vision, ECCV,* (2014), pp. 836–850. 3, 4

[WWLG09] WEISE T., WISMER T., LEIBE B., GOOL L. V.: In-hand scanning with online loop closure. In *International Conference on Computer Vision Workshops, ICCV Workshops* (2009), pp. 1630–1637. 2, 3

[WZB14] WANG K., ZHANG G., BAO H.: Robust 3d reconstruction with an RGB-D camera. *IEEE Transactions on Image Processing, TIP, 23*, 11 (2014), 4893–4906. 1

[YYFX18] YANG L., YAN Q., FU Y., XIAO C.: Surface reconstruction via fusing sparse-sequence of depth images. *IEEE Transactions on Visualization and Computer Graphics 24*, 2 (2018), 1190–1203. 2

[ZK14] ZHOU Q., KOLTUN V.: Color map optimization for 3d reconstruction with consumer depth cameras. *ACM Transactions on Graphics, TOG, 33*, 4 (2014), 155:1–155:10. 3

[ZK15] ZHOU Q., KOLTUN V.: Depth camera tracking with contour cues. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR,* (2015), pp. 632–638. 2

[ZMK13] ZHOU Q., MILLER S., KOLTUN V.: Elastic fragments for dense scene reconstruction. In *International Conference on Computer Vision, ICCV,* (2013), pp. 473–480. 2, 3, 7, 9

[ZNI*14] ZOLLHÖFER M., NIESSNER M., IZADI S., RHEMANN C., ZACH C., FISHER M., WU C., FITZGIBBON A. W., LOOP C. T., THEOBALT C., STAMMINGER M.: Real-time non-rigid reconstruction using an RGB-D camera. *ACM Transactions on Graphics, TOG, 33*, 4 (2014), 156:1–156:12. 3

[ZPK16] ZHOU Q., PARK J., KOLTUN V.: Fast global registration. In *European Conference on Computer Vision, ECCV,* (2016), pp. 766–782. 2

[ZZZL12] ZENG M., ZHAO F., ZHENG J., LIU X.: A memory-efficient kinectfusion using octree. In *Proceedings of the First International Conference on Computational Visual Media* (2012), International conference on Computational Visual Media, CVM, Springer-Verlag, pp. 234–241. 2