Motion Segmentation of Truncated Signed Distance Function based Volumetric Surfaces

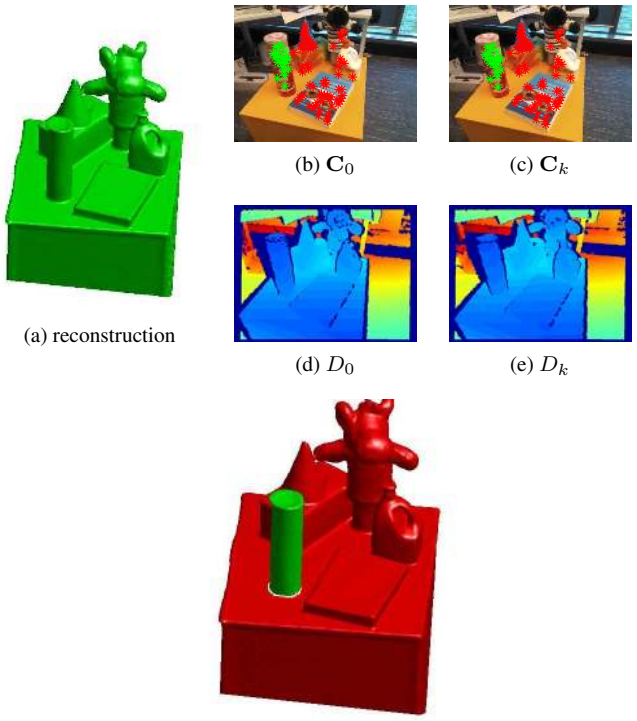
Samunda Perera, Nick Barnes, Xuming He
ANU, NICTA

Shahram Izadi, Pushmeet Kohli
Microsoft Research Cambridge

Ben Glocker
Imperial College London

Abstract

Truncated signed distance function (TSDF) based volumetric surface reconstructions of static environments can be readily acquired using recent RGB-D camera based mapping systems. If objects in the environment move then a previously obtained TSDF reconstruction is no longer current. Handling this problem requires segmenting moving objects from the reconstruction. To this end, we present a novel solution to the motion segmentation of TSDF volumes. The segmentation problem is cast as CRF-based MAP inference in the voxel space. We propose: a novel data term by solving sparse multi-body motion segmentation and computing likelihoods for each motion label in the RGB-D image space; and, a novel pairwise term based on gradients of the TSDF volume. Experimental evaluation shows that the proposed approach achieves successful segmentations on reconstructions acquired with KinectFusion. Unlike the existing solutions which only work if the objects move completely from their initially occupied spaces, the proposed method permits segmentation of objects when they start to move.



(a) reconstruction (b) $\mathbf{C}_0$ (c) $\mathbf{C}_k$ (d) D_0 (e) D_k (f) segmentation result

Figure 1: Example motion segmentation of the TSDF. A cylindrical object on a table undergoes translational motion. (a) The TSDF based reconstruction at time t_0. (b), (c) color images observed at time t_0 and t_k. (d), (e) corresponding depth images. (f) segmentation output using proposed method.

1. Introduction

The Truncated Signed Distance Function (TSDF) based volumetric surface representation format [4] represents a 3D environment as a voxel grid in which each voxel stores the signed distance to the nearest surface. The representation is widely used in recent RGB-D camera based environment mapping and localization systems like KinectFusion [12, 9], Patch Volumes [6] and CopyMe3D [3, 18]. This paper tackles the problem of segmenting such a volumetric surface reconstruction of a scene into distinctively moving objects.

More specifically, given a TSDF-based reconstruction of a static environment, consider the following scenario. The static environment is constituted of different objects which remain static during the reconstruction. Once the scene is reconstructed, one or more rigid objects in the scene start to move (Figure 1). Thus, the reconstruction no longer represents the current scene. Given the reference dense reconstruction of the previously static environment and images from a moving RGB-D camera observing this currently dynamic scene, this paper solves the problem of segmenting the reference reconstruction into object groups correspond-

978-1-4799-6683-7/15 $31.00 © 2015 IEEE
DOI 10.1109/WACV.2015.144

ing to the moving objects and the background.

Apart from the theoretical interest of segmenting TSDF volumes, solving the above problem has important practical applications. First of all, if moving objects can be segmented then they can be tracked over time and the reconstruction (both background and moving objects) can be continuously updated over time by transforming the objects and integrating new RGB-D data. For example, this would permit an area of the background that was earlier occluded by an object to be reconstructed. If moving objects are not handled and moving average based updates are used [12] then reconstructed objects have to be discarded and mapping artifcats can be observed. In addition, an automatic 3D object segmentation method would permit an autonomous robot to first reconstruct a scene using an RGB-D mapping system and then to observe and segment objects that start to move. The fact that the 3D segmentations carry rich structure of the objects could be useful in helping the robot better handle the situation involved. Moreover, motion segmentation of reconstructions would permit a user to actively move an object of interest to get a segmentation of it.

To this end we propose a novel method for motion segmentation of TSDF reconstructions. We cast the segmentation as a CRF-based MAP inference problem in the voxel space of the TSDF volume. By computing the motion of each moving object in the scene using sparse point track segmentation, we compute a unary likelihood/cost of assigning a particular motion label to a pixel in an observed RGB-D image. This gives rise to a novel data term involving the voxels. A novel pairwise term is constructed based directly on the gradients of the TSDF volume. Unlike existing approaches, this allows us to segment moving objects that move only slightly.

2. Related Work

Multi-body motion segmentation Existing work on multi-body motion segmentation has been largely on 2D/3D sparse point track segmentation [2] and on 2D/2.5D dense image segmentation [16, 17]. The work is mainly based on model selection [5], affinity clustering [13] and subspace clustering. Such work is useful for, but does not directly tackle dense segmentation of a 3D grid representation such as a TSDF.

TSDF segmentation In [9], Izadi et al. describe an occupancy-based approach to segment moving objects from a TSDF reconstruction. They first map a scene using KinectFusion. A human user then moves an object in the scene that needs to be segmented, e.g. a tea pot resting on a table (Figure 2). When the object is moved from its initial position to a new position, the initial position becomes unoccupied (free space). Izadi et al. use the camera pose obtained from KinectFusion along with the observed depth images to detect these newly unoccupied voxels representing the object. However, in order to correctly segment the object, the object has to be moved completely from its initially occupied space. If the object is only slightly or partially moved, the complete object cannot be extracted. The recent work by Herbst et al. [7] on change detection based segmentation of maps created using PatchVolumes has the same limitation. In contrast to these works, our method permits segmentation of objects when they start to move.

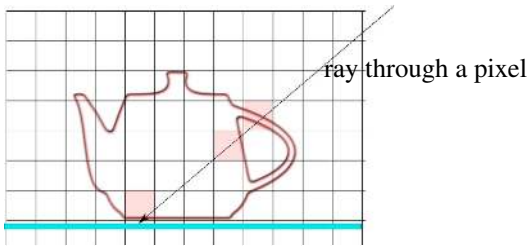


Figure 2: The object segmentation method of Izadi et al. [9]. The TSDF reconstruction represents a tea pot lying on a plane. When the tea pot is removed from the scene, the camera gives depth measurements corresponding to the plane. Based on this voxels corresponding to the tea pot can be identified.

The recent work by Ma and Sibley [11] solves motion based object discovery, object detection, tracking and reconstruction in an integrated system. Moving objects are discovered by parts of the scene that fail to track with the dominant object model and each object is reconstructed in its own TSDF volume in an online fashion. Jacquet et al. [10] considers the problem of reconstructing multiple moving objects in separate TSDF volumes allowing the volumes to interact with each other. While both [11] and [10] provide a segmentation of an environment as multiple TSDF volumes, they do not consider the segmentation of an existing reconstruction into multiple moving objects. For example, if an object was stationary during the reconstruction and later starts to move then unless the scene is reconstructed from scratch from all the pervious RGB-D images, [11] and [10] cannot segment the moving object from the reconstruction.

3. Input and Notation

The input to the motion segmentation problem consists of the TSDF reconstruction of the environment at time t_0, the RGB-D images observed at time t_0 and time t_k ($k > 0$) as well as the pose of the camera at time t_0. During time t_0 and t_k, objects in the scene undergo rigid motion. Figure 1 illustrates an example input to the problem.

Take any 3D Cartesian coordinate system attached to a static point of the environment and denote it as the global

coordinate system g. Let $\mathbf{p} \in \mathbb{R}^3$ be a point expressed in this global coordinate system. Then the TSDF value at the point $\mathbf{p}$ is denoted by $F(\mathbf{p})$. Let $\Gamma \subset \mathbb{R}^3$ be the continuous domain in which the TSDF reconstruction is defined. The TSDF reconstruction is discretized and made available as a set of voxels V. A fixed bijective mapping $\mathbf{v} : V \rightarrow \Gamma$ between the voxel elements and the points in the continuous domain is given.

The depth and color images are denoted by D_0, D_k and $\mathbf{C}_0, \mathbf{C}_k$ respectively. Note each image is indexed by the time instant it was captured. Each depth image D_k and corresponding color image $\mathbf{C}_k$ are assumed to be registered with each other. D_k and $\mathbf{C}_k$ are such that $D_k : \Omega \rightarrow \mathbb{R}$ and $\mathbf{C}_k : \Omega \rightarrow \mathbb{R}^3$ where $\Omega \subset \mathbb{R}^2$ is the image pixel domain. The set of pixels in an image is denoted by P. A fixed bijective mapping $\mathbf{u} : P \rightarrow \Omega$ between the pixel elements and the continuous image pixel domain exists. Note the 3D point cloud corresponding to the depth image D_k expressed in the camera coordinate frame at time t_k is given by $\mathbf{X}_k(\mathbf{u}) = D_k(\mathbf{u})\mathtt{K}^{-1}\begin{bmatrix}\mathbf{u} & 1\end{bmatrix}^T$ where $\mathtt{K}$ is the camera intrinsic matrix.

The pose of the camera at time t_0 is denoted by $\mathtt{T}_{g,0} \in \mathrm{SE}(3)$. Here, $\mathtt{T}_{g,0}$ is such that it transforms a point $\mathbf{x}_0 \in \mathbb{R}^3$ expressed in the camera coordinate frame at time t_0 to the global coordinate frame g according to the formula $\begin{bmatrix}\mathbf{x}_g & 1\end{bmatrix}^T = \mathtt{T}_{g,0}\begin{bmatrix}\mathbf{x}_0 & 1\end{bmatrix}^T$ where $\mathbf{x}_g$ is the coordinate of the point expressed in the global coordinate frame.

4. Problem Formulation

When objects move, the input TSDF reconstruction no longer represents the current scene. Thus, using current camera pose, correspondences between the newly observed RGB-D measurement and the TSDF reconstruction can only be established for the static part of the scene. Therefore, we compute the TSDF segmentation by computing the likelihood for underlying motions in the RGB-D frame observed before motion by using a minimum of two frames. The problem is formulated as a CRF-based MAP inference problem in the voxel grid space of the TSDF at the first frame.

The voxels in the TSDF volume represent free space, reconstructed surfaces or the interior of surfaces. Let $\Lambda \subset V$ be the set of voxels representing the reconstructed surfaces. Given a set of motion labels L, the task of segmenting the TSDF reconstruction involves assigning a label $f_p \in L$ to each valid voxel $p \in \Lambda$. Let $\mathbf{f}$ denote a particular labeling assignment. Further, let E be a set of unordered pairs $\{p, q\}$ of neighboring elements in Λ. Then using the pairwise CRF model, the MAP labeling $\mathbf{f}^*$ is expressed

$$\mathbf{f}^* = \underset{\mathbf{f} \in L^{|\Lambda|}}{\operatorname{argmin}} \sum_{p \in \Lambda} \mathcal{D}_p(f_p) + \sum_{\{p,q\} \in E} \mathcal{V}_{p,q}(f_p, f_q) \qquad (1)$$

where $\mathcal{D}_p(f_p) = d_p^{f_p}$ is the unary term (data term) and $\mathcal{V}_{p,q}$ is the pairwise term.

4.1. Generating Label Space

As noted, during time t_0 and t_k, some of the objects in the scene move. We determine the underlying motion groups by segmenting sparse 3D point correspondences between the RGB-D images at time t_0 and t_k. The sparse segmentation is computed by applying RANSAC (using 3-point samples) in a greedy manner [19] to extract the Euclidean transformation of each motion group. Here, we adopt the triangle sampling based method proposed in [15] to improve the odds of a 3-point sample being an all inlier sample. The segmentation is then refined using PEARL [5].

The output of sparse motion segmentation is the set of motion labels $L = \{1, \ldots, m\}$ present in the scene and a set $\{\mathtt{T}^l_{0,k} \in \mathrm{SE}(3)\}_{l \in L}$ of motion parameters of each group.

The notation $\mathtt{T}^l_{0,k}$ is as follows. Consider a point belonging to a motion group l. The point is observed by the camera at time t_0 and t_k giving 3D coordinate measurements $\mathbf{x}_0 \in \mathbb{R}^3$ and $\mathbf{x}_k \in \mathbb{R}^3$ respectively. Here, the coordinates are expressed with respect to the camera coordinate frame at the respective time instants. Then $\mathtt{T}^l_{0,k}$ is such that $\begin{bmatrix}\mathbf{x}_0 & 1\end{bmatrix}^T = \mathtt{T}^l_{0,k}\begin{bmatrix}\mathbf{x}_k & 1\end{bmatrix}^T$.

Using similar notation, the Euclidean transformation which brings $\mathbf{x}_k$ to the global coordinate frame is denoted by $\mathtt{T}^l_{g,k}$. It is straightforward to see that $\mathtt{T}^l_{g,k} = \mathtt{T}_{g,0}\mathtt{T}^l_{0,k}$ for all k. Note the relationship $\mathtt{T}^l_{g,0} = \mathtt{T}_{g,0}$ for all $l \in L$ as the reconstruction represents the environment at time t_0. Figure 3 summarizes these Euclidean transformations for a scene with two motion groups.

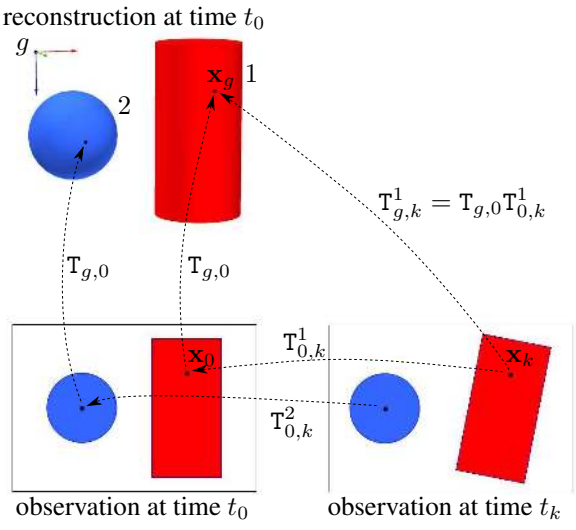
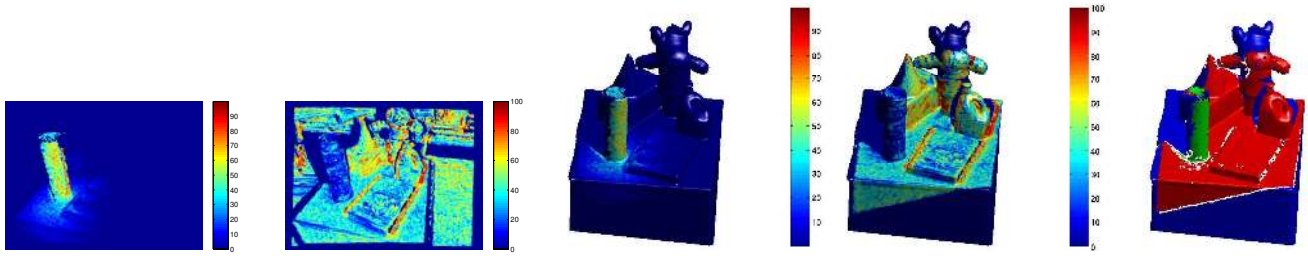


Figure 3: Euclidean transformation notation for a scene with two motion labels ($L = \{1, 2\}$).



(a) R_0^1 (b) R_0^2 (c) d_p^1 (d) d_p^2 (e) unary winner

Figure 4: Visualization of the data term. (a)-(b) image space unary costs. (c) cost for the motion label 1 (background motion). For voxels with no corresponding pixels, cost of $c = 0$ was used. (d) cost for the motion label 2 (motion of the cylinder) (e) per voxel winner (label corresponding to the minimum cost). Here, voxels having equal cost values in d_p^1 and d_p^2 are shown in blue color.

4.2. Data term

We construct the data term by computing unary costs for pixels in the RGB-D image space and associating them with corresponding voxels (see Figure 4). Let $R_k^l(\mathbf{u})$ denote the cost of assigning a particular motion label l to a pixel $\mathbf{u}$ in the RGB-D image at time t_k. Further, define the function which maps a pixel $\mathbf{u}$ in the RGB-D image at time t_k to one or more voxels in the TSDF reconstruction by

$$h(\mathbf{u}, D_k, \mathtt{T}_{g,k}^{L_k(\mathbf{u})}) : \Omega \times R \times \mathrm{SE}(3) \rightarrow \mathcal{P} \qquad (2)$$

where $L_k(\mathbf{u}) \in L$ is the label of pixel $\mathbf{u}$ and $\mathcal{P}$ is the power set of V, that is, the set of all possible subsets of V.

In order to obtain the voxels corresponding to a pixel in the RGB-D image at time t_k using the pixel to voxel mapping function h, the transformation $\mathtt{T}_{g,k}^{L_k(\mathbf{u})}$ should be selected from the set $\{\mathtt{T}_{g,k}^1, \ldots, \mathtt{T}_{g,k}^m\}$. However, as L_k are not known this cannot be performed for all but $k = 0$ [1]. At time t_0 all motion groups have the same transformation $\mathtt{T}_{g,0}$ (Figure 3). Thus pixels corresponding to the RGB-D image at time t_0 can be mapped to their corresponding voxels. Based on this the data term is given by

$$d_p^l = \begin{cases} R_0^l(\mathbf{u}) & p \in h(\mathbf{u}, D_0, \mathtt{T}_{g,0}) \\ c & p \notin \bigcup \{h(\mathbf{u}, D_0, \mathtt{T}_{g,0})\}_{\mathbf{u} \in P} \end{cases}. \qquad (3)$$

Here, c is any finite constant cost assigned for voxels with no corresponding pixels. Figure 4 illustrates this data term.

4.2.1 Image Space Unary Cost R_0^l

Using the Euclidean transformation $\mathtt{T}_{0,k}^l$, the point cloud corresponding to the RGB-D image at time t_k can be transformed to test the alignment with the point cloud corresponding to the RGB-D image at time t_0. The error of the alignment is computed using 3D position, color and surface normal information, and stands as our image space unary cost term R_0^l. Here, the point cloud $\mathbf{X}_k$ is first transformed into time t_0 by applying $\mathtt{T}_{0,k}^l$. The transformed point cloud is then used to obtain a synthesized depth image $\hat{D}_0^l$ and color image $\hat{\mathbf{C}}_0^l$ using Z-buffering. The corresponding 3D point cloud is given by $\hat{\mathbf{X}}_0^l$.

3D position alignment cost The 3D position alignment cost is based on the distance between the corresponding points.

$$r_{\mathrm{3D}}^l(\mathbf{u}) = \min(||\mathbf{X}_0(\mathbf{u}) - \hat{\mathbf{X}}_0^l(\mathbf{u})||, c_1)/c_1 \qquad (4)$$

Here, $c_1 > 0$ is an upper bound (truncation value) on the position alignment error.

Color compatibility cost The color compatibility of the alignment is computed based on normalized cross-correlation (NCC) between 3x3 image patches of $\mathbf{C}_0$ and $\hat{\mathbf{C}}_0^l$, centered on corresponding pixels. The sum γ of NCC over all 3 color channels gives the color compatibility. Note $-3 \leq \gamma \leq 3$. This is converted to a cost term by

$$r_{\mathrm{Color}}^l(\mathbf{u}) = \min(3 - \gamma(\mathbf{C}_0, \hat{\mathbf{C}}_0^l, \mathbf{u}), c_2)/c_2 \qquad (5)$$

where $0 < c_2 \leq 6$ is an upper bound.

Surface normal alignment cost The surface normal alignment cost is based on the angle difference between surface normals of the corresponding points. Let the unit surface normal maps for point clouds $\mathbf{X}_k$ and $\hat{\mathbf{X}}_k^l$ be $\mathbf{N}_k$ and $\hat{\mathbf{N}}_k^l$ respectively. Here, $\mathbf{N}_k : \Omega \rightarrow \mathbb{R}^3$. Then the normal alignment cost is given by

[1] If dense optical flow or per pixel tracking has been computed over all the image frames then the pixels in an image at time t_k can be associated with the pixels in the image at time t_0. In this case, image space unary costs can be mapped to the voxels using the camera pose $\mathtt{T}_{g,0}$ at time t_0. However, we do not compute such dense point correspondences.

$$r^l_{\text{Normal}}(\mathbf{u}) = \min(\arccos[\mathbf{N}_0(\mathbf{u}) \cdot \hat{\mathbf{N}}^l_0(\mathbf{u})], c_3)/c_3 \quad (6)$$

where $0 < c_3 < 180°$. Here, the surface normal maps are computed using Plane PCA.

Combined alignment cost The 3D position alignment cost, color compatibility cost and surface normal alignment cost are aggregated to produce a combined alignment error

$$r^l(\mathbf{u}) = \frac{s_1 r^l_{\text{3D}}(\mathbf{u}) + s_2 r^l_{\text{Color}}(\mathbf{u}) + s_3 r^l_{\text{Normal}}(\mathbf{u})}{s_1 + s_2 + s_3} \quad (7)$$

where s_1, s_2 and s_3 are mixing coefficients that determine the relative contribution of respective terms.

Background label unary cost attenuation Assuming moving objects are spatially concentrated, we increase the preference of pixels that are farther away from moving objects to have the background motion label B. Let $\rho(\mathbf{u}) : \Omega \rightarrow \mathbb{R}$ be a distance map indicating the 3D distance to the nearest object feature point, identified using the sparse segmentation result. Then the updated cost is given by

$$R^l_0(\mathbf{u}) = \begin{cases} \exp(-0.5\,\rho(\mathbf{u})^2/\sigma^2)\, r^l(\mathbf{u}) & l = B \\ r^l(\mathbf{u}) & \text{otherwise} \end{cases}. \quad (8)$$

where σ is a parameter that depends on the spatial size of the objects being considered.

4.2.2 Pixel to Voxel Mapping Function h

For a pixel $\mathbf{u}$ in the RGB-D image at time t_k, the 3D coordinates $\mathbf{x}_g(\mathbf{u}) \in \mathbb{R}^3$ of the corresponding surface point at time t_0 in the global coordinate frame can be computed using $\mathrm{T}_{g,k}^{L_k(\mathbf{u})}$. Here $\mathbf{x}_g$ is expected to lie on a zero crossing voxel. In order to gain more support for the segmentation, we return all voxels along the gradient vector at the voxel $\mathbf{x}_g$. This is achieved by marching the corresponding ray going through the voxel, from + to - TSDF values (forward) or vice versa. Algorithm 1 illustrates this approach.

Algorithm 1 Pixel to Voxel Mapping

input pixel $\mathbf{u}$, depth image D, Euclidean transformation T, truncation distance μ, TSDF volume F and its gradient ∇F

output set of voxels v

$\mathbf{p} \leftarrow$ point given by $D(\mathbf{u})$ expressed in global coordinates
$\mathbf{ray}^{\text{dir}} \leftarrow \nabla F(\mathbf{p})$
$\mathbf{ray}^{\text{dir}} \leftarrow$ normalize($\mathbf{ray}^{\text{dir}}$)
$\mathbf{p} \leftarrow \mathbf{p} - \mathbf{ray}^{\text{dir}}\mu$
if $\mathbf{p}$ outside bounding box of volume **then**
  $\mathbf{p} \leftarrow$ first voxel along $\mathbf{ray}^{\text{dir}}$
end if
$v = \emptyset$
hit = **false**, step = 0
step_size = smallest length of a voxel in all three dimensions
while step $\leq$ $(2\mu/\text{step\_size})$ and $\mathbf{p}$ is within the bounding box of volume **do**
  step = step + 1
  $\mathbf{p} \leftarrow \mathbf{p} + \mathbf{ray}^{\text{dir}}$ step_size
  if $|\mathrm{F}(\mathbf{p})| < \mu$ **then**
    hit = **true**
    $v = v \cup$ voxel at $\mathbf{p}$
  else if hit == **true** and $|\mathrm{F}(\mathbf{p})| == \mu$ **then**
    break
  end if
end while
return v

For a pixel $\mathbf{u}$ in the RGB-D image at time t_k, the 3D coordinates $\mathbf{x}_g(\mathbf{u}) \in \mathbb{R}^3$ of the corresponding surface point at time t_0 in the global coordinate frame can be computed using $\mathrm{T}_{g,k}^{L_k(\mathbf{u})}$. Here $\mathbf{x}_g$ is expected to lie on a zero crossing voxel. In order to gain more support for the segmentation, we return all voxels along the gradient vector at the voxel $\mathbf{x}_g$. This is achieved by marching the corresponding ray going through the voxel, from + to - TSDF values (forward) or vice versa. Algorithm 1 illustrates this approach.

4.3. Pairwise term

Unlike the data term which was computed using the observed RGB-D images, the pairwise term is computed solely based on the gradient of the TSDF reconstruction. The TSDF values $F(\mathbf{p})$ form a scalar field over the 3D volume defined by the voxel grid. The gradient of this scalar field is a 3D vector field denoted by

$$\nabla F(\mathbf{p}) = \begin{bmatrix} \frac{\partial F}{\partial x} & \frac{\partial F}{\partial y} & \frac{\partial F}{\partial z} \end{bmatrix}^T \quad (9)$$

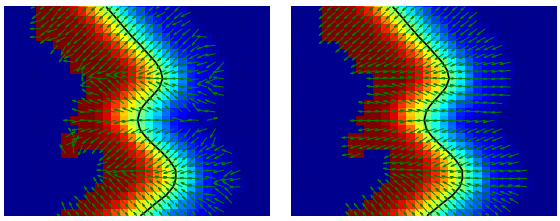
where $\begin{bmatrix} x & y & z \end{bmatrix}^T = \mathbf{p}$. Here, discrete approximations to the directional derivatives $\frac{\partial F}{\partial x}$, $\frac{\partial F}{\partial y}$, $\frac{\partial F}{\partial z}$ at each voxel can be computed using centered differences (for interior voxels) and forward/backward differences (for border voxels). The computed directional derivatives can be then scaled appropriately in each dimension to account for the physical spacing between voxels in each dimension.

The 3D gradients thus computed can be noisy, particularly near truncation borders. Therefore, the bilateral filter [14] is applied along each dimension of the 3D gradient volume. That is, given the 3D gradients $\nabla F(\mathbf{p}) = \begin{bmatrix} F_1(\mathbf{p}) & F_2(\mathbf{p}) & F_3(\mathbf{p}) \end{bmatrix}^T$ at each voxel position $\mathbf{p}$ in the voxel domain V, the filtered gradients $\tilde{\nabla F}(\mathbf{p}) = \begin{bmatrix} \tilde{F}_1(\mathbf{p}) & \tilde{F}_2(\mathbf{p}) & \tilde{F}_3(\mathbf{p}) \end{bmatrix}^T$ are computed as

$$\tilde{F}_i(\mathbf{p}) = \frac{1}{W_\mathbf{p}} \sum_{\mathbf{q} \in V} \exp\left(-\frac{||\mathbf{p}-\mathbf{q}||^2}{\sigma_s^2}\right) \exp\left(-\frac{|F_i(\mathbf{p}) - F_i(\mathbf{q})|^2}{\sigma_r^2}\right) F_i(\mathbf{q}) \quad (10)$$

for $i = 1, 2, 3$, where $W_\mathbf{p}$ is a normalizing constant. Here, σ_s and σ_r are the spatial and range standard deviations respectively. The filtered gradients are then normalized to unit

magnitude.



(a) (b)

Figure 5: Gradient of a 2D slice of the TSDF (a) The unit gradient vectors computed using finite differences represented as arrows (b) The bilateral filtered output. The black curve shows the implicit surface.

Figure 5 illustrates the gradients computed for a 2D slice of the TSDF volume along with the filtered result. As the gradient of a function is perpendicular to the level set of the function at the point, the gradient at the zero crossing (zero level) indicates the surface normal direction. The voxels that lie along the surface normal direction exhibit similar gradient directions. Based on the filtered gradients, the pairwise term of the segmentation cost function is thus constructed as

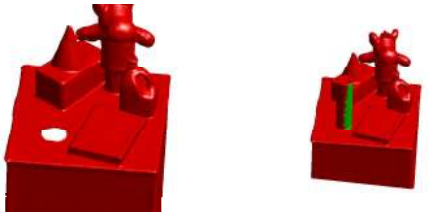
$$\mathcal{V}_{p,q}(f_p, f_q) = \lambda \, \mathrm{T}(f_p \neq f_q) \exp\left(\frac{-\theta_{p,q}^2}{2\sigma_\theta^2}\right) \qquad (11)$$

where $\theta_{p,q} = \arccos\left(|\nabla\tilde{F}(\mathbf{p}) \cdot \nabla\tilde{F}(\mathbf{q})|\right)$ and σ_θ is an input parameter. $T(\cdot)$ is such that $T(\textit{true}) = 1$ and $T(\textit{false}) = 0$. Note the pairwise term introduces a cost for neighboring voxels with similar gradient direction to have different labels. λ is a scalar controlling the effect of the pairwise term.

5. Experimental Results and Discussion

The proposed motion segmentation method was evaluated using several TSDF based reconstructions obtained using an implementation of the KinectFusion system. A 0.75 m x 0.75 m x 0.75 m volume of an environment was reconstructed using a voxel grid with a resolution of 256 x 256 x 256. The truncation parameters were set as $c_1 = 30$mm, $c_2 = 5$ and $c_3 = 45^\circ$ and the mixing coefficients were set as $s_1 = 2$, $s_2 = 4$ and $s_3 = 1$. The label corresponding to the background motion (required in background unary cost attenuation) was identified based on the largest depth range and the largest sparse feature point group. The parameter σ was computed using the maximum volume of the 3D convex hulls of the sparse feature points corresponding to each object motion group. If the maximum volume is v then σ was set to $\sigma = 3v^{1/3}$. The data term was multiplied by 1000 and the pairwise term scale λ was set to 1e6. The pairwise term was computed using the 8-connectivity of voxels and σ_θ was set to 5°. α-Expansion [1] was used to solve the segmentation cost function.

Figure 6 shows the segmentation results for the example sequence obtained. The cylinder was correctly segmented from the planar table top. The figure also shows the segmentation result obtained using the existing method by Izadi et al [9]. As noted their method requires completely moving the cylinder from its previously occupied position. However, as the cylinder was only slightly moved (Figure 1), their method was unable to extract the complete object.



(a) proposed (b) existing

Figure 6: Segmentation results for the cylinder sequence. (a) the segmented cylinder removed to show the segmentation boundary. (b) result using the existing method [9].

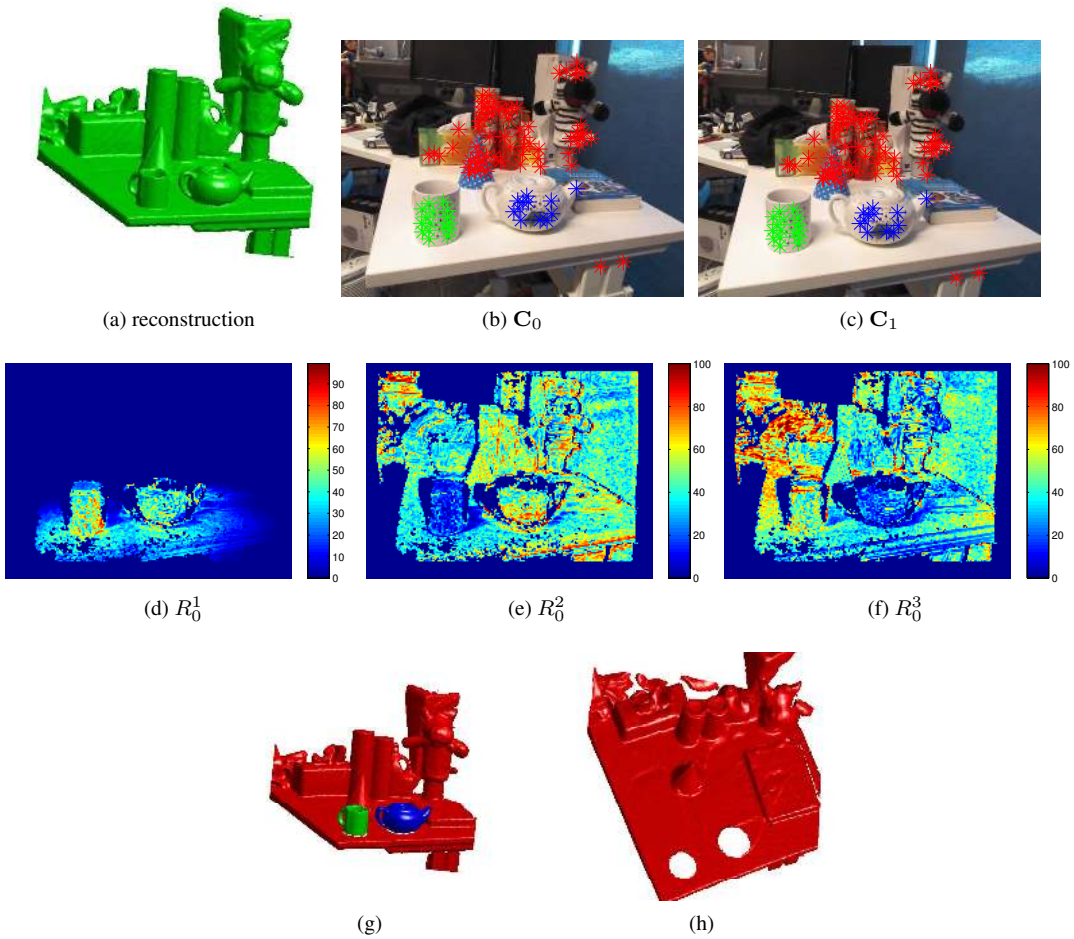
In another test data sequence, a tea pot and a mug was slightly moved on a table. Figure 7 shows the results for this **TeaPotMug** sequence. The segmentation of sparse feature points identified three motion groups corresponding to the two moving objects and the background. The unary costs obtained for each motion label were then used to segment the TSDF volume. The tea pot and mug were successfully segmented from the table.

The **CarJacket** sequence features a toy car moving on top of a jacket lying on the floor (Figure 8). Here, the goal of this experiment was to show that the proposed TSDF based surface segmentation methodology is not limited to segmenting objects lying on a plane. As expected, the proposed method could successfully segment the car from rest of the scene. Supplementary material contains more results.

We list numerical results on the segmentation accuracy of the proposed method in Table 1. Here the accuracy was computed by projecting the segmented volumes to manually labeled depth images and counting the correctly labeled pixels (expressed as a percentage of the total number of pixels). As noted, our method outperforms the existing method in all three sequences. The table also gives the running time results of the α-expasion stage (Intel Core i7 1.6GHz CPU).

6. Conclusion and Future Work

In this paper, we considered the problem of motion segmentation of TSDF reconstructions. We proposed an au-



(a) reconstruction (b) $\mathbf{C}_0$ (c) $\mathbf{C}_1$

(d) R_0^1 (e) R_0^2 (f) R_0^3

(g) (h)

Figure 7: **TeaPotMug** sequence and segmentation results. (a)-(c) input. (d)-(f) image space unary costs. (g), (h) segmentation result using the proposed method.

tomatic method capable of successfully segmenting TSDF volumes by constructing appropriate data and pairwise terms. Existing TSDF based mapping systems either cannot handle the motion of objects which are already reconstructed, or cannot segment moving objects as and when they move. The TSDF segmentation method proposed in this paper addresses these limitations. In future, we plan to fully integrate the method with KinectFusion to update the reconstruction with new data. Further, we plan to use cost volume filtering [8] for the inference stage to achieve a faster running time.

Table 1: Quantitative results. segmentation accuracy of Izadi et al. [9] and ours. $|L|$ = #labels, $|\Lambda|$ = #valid voxels, $|E|$ = #edges, time = running time of α-expasion stage.

Sequence	$\|L\|$	$\|\Lambda\|$	$\|E\|$	accuracy: [9]	accuracy: ours	time
Cylinder	2	1.3M	3.6M	94.94%	99.95%	2.37 s
TeaPotMug	3	1.3M	3.7M	89.40%	99.79%	10.28 s
CarJacket	2	1.3M	3.6M	97.67%	99.91%	2.35 s

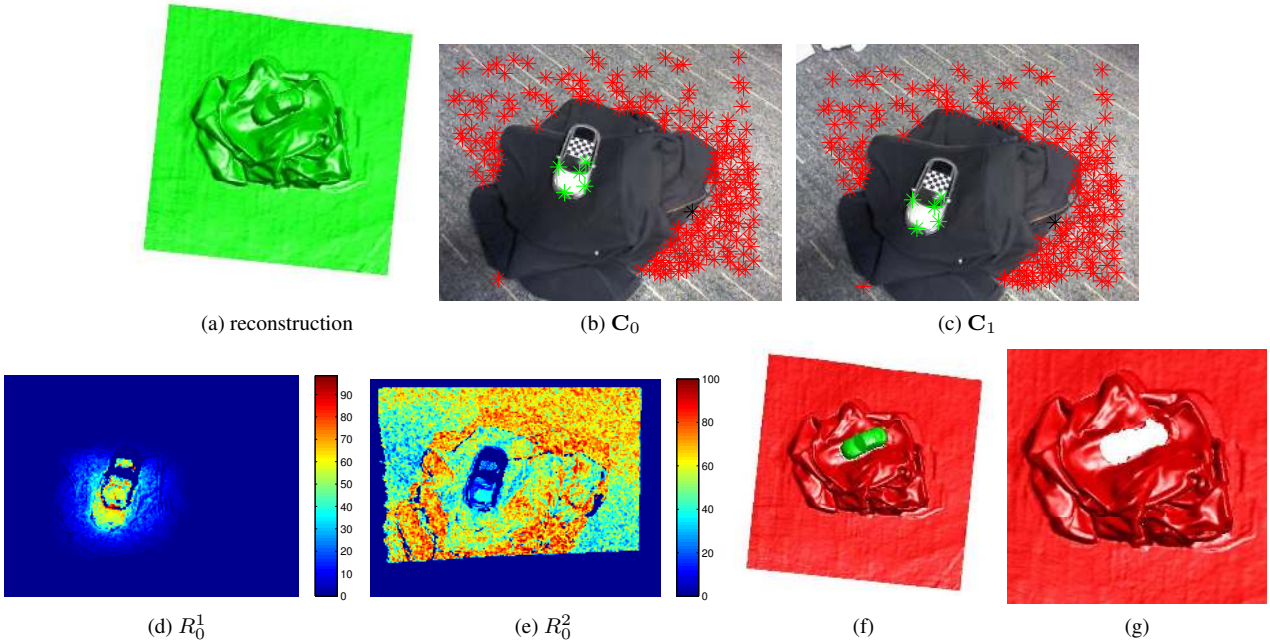
References

[1] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Trans.*, 23(11):1222–1239, 2001.

[2] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *ECCV 2010*, pages 282–295. Springer Berlin Heidelberg, 2010.

[3] E. Bylow, J. r. Sturm, C. Kerl, F. Kahl, and D. Cremers. Real-time camera tracking and 3D reconstruction using signed dis-

Acknowledgments: NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications, and the Digital Economy, and the Australian Research Council (ARC) through the ICT Centre of Excellence Program. This research was also supported in part by ARC through its Special Research Initiative (SRI) in Bionic Vision Science and Technology grant to Bionic Vision Australia (BVA). Thanks to NICTA students for helping with data capture.



(a) reconstruction (b) $\mathbf{C}_0$ (c) $\mathbf{C}_1$

(d) R_0^1 (e) R_0^2 (f) (g)

Figure 8: Results for the **CarJacket** sequence. (a)-(c) input. (d), (e) image space unary costs. (f), (g) result using the proposed method.

tance functions. In *Robotics: Science and Systems (RSS), Online Proceedings*, volume 9, page 8, 2013.

[4] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 303–312, New York, NY, USA, 1996. ACM.

[5] A. Delong, A. Osokin, H. Isack, and Y. Boykov. Fast approximate energy minimization with label costs. *International Journal of Computer Vision*, 96(1):1–27, 2012.

[6] P. Henry, D. Fox, A. Bhowmik, and R. Mongia. Patch volumes: Segmentation-based consistent mapping with RGB-D cameras. In *3D Vision, 2013 International Conference on*, pages 398–405, June 2013.

[7] E. Herbst, P. Henry, and D. Fox. Toward online 3-D object segmentation and mapping. In *IEEE ICRA*, 2014.

[8] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. *Pattern Analysis and Machine Intelligence, IEEE Trans.*, 35(2):504–511, 2013.

[9] 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 the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM.

[10] B. Jacquet, C. Hne, R. Angst, and M. Pollefeys. Multi-body depth-map fusion with non-intersection constraints. In *ECCV 2014*, pages 735–750. Springer International Publishing, 2014.

[11] L. Ma and G. Sibley. Unsupervised dense object discovery, detection, tracking and reconstruction. In *ECCV 2014*, pages 80–95. Springer International Publishing, 2014.

[12] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *ISMAR 2011*, pages 127–136, Oct 2011.

[13] P. Ochs and T. Brox. Higher order motion models and spectral clustering. In *CVPR 2012*, pages 614–621, June 2012.

[14] S. Paris and F. Durand. A fast approximation of the bilateral filter using a signal processing approach. In *ECCV 2006*, pages 568–580. Springer Berlin Heidelberg, 2006.

[15] S. Perera and N. Barnes. Maximal cliques based rigid body motion segmentation with a RGB-D camera. In *ACCV 2012*, pages 120–133. Springer Berlin Heidelberg, 2013.

[16] A. Roussos, C. Russell, R. Garg, and L. Agapito. Dense multibody motion estimation and reconstruction from a handheld camera. In *ISMAR 2012*, pages 31–40, Nov 2012.

[17] J. Stückler and S. Behnke. Efficient dense 3D rigid-body motion segmentation in RGB-D video. In *BMVC*. BMVA Press, 2013.

[18] J. Sturm, E. Bylow, F. Kahl, and D. Cremers. CopyMe3D: Scanning and printing persons in 3D. In *Pattern Recognition*, volume 8142, pages 405–414. Springer Berlin Heidelberg, 2013.

[19] P. H. S. Torr. Geometric motion segmentation and model selection. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 356(1740):1321–1340, 1998.