Depth Camera Tracking with Contour Cues

Qian-Yi Zhou Vladlen Koltun
Intel Labs

Abstract

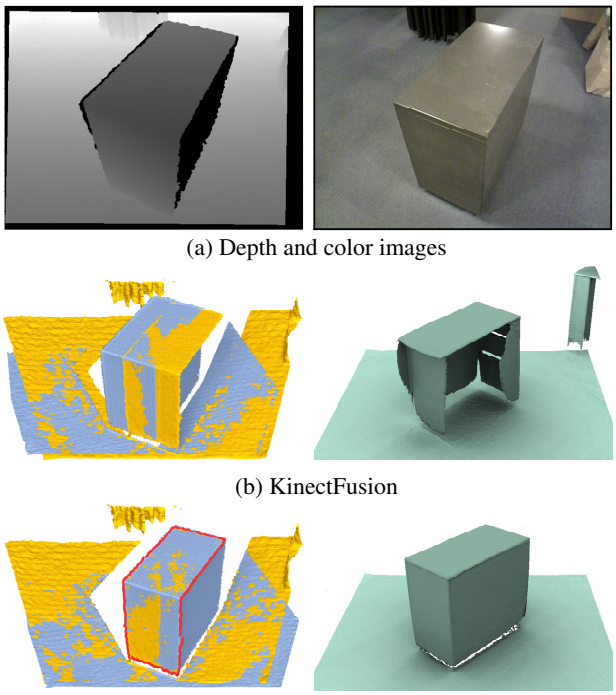
We present an approach for tracking camera pose in real time given a stream of depth images. Existing algorithms are prone to drift in the presence of smooth surfaces that destabilize geometric alignment. We show that useful contour cues can be extracted from noisy and incomplete depth input. These cues are used to establish correspondence constraints that carry information about scene geometry and constrain pose estimation. Despite ambiguities in the input, the presented contour constraints reliably improve tracking accuracy. Results on benchmark sequences and on additional challenging examples demonstrate the utility of contour cues for real-time camera pose estimation.

1. Introduction

Tracking self-motion is a primary function of visual perception in animals [7, 25]. In computer vision, the corresponding problem of visual odometry underlies a host of applications and has been extensively studied [5, 18, 11]. Our work concerns depth cameras, which are increasingly utilized in computer vision systems. Our goal is to improve the accuracy of depth camera tracking, particularly in challenging scenarios that currently lead to odometry drift.

The influential KinectFusion system [16] demonstrated real-time depth camera tracking and dense scene reconstruction by registering incoming depth images to a volumetric representation of the scene. Our work extends these ideas by integrating occluding contours into the optimization objective and showing that explicit handling of contours can lead to significant gains in tracking accuracy. A different extension of the KinectFusion approach was developed by Bylow et al. [2], who derived a principled optimization algorithm but did not track occluding contours. Our experiments demonstrate that contour tracking has significant benefits.

A number of odometry systems use both depth and color images [1, 9, 26]. Our work aims to maximize tracking accuracy without relying on a color image stream. One reason is that some depth cameras are not accompanied by color cameras. Another is that even if a color camera is present, its viewpoint is different and its shutter may not be perfectly synchronized with the depth camera. Finally, we aim for systems that function even in minimal lighting.



(a) Depth and color images

(b) KinectFusion

(c) Our approach

Figure 1. (a) Depth and color images from an input sequence. (b) Surface registration slips on planar surfaces, leading to tracking drift and reconstruction failure. (c) Our approach establishes contour constraints that stabilize real-time camera tracking (red). The color image (a, right) is shown for clarity and is not used by either approach.

Our approach is based on tracking occluding contours and using contour cues to constrain registration. This addresses a common failure mode of geometric registration approaches based on the iterative closest point (ICP) algorithm and its variants [19], namely instability in the presence of smooth surfaces [6]. This is illustrated in Figure 1, which shows a cabinet being imaged by a depth camera. In some aspects, the cabinet is seen as a collection of large planar surfaces that cause geometric alignment to slip and cam-

era tracking to drift. This behavior can be easily observed in practice and is also apparent in benchmark odometry sequences [22]. Our solution integrates contour constraints into the registration objective, stabilizing camera tracking in challenging scenarios.

Occluding contours were considered a primary source of information in the early days of computer vision [14, 12, 3, 8]. They are now used in state-of-the-art multi-view stereo systems to inform shape reconstruction given calibrated camera parameters [23, 21]. Our work leverages contour cues in a real-time tracking system that operates on high frame-rate depth image streams.

Merrell et al. [15] used visibility constraints in real-time surface reconstruction, but the camera was assumed to be localized by other means. Wang et al. [24] use contour cues for alignment of wide-baseline range scans, but their formulation was not designed for real-time tracking. Our approach jointly optimizes for projective correspondences and robust contour constraints in a high-performance real-time framework.

Experimental results on challenging input sequences demonstrate that our formulation significantly improves depth camera tracking accuracy.

2. Method

Occluding contours provide powerful geometric constraints due to the tangency property: for all points along the contour generator, the normal is orthogonal to the view ray [3]. This is the foundation of our approach. We recover normals along contour generators from depth image gradients. Using the recovered normal information, we incorporate contour constraints into a surface registration framework. A joint optimization objective integrates surface correspondence terms and contour constraints. This stabilizes the registration and significantly reduces drift in challenging scenarios.

We build on the KinectFusion framework [16]. In particular, we maintain a truncated signed distance function $\mathbb{F}$ as a volumetric representation of the scene [4]. Each depth image D_i is registered to $\mathbb{F}$ to estimate a camera pose $\mathbf{T}_i$, then integrated to keep $\mathbb{F}$ up to date. Prior to registration, D_i is smoothed by a bilateral filter and back-projected into the sensor's coordinate system as a set of points $\mathbf{V}_i$. To create 3D points and normals from $\mathbb{F}$, per-pixel ray casting is used to synthesize a proxy depth image $\widehat{D}_{i-1}$ at pose $\mathbf{T}_{i-1}$. It is back-projected as a set of points $\widehat{\mathbf{V}}_{i-1}$ with normals $\widehat{\mathbf{N}}_{i-1}$ and transformed into the global coordinate system as $\widehat{\mathbf{V}}^g_{i-1}$ and $\widehat{\mathbf{N}}^g_{i-1}$. The objective of registration is to find a transformation $\mathbf{T}_i$ that aligns $\mathbf{T}_i\mathbf{V}_i$ and $\widehat{\mathbf{V}}^g_{i-1}$.

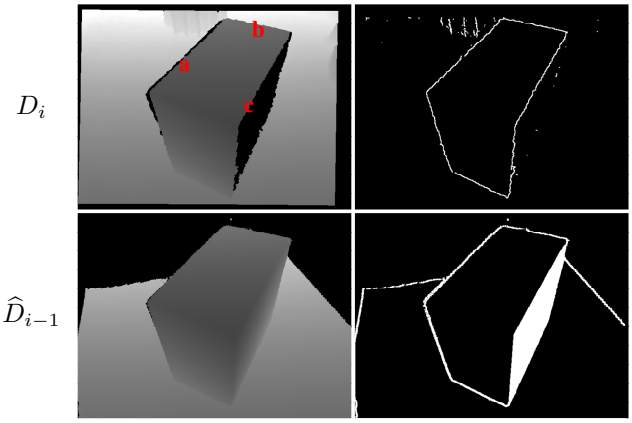


Figure 2. Top: raw depth image D_i (left) and detected occluding contours (right). Missing data may lead to inaccurate contours. Bottom: depth image $\widehat{D}_{i-1}$ synthesized from the constructed volumetric representation (left); robust normal estimation and resulting contour correspondence candidates (right). Despite ambiguities in the input, detected contours in D_i (top right) have appropriate correspondences in the candidate set (bottom right).

2.1. Contour detection

We begin by detecting occluding contours in depth image D_i. To prepare the image, we inpaint regions with missing depth information by scanning along horizontal lines and filling in missing intervals with farther adjacent depth values. This fills in occlusion regions that are caused by the separation between the infrared projector and the camera in structured light depth sensors [13]. In the inpainted depth image D'_i, pixels at depth discontinuities are considered contour generators. The set of contour generators is denoted by $\mathbf{C}_i$:

$$\mathbf{C}_i = \left\{ \mathbf{s} \in D_i \;:\; \exists \mathbf{t} \in \mathcal{N}^8_{\mathbf{s}},\ \text{s.t. } D'_i(\mathbf{s}) - D'_i(\mathbf{t}) > \delta \right\}, \quad (1)$$

where $\mathcal{N}^8_{\mathbf{s}}$ is the 8-neighborhood of $\mathbf{s}$ in D'_i and δ is a depth discontinuity threshold, set to 0.05 meters based on typical sensor noise magnitudes [10].

Note that surfaces viewed at grazing angles can disappear entirely from the depth image due to distortion of the projected pattern and the Fresnel effect (Figure 2, top left, point **c**). Such cases are handled transparently by our approach. The missing regions are filled in and inner boundaries are automatically flagged as contour generators, as shown in Figure 2. The detected contours may not align with the true contours, but inconsistencies are accommodated by subsequent processing stages.

2.2. Contour correspondence

The next processing step is to establish correspondences between detected contour generators $\mathbf{C}_i$ in D_i and points in the synthesized depth image $\widehat{D}_{i-1}$, which represents the

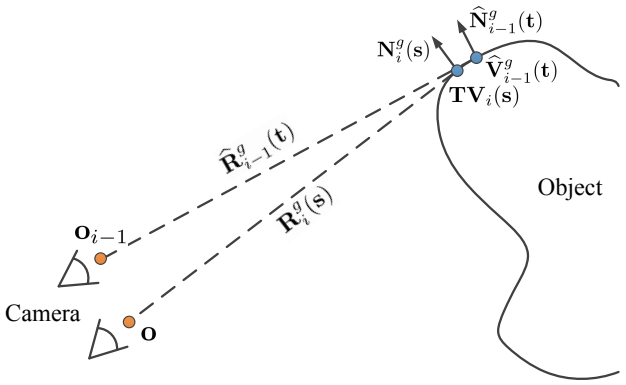
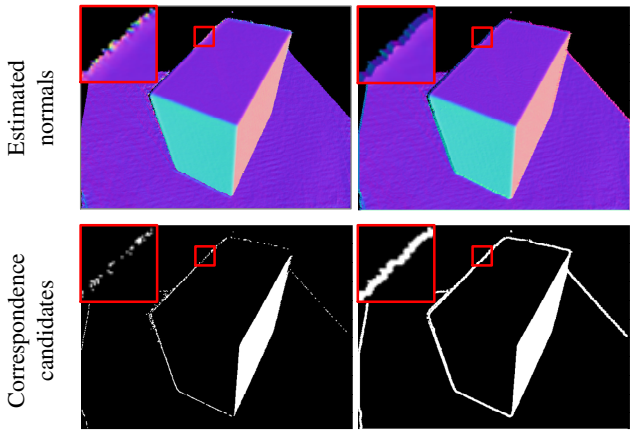


Figure 3. Notation for contour correspondence criteria.



(a) SDF derivative (b) Our approach

Figure 4. Normal estimation. (a) Numerical differentiation of the signed distance function is unstable along edges. (b) We use an alternative derivation to estimate the normal from the projected depth image.

scene model. These correspondences are computed at each step of the ICP procedure and are used to formulate contour constraint terms. These contour terms augment the registration objective and constrain the recomputed pose.

Let $\mathbf{T}$ be the pose of D_i at the current ICP iteration. For each point $\mathbf{s} \in \mathbf{C}_i$, we search for a point $\mathbf{t}$ in $\widehat{D}_{i-1}$ such that the distance between $\mathbf{s}$ and $\mathbf{t}$ in the global coordinate frame is small and their normal vectors are aligned. The distance criterion can be expressed as

$$\|\mathbf{TV}_i(\mathbf{s}) - \widehat{\mathbf{V}}^g_{i-1}(\mathbf{t})\| < \varepsilon. \tag{2}$$

We set $\varepsilon = 0.1$ meters. Enforcing normal alignment is harder. A reliable estimate of the normal $\mathbf{N}_i(\mathbf{s})$ is difficult to obtain from D_i due to missing data and lateral noise along edges [17]. Small perturbations of local silhouette geometry can yield drastic perturbations of the normal. Since we lack precise information about silhouette geometry, we do not have a reliable estimate of the normal. Our solution is to avoid computing the silhouette normal $\mathbf{N}_i(\mathbf{s})$ altogether, while still enforcing the normal alignment criterion. We use the fact that $\mathbf{N}_i(\mathbf{s})$ must be (nearly) perpendicular to the view ray [3]. Let $\mathbf{R}^g_i(\mathbf{s})$ denote the view ray:

$$\mathbf{R}^g_i(\mathbf{s}) = \frac{\mathbf{TV}_i(\mathbf{s}) - \mathbf{o}}{\|\mathbf{TV}_i(\mathbf{s}) - \mathbf{o}\|}, \tag{3}$$

where $\mathbf{o}$ is the camera origin in the global coordinate frame. Let $\mathbf{N}^g_i(\mathbf{s}) = \mathbf{TN}_i(\mathbf{s})$ be the normal of $\mathbf{s}$ in the global frame. We have

$$\mathbf{R}^g_i(\mathbf{s})^\top \mathbf{N}^g_i(\mathbf{s}) \approx 0. \tag{4}$$

As illustrated in Figure 3, we can assume under reasonable assumptions that the angle between $\mathbf{R}^g_i(\mathbf{s})$ and the view ray at $\mathbf{t}$, denoted as $\widehat{\mathbf{R}}^g_{i-1}(\mathbf{t})$, is small. Thus (4) is approximately equivalent to

$$\widehat{\mathbf{R}}^g_{i-1}(\mathbf{t})^\top \mathbf{N}^g_i(\mathbf{s}) \approx 0 \tag{5}$$

and the normal alignment criterion can be approximated as

$$\widehat{\mathbf{R}}^g_{i-1}(\mathbf{t})^\top \widehat{\mathbf{N}}^g_{i-1}(\mathbf{t}) < \zeta. \tag{6}$$

We use $\zeta = \cos(75°)$ and treat (6) as a necessary condition for establishing a contour correspondence with $\mathbf{t}$. Intuitively, the reduced criterion maintains that contours remain near-tangent to view rays under small camera motion. The reduced criterion is not as strict as genuine normal alignment at $\mathbf{s}$ and $\mathbf{t}$, but it can be enforced despite sensor noise and is effective in practice.

A significant advantage of criterion (6) is that it is invariant to the pose $\mathbf{T}$ and can thus can be precomputed before the iterative registration procedure. We thus compute a set of contour correspondence candidates $\widehat{\mathbf{C}}^g_{i-1}$ (Figure 2 bottom) and construct a kd-tree structure over this set in the global frame. In each ICP iteration, given $\mathbf{T}$ and $\mathbf{s} \in \mathbf{C}_i$ we look up the nearest point to $\mathbf{TV}_i(\mathbf{s})$ in $\widehat{\mathbf{C}}^g_{i-1}$ and validate equation (2).

2.3. Normal estimation

To identify contour correspondence candidates $\widehat{\mathbf{C}}^g_{i-1}$, we need to estimate the normal $\widehat{\mathbf{N}}^g_{i-1}(\mathbf{t})$ for points $\mathbf{t}$ in $\widehat{D}_{i-1}$. Surface normals can be estimated from the numerical gradient of the signed distance function $\mathbb{F}$ [16]. This approach works well for smooth surfaces but is unstable along edges, as shown in Figure 4. We develop an alternative normal estimation procedure designed to support robust contour correspondence estimation.

We estimate normals from the synthesized depth image $\widehat{D}_{i-1}$. This is not completely straightforward because $\widehat{D}_{i-1}$ was produced by a projective transformation and we seek a

fast image-based procedure that will yield surface normals in the global coordinate frame.

We begin by filling in missing depth values using the inpainting procedure described in Section 2.1. Let $h(u, v)$ denote the depth function defined by the inpainted depth image $\widehat{D}'_{i-1}$ over the image domain. This function can be alternatively represented as the zero level set of an implicit function $F(x, y, z)$ defined in the coordinate frame of the camera:

$$F(x, y, z) = h(u, v) - z, \tag{7}$$

where

$$x = \frac{1}{f_x}(u - c_x)h(u, v), \tag{8}$$

$$y = \frac{1}{f_y}(v - c_y)h(u, v). \tag{9}$$

(c_x, c_y) is the optical center and (f_x, f_y) are the focal lengths.

The normal at $\mathbf{s} = (u, v)$ is defined as the gradient of $F(x, y, z)$, i.e.

$$\widehat{\mathbf{N}}_{i-1}(\mathbf{s})^\top = \nabla F(x, y, z) = \left(\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}, \frac{\partial F}{\partial z}\right). \tag{10}$$

Let $\mathbf{g}(u, v, z) = (x, y, z)$ be a function in $\mathbb{R}^3 \to \mathbb{R}^3$. The gradient of $(F \circ \mathbf{g})(u, v, z)$ can be derived from equation (7):

$$\nabla(F \circ \mathbf{g})(u, v, z) = \left(\frac{\partial h}{\partial u}, \frac{\partial h}{\partial v}, -1\right), \tag{11}$$

where $(\frac{\partial h}{\partial u}, \frac{\partial h}{\partial v})$ is the image gradient computed by a 7×7 Sobel filter. By the chain rule,

$$\nabla(F \circ \mathbf{g})(u, v, z) = \nabla F(x, y, z)\mathbf{J}_{\mathbf{g}}(u, v, z). \tag{12}$$

The Jacobian matrix $\mathbf{J}_{\mathbf{g}}(u, v, z)$ can be computed from equations (8) and (9). We solve the linear system to estimate the normal $\widehat{\mathbf{N}}_{i-1}(\mathbf{s})$. This procedure is performed in real time on graphics hardware and yields reliable normal estimates throughout the image. Crucially, this includes normals along occluding contours, as illustrated in Figure 4(b).

2.4. Optimization

Let $\mathcal{K} = \{(\mathbf{s}, \mathbf{t})\}$ be the combined set of correspondence pairs obtained by projective data association [16] and by the contour correspondence procedure described in Section 2.2. Our optimization objective integrates contour alignment and surface alignment:

$$E(\mathbf{T}) = \sum_{(\mathbf{s},\mathbf{t})\in\mathcal{K}} w_{\mathbf{s},\mathbf{t}} \left(\left(\mathbf{T}\mathbf{V}_i(\mathbf{s}) - \widehat{\mathbf{V}}^g_{i-1}(\mathbf{t})\right)^\top \widehat{\mathbf{N}}^g_{i-1}(\mathbf{t})\right)^2,$$

where $w_{\mathbf{s},\mathbf{t}}$ is a weight that determines the relative influence of surface correspondences and contour correspondences. The objective is optimized iteratively [16]. We set $w_{\mathbf{s},\mathbf{t}}$ to 1 for surface correspondences, and to w_0 for contour correspondences. Setting $w_0 = 0$ reduces objective $E(\mathbf{T})$ to standard point-to-plane ICP and makes our system equivalent to KinectFusion. Increasing w_0 increases the strength of contour constraints. Our approach is not sensitive to small changes in w_0. Any value between 1 and 16 satisfactorily enforces contour constraints and stabilizes tracking. We set $w_0 = 4$ in all experiments.

3. Results

We evaluate the presented approach on sequences from the TUM RGB-D benchmark [22]. We focus on four sequences that demonstrate 3D scanning of objects. Our primary comparison is to two pure depth camera tracking approaches that do not rely on additional information channels: KinectFusion [16] (PCL implementation [20]) and the algorithm of Bylow et al. [2] (implementation provided by the authors). Each method produces a camera trajectory, which is compared against ground truth using the RMSE metric suggested by Sturm et al. [22]. The results are provided in Table 1. The presented approach yields the most accurate camera trajectories. Figure 5 shows the scene models produced by different algorithms on two of the benchmark sequences. The models produced by our approach are significantly better than the output of prior techniques.

For reference, we also report the performance of two techniques that use the color image stream in addition to the depth stream: the RGB-D odometry approach of Kerl et al. [9] (authors' implementation) and the algorithm of Whelan et al. [26] (our implementation). The results are given in Table 1. Our approach is more accurate than the reference techniques, without using color information.

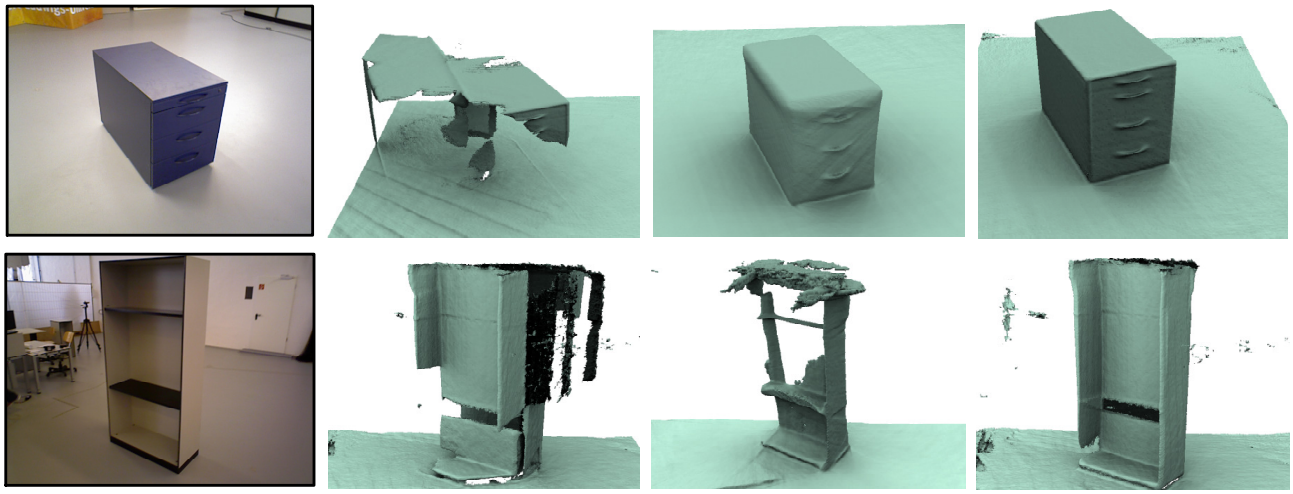
Figure 6 shows a variety of commonly encountered objects scanned with an Asus Xtion Live sensor and reconstructed by the presented approach. For all of these sequences, the surface alignment used by KinectFusion slipped and led to catastrophic loss of track at the frames shown in the figure. In contrast, the enforcement of contour cues by our approach prevented drift and consistently led to stable tracking of camera motion and reconstruction of scene geometry.

4. Discussion

We presented a depth camera tracking approach that uses contour cues for stabilizing tracking and reconstruction. This fixes catastrophic drift that often occurs in the presence of smooth surfaces. Although normals at boundaries may not be well-defined, we show how to robustly and efficiently establish and utilize generalized contour constraints.

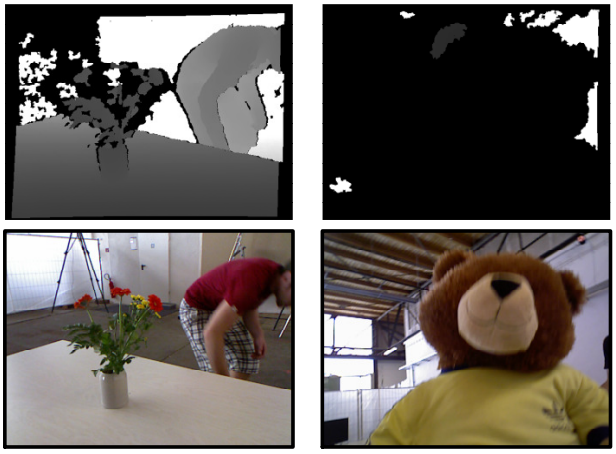
	KinectFusion [16]	Bylow et al. [2]	Our approach	Kerl et al. [9]	Whelan et al. [26]
fr3/cabinet	0.624	0.020	**0.015**	0.323	0.021
fr3/large_cabinet	0.275	0.109	**0.051**	0.103	0.055
fr3/structure_notexture_far	0.124	0.037	**0.026**	0.047	0.029
fr3/structure_notexture_near	0.252	**0.016**	0.023	0.384	0.023
Average	0.319	0.046	**0.029**	0.214	0.032

Table 1. Accuracy of estimated camera trajectories for TUM RGB-D benchmark sequences. (RMSE in meters.)



Reference color image | KinectFusion [16] | Bylow et al. [2] | Our approach

Figure 5. Qualitative comparison of scene models reconstructed by different algorithms on two TUM benchmark sequences. The color images are only shown for reference and were not used by any of the techniques.



Movement in the scene | Missing data

Figure 7. Failure cases from the TUM benchmark. Our approach and other approaches failed at the frames shown in the figure.

Our camera tracking approach only uses depth images, runs in real time, and significantly improves tracking accuracy.

There are limitations and opportunities for future work. When the depth image is missing a lot of data, for example due to highly specular or translucent surfaces, the method can fail. Without visible boundaries, for example when scanning a large featureless wall, tracking can still drift. Furthermore, there are degenerate scenarios in which even boundary cues cannot stabilize tracking: for example, rotating the camera above a round tabletop. Finally, like prior approaches, our work assumes that the scene is static. Figure 7 shows examples from the TUM RGB-D benchmark [22] in which key assumptions made by our work and by prior approaches were violated and the methods failed due to movement in the scene or missing data.

Acknowledgements

We thank Erik Bylow and colleagues for providing their implementation [2], and Sungjoon Choi and Stephen Miller for help with experiments.

References

[1] A. Bachrach, S. Prentice, R. He, P. Henry, A. S. Huang, M. Krainin, D. Maturana, D. Fox, and N. Roy. Estimation, planning, and mapping for autonomous flight using an RGB-D camera in GPS-denied environments. *International Journal of Robotics Research*, 31(11), 2012. 1

[2] E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers. Real-time camera tracking and 3D reconstruction using signed distance functions. In *RSS*, 2013. 1, 4, 5

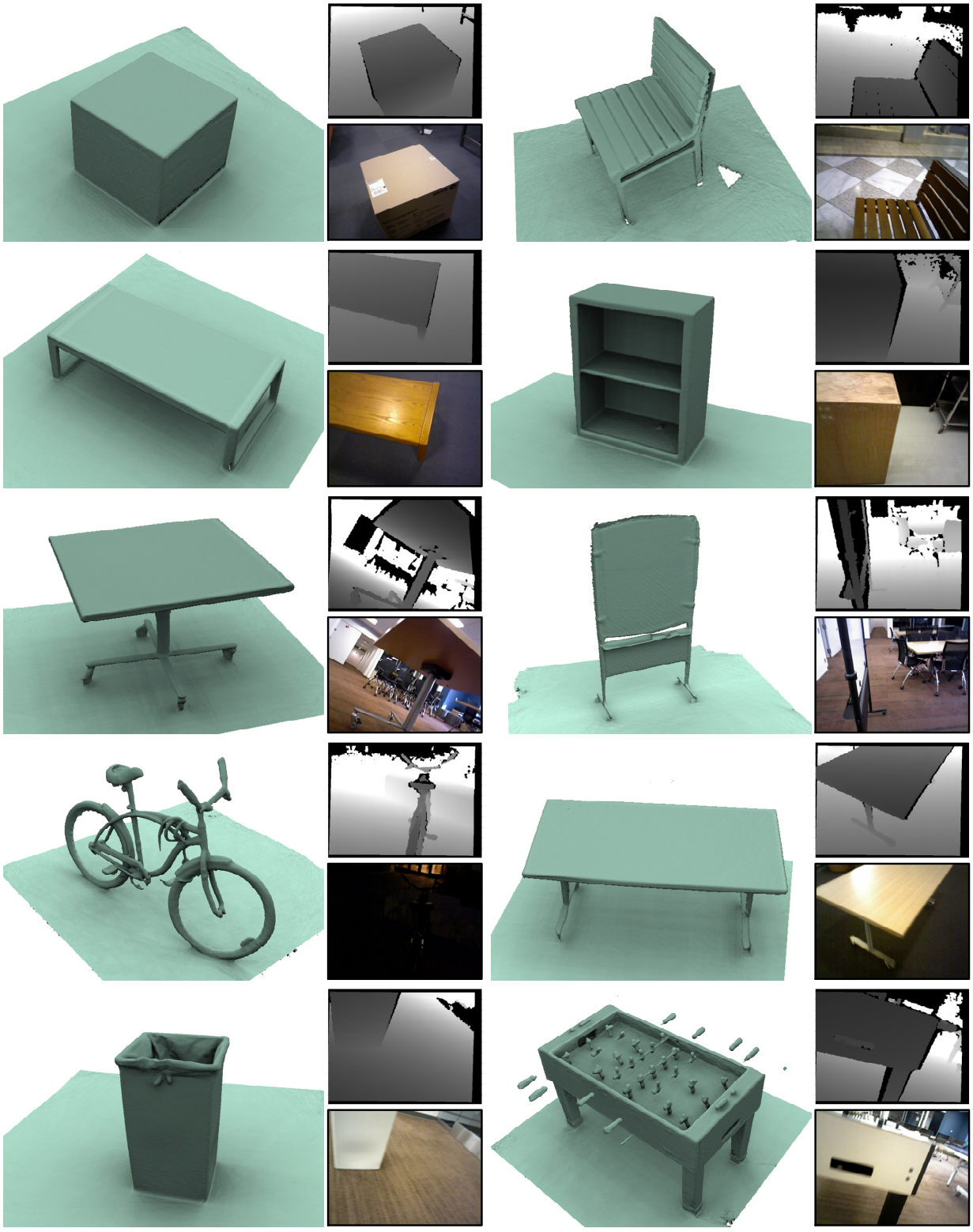


Figure 6. A variety of objects reconstructed by the presented approach. For all of these sequences, KinectFusion suffered catastrophic tracking failure at the frames shown in the insets. The presented approach stabilized camera tracking and enabled successful reconstruction.

[3] R. Cipolla and A. Blake. Surface shape from the deformation of apparent contours. *IJCV*, 9(2), 1992. 2, 3

[4] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, 1996. 2

[5] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *ICCV*, 2003. 1

[6] N. Gelfand, S. Rusinkiewicz, L. Ikemoto, and M. Levoy. Geometrically stable sampling for the ICP algorithm. In *3DIM*, 2003. 1

[7] J. J. Gibson. *The Ecological Approach To Visual Perception*. Psychology Press, 1986. 1

[8] K. Karsch, Z. Liao, J. Rock, J. T. Barron, and D. Hoiem. Boundary cues for 3D object shape recovery. In *CVPR*, 2013. 2

[9] C. Kerl, J. Sturm, and D. Cremers. Robust odometry estimation for RGB-D cameras. In *ICRA*, 2013. 1, 4, 5

[10] K. Khoshelham and S. O. Elberink. Accuracy and resolution of Kinect depth data for indoor mapping applications. *Sensors*, 12(2), 2012. 2

[11] G. Klein and D. W. Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, 2007. 1

[12] J. J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13(3), 1984. 2

[13] K. Konolige and P. Mihelich. Technical description of Kinect calibration, 2012. http://wiki.ros.org/kinect_calibration/technical. 2

[14] D. Marr. Analysis of occluding contour. *Proceedings of the Royal Society of London*, 197, 1977. 2

[15] P. Merrell, A. Akbarzadeh, L. Wang, P. Mordohai, J. Frahm, R. Yang, D. Nistér, and M. Pollefeys. Real-time visibility-based fusion of depth maps. In *ICCV*, 2007. 2

[16] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *ISMAR*, 2011. 1, 2, 3, 4, 5

[17] C. V. Nguyen, S. Izadi, and D. Lovell. Modeling Kinect sensor noise for improved 3D reconstruction and tracking. In *3DIMPVT*, 2012. 3

[18] D. Nistér, O. Naroditsky, and J. R. Bergen. Visual odometry. In *CVPR*, 2004. 1

[19] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *3DIM*, 2001. 1

[20] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *ICRA*, 2011. 4

[21] Q. Shan, B. Curless, Y. Furukawa, C. Hernandez, and S. M. Seitz. Occluding contours for multi-view stereo. In *CVPR*, 2014. 2

[22] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IROS*, 2012. 2, 4, 5

[23] H. Vu, P. Labatut, J. Pons, and R. Keriven. High accuracy and visibility-consistent dense multiview stereo. *PAMI*, 34(5), 2012. 2

[24] R. Wang, J. Choi, and G. G. Medioni. 3D modeling from wide baseline range scans using contour coherence. In *CVPR*, 2014. 2

[25] W. H. Warren. Self-motion: Visual perception and visual control. In *Perception of Space and Motion*. Academic Press, 1995. 1

[26] T. Whelan, H. Johannsson, M. Kaess, J. Leonard, and J. McDonald. Robust real-time visual odometry for dense RGB-D mapping. In *ICRA*, 2013. 1, 4, 5