Real-Time Salient Closed Boundary Tracking via Line Segments Perceptual Grouping

Xuebin Qin*, Shida He, Camilo Perez Quintero, Abhineet Singh, Masood Dehghan and Martin Jagersand

***Abstract*— This paper presents a novel real-time method for tracking salient closed boundaries from video image sequences. This method operates on a set of straight line segments that are produced by line detection. The tracking scheme is coherently integrated into a perceptual grouping framework in which the visual tracking problem is tackled by identifying a subset of these line segments and connecting them sequentially to form a closed boundary with the largest saliency and a certain similarity to the previous one. Specifically, we define a new tracking criterion which combines a grouping cost and an area similarity constraint. The proposed criterion makes the resulting boundary tracking more robust to local minima. To achieve real-time tracking performance, we use Delaunay Triangulation to build a graph model with the detected line segments and then reduce the tracking problem to finding the optimal cycle in this graph. This is solved by our newly proposed closed boundary candidates searching algorithm called "Bidirectional Shortest Path (BDSP)". The efficiency and robustness of the proposed method are tested on real video sequences as well as during a robot arm pouring experiment.**

I. INTRODUCTION

Closed boundaries are common elements in real world scenes. Hence, real-time closed boundary tracking is important in robot vision. As an example, consider Fig. 1, where the rim contour of a bowl is tracked. In real-world indoor images and robot applications, conventional trackers can fail, e.g. in the presence of texture-less target regions, non-rigid deformations, non-Lambertian surfaces, changing lighting conditions, cluttered background and drastic changing of supportive regions.

Template-based high DOF trackers can estimate image transformations such as affine and homography of a planar object region (or contour) from one frame to the next [1]. Many template trackers using different appearance models [2], [3], [4] and search methods [5], [6] have been proposed and achieve good performance. However, they are highly dependent on stable textures and sensitive to the presence of local minima. Keypoints based approaches [7], [8] are relatively robust to local minima, but they require many accurate feature points to be detected that can be difficult to achieve in practice.

Non-planar contours tracking can be approached as a pose estimation problem [9] when 3D models are available. In unstructured, natural environments 3D models are seldom available. In these cases non-planar contours tracking can be approached as non-rigid tracking. Pixel-wise segmentation based [10] and contour evolving based [11], [12], [13] methods are usually employed to track those objects. However, segmentation based methods do not work well in tracking targets whose appearances change significantly or targets which are comprised of several regions with great differences. Given an initial contour from the previous frame, contour based tracking is performed by searching the target contour based on minimizing a suitable energy [1]. These methods are more likely to trap in local minima in cluttered environments [14].

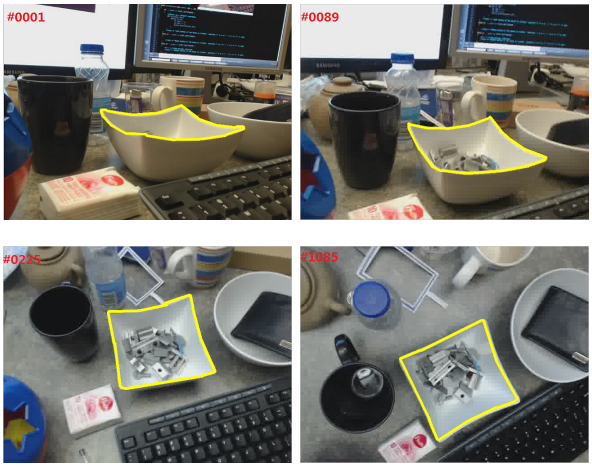


Fig. 1. Tracking the rim boundary of a bowl with the following characteristics: (1) the rim of the bowl is non-planar, (2) the bowl itself has no salient stable textures, (3) the viewpoint changes dramatically.

Another promising technique for closed boundary tracking is perceptual grouping, which has been widely used in salient closed boundary extraction from static images [15], [16], [17]. Schoenemann and Cremers [18] perform contour tracking by integrating pixel-wise perceptual grouping and elastic shape matching into one solvable optimization problem. Unfortunately, this method is not real-time without using GPU processing, ruling out light weight or low power robotics applications. In [19], rough contour tracking and shape context matching are performed separately to improve time efficiency and handle cluttered background. However, the use of a fixed shape template restricts the ability to track non-planar closed boundaries with out-of-plane motion.

Despite all different attempts, tracking of boundary targets in unstructured environment is still a challenging problem.

The authors are with the Dept. of Computing Science, University of Alberta, Canada. {xuebin, shida3, caperez, asingh1, masood1, mj7}@ualberta.ca.

Xuebin was supported by China Scholarship Council and University of Alberta.

(a) (b) (c) (d)

Fig. 2. Illustration of closed boundary tracking: (a) Current image and the tracked boundary (B_p) from the last frame (yellow polygon), (b) Detected line segments, (c) Gap filling among line segments in the buffer (green) region of boundary B_p, (d) Tracked boundary (B_c) of the current frame.

This paper address this problem by presenting a novel line segments grouping based method for tracking salient closed boundaries. Our key contributions are:

- We define a salient closed boundary tracking criterion by combining a boundary grouping cost [17] and a regularization constraint on the boundary's area variation, which improves the tracking robustness greatly.
- We construct a graph model $G(V, E)$ with the detected line segments by Delaunay Triangulation and develop a novel real-time searching method called Bidirectional Shortest Path searching algorithm (BDSP) for optimal closed boundary searching.
- To evaluate the performance of our tracking scheme, we collected and annotated nine video sequences (9598 frames) of typical closed boundaries, which are challenging to track in real robot applications.

We implement our tracking method[1] and test it on our newly collected real world video sequences[2], and compare it against state-of-the-art trackers: RKLT [20], ESM [6], HoughTrack [10] and a tracker adapted from the contour grouping method RRC [17]. We also test the performance of our method during a real robot arm experiment following a moving bowl and pouring cereal into it.

The remainder of this paper is organized as follows. Section II formulates the problem by introducing the newly proposed tracking criterion. Section III presents the details of the graph modeling and the novel optimization algorithm. Section IV describes experimental results on the newly built real world dataset and an application of robot arm pouring. A brief conclusion is given in Section V.

II. PROBLEM FORMULATION

Given a video sequence, we refer to the process of extracting corresponding salient closed boundaries from sequential video frames as boundary tracking. The target closed boundary is usually initialized by selecting a coarse polygon manually in the first frame. The main idea of the coming frames' boundary tracking is identifying a subset of line segments produced by line detector and connecting them to form a closed boundary which corresponds to the boundary tracked in the previous frame, see Fig. 2.

[1]https://github.com/NathanUA/SalientClosedBoundaryTracking
[2]https://github.com/NathanUA/SalientClosedBoundaryTrackingDataset

Therefore, the closed boundary tracking problem can be reduced to a prior shape constrained perceptual grouping problem. We define a tracking criterion which takes both grouping cost and shape constraint into consideration. The grouping cost Γ, introduced in [17], is given by:

$$\Gamma_B = \frac{|B_G|}{\iint_{R(B)} dx\, dy}, \tag{1}$$

where $|B_G|$ denotes the summation of the gap filling segments (blue segments in Fig.2c) length along the boundary B. The denominator $\iint_{R(B)} dx\, dy$ is the area of region $R(B)$ which is enclosed by the boundary B.

Although the distance based filtering (the green region shown in Fig. 2c) excludes many unrelated line segments, the above grouping cost still can not handle grouping illusions caused by noisy segments. As shown in Fig. 3, it is clear that $\Gamma_{B_1} < \Gamma_{B_2}$ which results in an incorrectly grouped boundary B_1. To eliminate this kind of error without significantly increasing time complexity, we propose to use a simple area similarity $S_{Bp\_Bc} < S_e$ (e.g. 0.9) between the searched boundary (B_c) and the prior shape (B_p) to constrain the grouping process as follows:

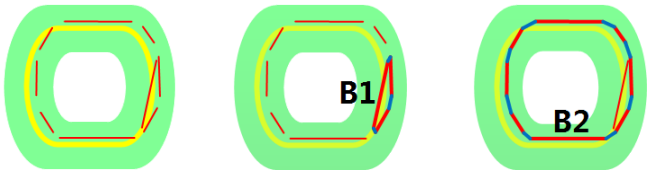
$$S_{Bp\_Bc} = min(\frac{\iint_{R(Bp)} dxdy}{\iint_{R(Bc)} dxdy}, \frac{\iint_{R(Bc)} dxdy}{\iint_{R(Bp)} dxdy}), \tag{2}$$

where $\iint_{R(Bp)} dxdy$ is the area of region $R(Bp)$ which is enclosed by the prior shape boundary Bp. Bp is the initialization or the tracked boundary from the last frame. (Bc is the to-be-tracked boundary).

To solve the grouping problem, we map the detected and generated (fill-in) line segments and their endpoints to an undirected graph $G = (V, E)$. Line segments and endpoints correspond to graph edges and graph vertices respectively, and thus closed boundaries correspond to graph cycles. Now, the problem of closed boundary grouping is converted into an optimal graph cycle searching problem. We develop a novel graph based optimization method to find the optimal boundary (quasi-optimum) in real-time.

III. METHOD

The workflow of our salient closed boundary tracking method is shown in Fig. 4. First, we introduce the line



(a) Lines within buffer region of prior boundary (b) Wrong grouping (c) Correct grouping

Fig. 3. Illustration of wrong boundary grouping: Yellow contours are prior boundaries. Line segments noise often results in wrong grouping of boundary (B_1) without using area constraint.

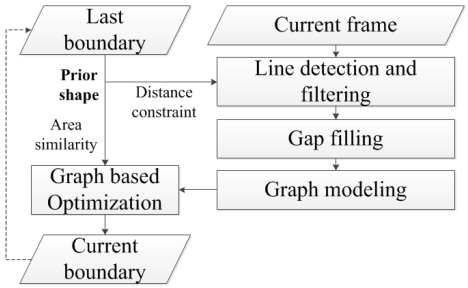


Fig. 4. Workflow of boundary tracking

detection and filtering. Then, the details of gap filling are presented, followed by graph modeling and optimization.

A. Line Detection and Filtering

Straight line segments are fundamental elements in our tracking method. We use EDLines [21], a real-time line segments detector, for automatic detection of boundary line segments. Detected line segments are represented by pairs of endpoints. In each incoming frame, line detection is conducted on the whole frame. Lines of interest are filtered by a distance constraint (with similar effect of the green buffer region shown in Fig. 2c) from the previous boundary. Specifically, three distances of a line from its two end-points and mid-point to the prior boundary are computed. If the average of these distances is smaller than certain threshold (e.g. 20 pixels), it will be retained. Lines of interest are not directly detected from the frame subregion defined by the green buffer because masking an image with an irregular buffer region takes more time.

B. Gap Filling

After obtaining the line segments of interest, Delaunay Triangulation (DT) [22] is introduced to generate virtual fragments and fill the gaps among disconnected segments, as illustrated in Fig. 5. First, line segments are simplified as endpoints (see Fig. 5a and Fig. 5b). Then, DT is conducted on these endpoints (Fig. 5c). Finally, we superimpose the detected line segments in Fig. 5a over the generated DT edges in Fig. 5c. Generated DT edges that overlap the detected line segments are removed. The final result of gap filling is an undirected graph structure as shown in Fig. 5d. To distinguish two kinds of line segments, we refer to the detected line segments (the red lines in Fig. 5d) as *detected* segments and the gap filling segments (the blue lines in Fig. 5d) as *generated* segments.

C. Graph Modeling

The gap filling process constructs the structure of an undirected graph $G = (V, E)$, which maps endpoints and segments (both *detected* and *generated*) to graph vertices V and edges E respectively. Then, we define the edge-weight function for each edge $e \in E$ similar to [17]. Given a graph edge e_i, we set its weight to

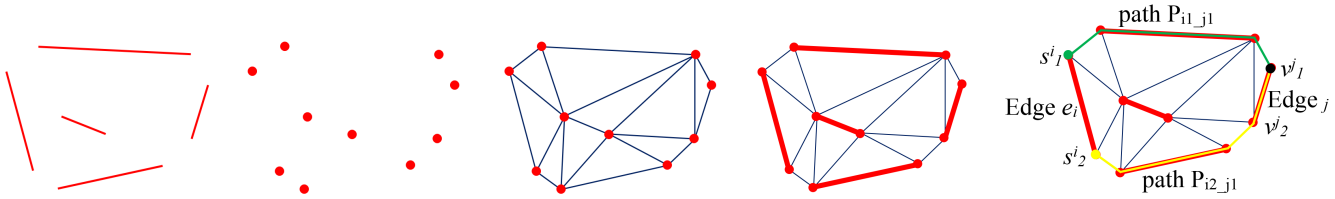
$$w(e_i) = \begin{cases} 0 & e_i \text{ is a } \textit{detected} \text{ segment} \\ |P_1^i P_2^i| & e_i \text{ is a } \textit{generated} \text{ segment} \end{cases} \quad (3)$$

where $|P_1^i P_2^i|$ is the length of the corresponding line segment $P_1^i P_2^i$ of the graph edge e_i.

D. Graph based Optimization

Now, our goal is to find the optimal graph cycle which has the minimum boundary cost, based on (1), and satisfies the similarity constraint in (2) simultaneously. The key problem is that both the boundary cost and the similarity constraint cannot be determined when the boundary itself is unknown. Furthermore, they are difficult to be integrated into one cost function. So we developed a novel heuristic search method to obtain the optimal cycle. Our method has two steps: generating boundary candidates and finding the optimal one from these candidates.

Given a graph, exhaustive searching is time-consuming and unfavorable. In an attempt to avoid an exhaustive search, we propose a method called "Bidirectional Shortest Path (BDSP)" to generate cycle candidates. This method is based on the hypothesis that a cycle with smaller total weight is more likely to be the optimal boundary. Given a graph which contains n *detected* line segments, we sample half $n/2$ of them and search $(n-1)$ (the current segment is excluded) cycle candidates for each sampled (*detected*) segment. As shown in Fig. 5e, for each sampled *detected* edge e_i(edges with odd or even indices), we search shortest paths from its two vertices s_1^i and s_2^i to the same third vertex v_1^j using Dijkstra [23]. The weight of e_i is set to infinity other than its original weight zero during searching. The edge e_i, shortest paths P_{i1_j1} and P_{i2_j1} construct a cycle candidate C_{i_j1}. Vertex v_1^j and Vertex v_2^j belong to the same edge e_j and the weight of e_j is zero, thus, taking v_1^j or v_2^j as the third vertex usually produce the same cycle. Hence, the total number of the cycle candidates is $n(n-1)/2$. Having obtained these cycle candidates, we can search for the optimal closed boundary by computing their boundary costs (1) and similarity constraints (2) easily. The optimal boundary searching algorithm is shown in Algorithm 1.



(a) Line segments (b) Endpoints (c) Gap filling by DT (d) Graph structure (e) Cycle searching

Fig. 5. (a)-(d) Graph structure construction by gap filling, (e) Cycle candidates searching by BDSP: Edge e_i is the current edge and v_1^j is the third vertex. Edge e_i, shortest paths P_{i1_j1} and path P_{i2_j1} construct a closed cycle candidate.

Algorithm 1 Optimal cycle searching

Input: Undirected graph $G = (V, E)$ and a shape prior represented by a set of ordered points

Output: The optimal cycle C_{opt}

for $i = 0; i < n; i+ = 2$ **do**
  Use BDSP to search cycle candidates $C_{i\bullet}$
  for $j = 0; j < 2(n-1); j+ = 2$ **do**
    if $i == 0 \&\& j == 0$ **then**
      $C_{opt} = C_{ij}$
    end if
    if $S_{C_{ij}\_Bp} > S_e$ **then**
      if $\Gamma_{C_{ij}} < \Gamma_{C_{opt}}$ **then**
        $C_{opt} = C_{ij}$
      end if
    end if
  end for
end for
return C_{opt}

Notes: $C_{i\bullet}$ are the $(n-1)$ cycle candidates related to edge e_i.

IV. EXPERIMENTAL RESULTS

We validate our tracking scheme using a number of comparative experiments and a real robot arm experiment.

A. Dataset and Evaluation Measure

We collected nine video sequences of salient closed boundaries, as shown in Fig. 7. Each sequence is about 30 sec (30 fps) and the frame size is 640×480 (width×height). There are 9598 frames in total. In each sequence, different motion styles such as translation, rotation and viewpoint changing are all performed. We annotate them by drawing polygons which are well matched with the salient closed boundaries in human vision.

To evaluate our proposed method quantitatively, we define the error metric as the alignment error (E_{AL}) of tracked closed boundary and ground truth (B_{gt}) as: $E_{AL} = max\{\frac{B_i \otimes Dist_{B_{gt}}}{P_{B_i}}, \frac{B_{gt} \otimes Dist_{B_i}}{P_{B_{gt}}}\}$, where $\otimes$ denotes convolution, B_i is the boundary binary image, $Dist_{B_i}$ is the distance transform image of B_i and P_{B_i} indicates the perimeter of boundary B_i. We use the success rate to measure a tracker's overall accuracy [24]. The success rate on a sequence is defined as the ratio of frames where the tracking error E_{AL} is less than a threshold of e_p pixels and the total frames.

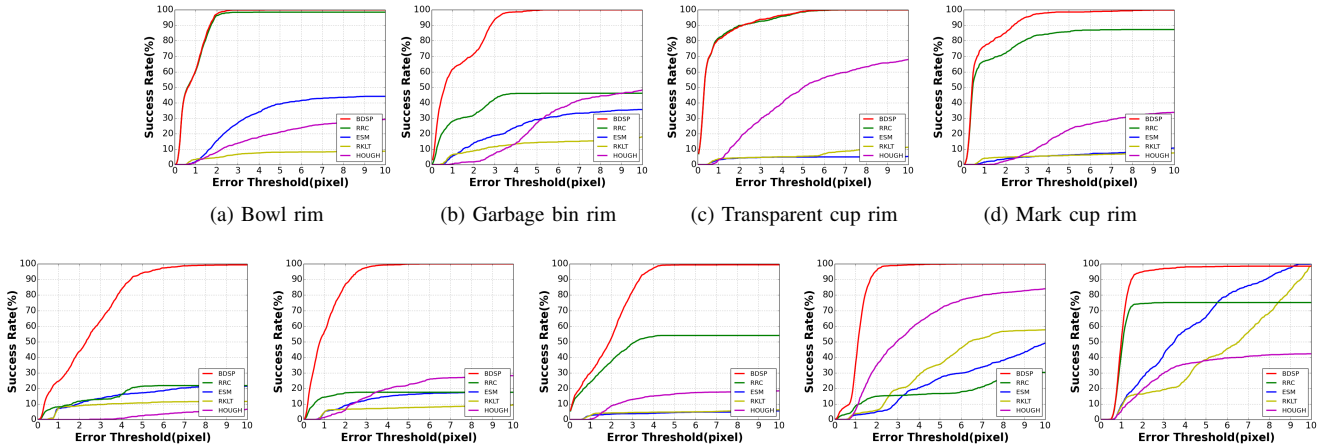
B. Results

We compare our tracking method BDSP against following methods: ESM [6] which is a popular registration based homography tracker, RKLT [20] which is a cascade registration based tracker that can handle partial appearance changing and occlusion by RANSAC, HoughTrack [10] which is a state-of-the-art segmentation based tracker that can provide us accurate contour and a tracker adapted from edge grouping method RRC [17]. Both ESM and RKLT are tested with appearance model NCC, which are implemented in a modular tracking framework (MTF) [24]). We initialize them by selecting a quadrilateral which encloses the target boundary at the first frame. Then boundaries of following frames are computed by homography transformations with respect to the first frame (all boundaries are assumed to be planar). We modified RRC by substituting its line detector for EDlines [21] and added a buffer search region as shown in Fig. 2c.

The success rate curves of the above four methods and our method are illustrated in Fig. 6. As we can see, the proposed method (BDSP) performs better than others in almost all of these sequences. Both registration based trackers ESM and RKLT fail quickly because they are sensitive to appearance changing, as blue and cyan boundaries shown in Fig. 7a to Fig. 7g. Although HoughTrack performs better than registration based trackers thanks to its model updating, it corrupts quickly when the content of target region are heterogeneous as pink boundaries shown in Fig. 7. Fig. 7a and Fig. 7c show an empty bowl and an empty transparent cup with relative clean background. The corresponding tracking results illustrated in Fig. 6a and Fig. 6c show that the RRC tracker produces almost the same success rate with our method. But it is very susceptible to noise, as the tracked green boundaries shown in Fig. 7. The intact video results are included in the supplementary video[3].

We also measured the average processing speed of our method for each of the nine video sequences on a machine with a quad core 3.10 GHz Intel Core i5 processor, 16 GB RAM and Ubuntu 14.04 64-bit OS. Our method is implemented in C++ using OpenCV and Boost library. Table.

[3]https://youtu.be/RXjD0yHkukI



(a) Bowl rim (b) Garbage bin rim (c) Transparent cup rim (d) Mark cup rim

(e) Mark cup pouring tea (f) Tool box rim (g) Non-planar bowl rim (h) Mark cup contour (i) Book stand contour

Fig. 6. Success rates of BDSP, RRC, ESM, RKLT and HoughTrack on the video sequences of Fig. 7.

TABLE I. GRAPH SCALE AND TIME EFFICIENCY

Video	Fig.7a	Fig.7b	Fig.7c	Fig.7d	Fig.7e	Fig.7f	Fig.7g	Fig.7h	Fig.7i
Edges	445	508	550	672	753	362	419	505	286
Nodes	80	88	96	112	124	62	70	89	53
lt(ms)	4.34	8.78	4.15	4.69	5.17	4.33	5.09	5.07	3.79
gt(ms)	14.45	18.85	19.00	26.10	32.13	8.68	16.93	17.72	6.56
fps	54.12	37.55	43.21	32.48	26.80	76.81	59.74	43.88	96.57

Notes: ms denotes millisecond.

I illustrates the average graph size, which is indicated by numbers of edges and nodes, the average time costs of line detection (lt), grouping time (gt) and the average frequency per second (fps). The total tracking time per each frame contains line detection time and grouping time. We compute the instantaneous fps of each frame and then average them over the whole sequence to get the average fps, as shown in the last row of Table I. As we can see our method is acceptable for real-time tracking.

C. Robot Arm Pouring Experiment

Our salient closed boundary tracker has been used successfully in a real robot arm pouring experiment. The task is to track and follow a moving bowl and then pour cereal into it. The difficulties of this experiment are that the bowl is non-Lambertian and has no salient textures.

The setup of our experiment is shown in Fig. 8a. The system includes a set of WAM arm and a Kinect. The 3D coordinates of the WAM arm and the Kinect are registered. We initialize the bowl rim and track it in RGB video stream captured by the Kinect. Meanwhile, we map the tracked closed boundary, which is represented by a set of 2D image points, to 3D points acquired by the Kinect depth sensor, as shown in Fig. 8b and Fig. 8c. The centroid of the bowl is computed based on these mapped 3D contour points and is taken as the pouring target position. We pre-compute the shifting of the WAM hand to the bowl centroid. When the distance $Dist_{r\_b}$ between the centroid of the tracked bowl and the robot hand satisfies certain thresholds($e_{low} < Dist_{r\_b} < e_{high}$), the WAM arm will pour the cereal into the bowl, as shown in Fig. 8d. Without having the tracking points of the contour provided by our tracker it will be very difficult to find the 3D center of the bowl with any other types of trackers. The experiment shows that our tracker is stable and efficient in real robot application. A demonstration of the pouring task can be seen in the accompanying video.

V. CONCLUSIONS

We presented a novel real-time method for salient closed boundary tracking. By combining a saliency measure and an area constraint as tracking criterion, the proposed method improves the tracking performance greatly. A bidirectional shortest path based boundary candidates searching algorithm enables the real-time solvability of the combined tracking criterion. We validated it quantitatively on various real-world video sequences. Since it is robust and fast enough, it has been used successfully in real robot pouring experiment where other trackers have failed. Our future work will focus on addressing the problem of tracking boundaries, which are hard to be described by straight line segments, as well as the problem of self-occlusions.

REFERENCES

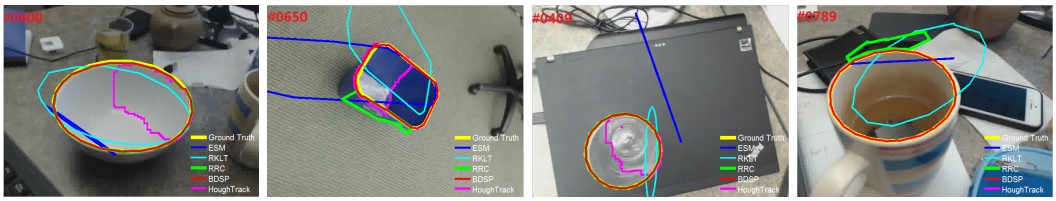
[1] A. Yilmaz, X. Li, and M. Shah, "Object contour tracking using level sets," in *Asian Conference on Computer Vision*, 2004.

[2] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proceedings of the 7th International Joint Conference on Artificial Intelligence, IJCAI '81, Vancouver, BC, Canada, August 24-28, 1981*, 1981, pp. 674–679.

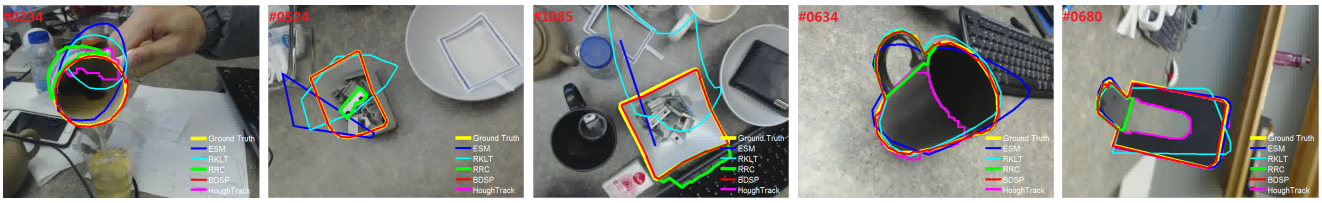
[3] G. G. Scandaroli, M. Meilland, and R. Richa, "Improving ncc-based direct visual tracking," in *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI*, 2012, pp. 442–455.

[4] N. D. H. Dowson and R. Bowden, "Mutual information for lucas-kanade tracking (MILK): an inverse compositional formulation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 1, pp. 180–185, 2008.

[5] S. Baker and I. A. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International Journal of Computer Vision*, vol. 56, no. 3, pp. 221–255, 2004.

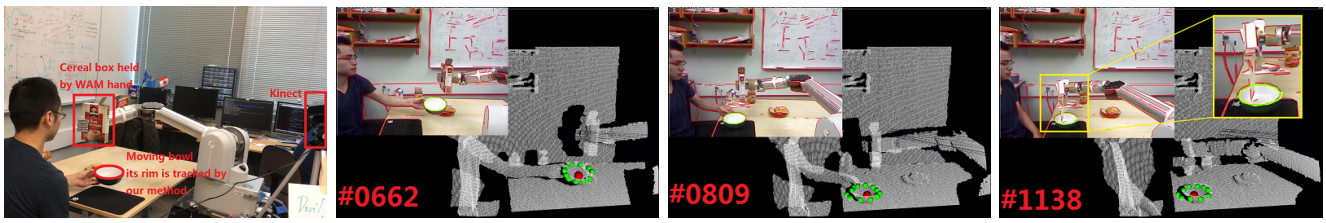


(a) Bowl rim (b) Garbage bin rim (c) Transparent cup rim (d) Mark Cup rim



(e) Mark cup rim pouring (f) Tool box rim (g) Non-planar bowl rim (h) Mark cup contour (i) Book stand contour

Fig. 7. Tracked boundaries on typical frames.



(a) Experiment setup (b) Following (c) Following (d) Following and accurate pouring

Fig. 8. Robot pouring experiment: (a) A human is moving the bowl continuously under the surveillance of a Kinect. (b)(c) Our algorithm tracks the bowl rim and maps its 2D image points to 3D points in the robot coordinates through the Kinect which is registered with the robot coordinates, then, makes the robot arm follows the moving bowl. (d) The robot hand pours the cereal into the bowl accurately according to our tracking result.

[6] S. Benhimane and E. Malis, "Real-time image-based tracking of planes using efficient second-order minimization," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, September 28 - October 2, 2004*, 2004, pp. 943–948.

[7] J. Shi and C. Tomasi, "Good features to track," in *Conference on Computer Vision and Pattern Recognition, CVPR 1994, 21-23 June, 1994, Seattle, WA, USA*, 1994, pp. 593–600.

[8] S. Gauglitz, T. Höllerer, and M. Turk, "Evaluation of interest point detectors and feature descriptors for visual tracking," *International Journal of Computer Vision*, vol. 94, no. 3, pp. 335–360, 2011.

[9] C. Choi and H. I. Christensen, "3d textureless object detection and tracking: An edge-based approach," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2012, Vilamoura, Algarve, Portugal, October 7-12, 2012*, 2012, pp. 3877–3884.

[10] M. Godec, P. M. Roth, and H. Bischof, "Hough-based tracking of non-rigid objects," *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1245–1256, 2013.

[11] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. J. Yezzi, "Tracking deforming objects using particle filtering for geometric active contours," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 8, pp. 1470–1475, 2007.

[12] C. Bibby and I. D. Reid, "Real-time tracking of multiple occluding objects using level sets," in *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, 2010, pp. 1307–1314.

[13] X. Sun, H. Yao, S. Zhang, and D. Li, "Non-rigid object contour tracking via a novel supervised level set model," *IEEE Trans. Image Processing*, vol. 24, no. 11, pp. 3386–3399, 2015.

[14] M. Pressigout and É. Marchand, "Real time planar structure tracking for visual servoing: a contour and texture approach," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, Edmonton, Alberta, Canada, August 2-6, 2005*, 2005, pp. 251–256.

[15] J. H. Elder and S. W. Zucker, "Computing contour closure," in *Computer Vision - ECCV'96, 4th European Conference on Computer Vision, Cambridge, UK, April 15-18, 1996, Proceedings, Volume I*, 1996, pp. 399–412.

[16] S. Wang, T. Kubota, J. M. Siskind, and J. Wang, "Salient closed boundary extraction with ratio contour," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 4, pp. 546–561, 2005.

[17] J. S. Stahl and S. Wang, "Edge grouping combining boundary and region information," *IEEE Trans. Image Processing*, vol. 16, no. 10, pp. 2590–2606, 2007.

[18] T. Schoenemann and D. Cremers, "A combinatorial solution for model-based image segmentation and real-time tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 7, pp. 1153–1164, 2010.

[19] Z. Liu, H. Shen, G. Feng, and D. Hu, "Tracking objects using shape context matching," *Neurocomputing*, vol. 83, pp. 47–55, 2012.

[20] X. Zhang, A. Singh, and M. Jägersand, "RKLT: 8 DOF real-time robust video tracking combing coarse ransac features and accurate fast template registration," in *12th Conference on Computer and Robot Vision, CRV 2015, Halifax, NS, Canada, June 3-5, 2015*, 2015, pp. 70–77.

[21] C. Akinlar and C. Topal, "Edlines: A real-time line segment detector with a false detection control," *Pattern Recognition Letters*, vol. 32, no. 13, pp. 1633–1642, 2011.

[22] F. P. Preparata and M. I. Shamos, *Computational Geometry - An Introduction*, ser. Texts and Monographs in Computer Science. Springer, 1985.

[23] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[24] A. Singh, A. Roy, X. Zhang, and M. Jägersand, "Modular decomposition and analysis of registration based trackers," in *13th Conference on Computer and Robot Vision, CRV 2016, Victoria, BC, Canada, June 1-3, 2016*, 2016, pp. 85–92.