Automatic Video Segmentation and Object Tracking with Real-Time RGB-D Data

I-Kuei Chen†, Szu-Lu Hsu†, Chung-Yu Chi†, and Liang-Gee Chen*
DSP/IC Design Lab, National Taiwan University, Taiwan
†{eugenegx, bismarck, craig}@video.ee.ntu.edu.tw *lgchen@video.ee.ntu.edu.tw

***Abstract*—In this paper, we propose a novel system that is able to automatically segment objects in real-time video by RGB-D information without human annotation (labeling). And it can track each cluster as an individual object with spatial-temporal information regardless of the difficult situation (low luminance and cluttered backgrounds). The introduced approach is efficient enough to operate tracking in real-time.**

I. Introduction

Computer vision-based segmentation techniques have been studied extensively and have been applied successfully to content-based image retrieval or object recognition. Our goal in this paper is to implement a robust real-time object tracking system with video segmentation at the same time. It also serves as a good tool for consumers to extract the real world video segmentation of the desired object.

Point Cloud Library [1] (PCL) is an open-source project, including numerous state-of-the-art algorithms for n-dimensional point clouds and 3D geometry processing. Our approach is inspired by a recent segmentation work [2] on PCL. Although the previous works in both image segmentation [3] and video segmentation [4] already generate nearly perfect results, they still need a large amount of human participation and this becomes a critical defect when the target video is too large and too long to be handled by human users. In contrast, our tracking system can perform video segmentation automatically without any human annotation.

With RGB-D information, it's possible for our system to robustly track objects even in low-luminance situation or cluttered backgrounds and to track the targeted object robustly. We will emphasize the results of these three special situations in the section of Experiment Results.

In the remainder of the paper we detail our approach with a comprehensive system diagram(Fig. 2). We then provide the experimental results for challenging complicated scenes that show our system achieves state-of-the-art accuracy and demonstrate improved generalization with automatic and real-time video segmentation tracking system for objects.

II. Algorithm

The main result of this system is showed in Fig. 1(a) and RGB-D data provides 3D position information for each pixel. We separate our algorithm into three stages. First, the system decides the main plane based on RGB-D information. But because of the low resolution and inevitable noises of the depth sensor, we refine the plane for better performance. Then the system segments out the candidate clusters which are assumed to be physically meaningful objects from the main plane. In the end, we track the cluster of the desired object in each frame and give the cluster the same label in the whole sequence. The details of the algorithm will be introduced as follows.

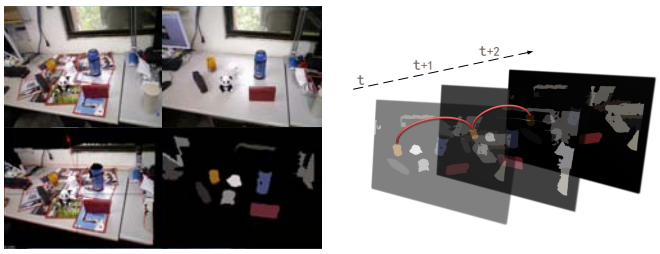
A. Main Plane Segmentation and Refinement

To get the features of geometry properties on local surfaces we adopt cross-product methods to get normal vectors of local regions. Next, the concept of connected component is applied to decide which plane the pixel belongs to [2]. Two pixels will be classified into the same plane if normal vectors and depth values of these two pixels are continuous enough. The decision function is showed as follow.

$$C(p_1,p_2)=\begin{cases}\text{true}, & \text{if } ((D_{normal}<thresh_{normal}) \\ & \&\&(D_{depth}<thresh_{depth})) \\ \text{false}, & \text{otherwise}\end{cases} \tag{1}$$

p is the point representation for each pixel with position and full plane equation. $D_{normal}(p_1,p_2)=\overrightarrow{n_{p_1}}\cdot\overrightarrow{n_{p_2}}$ indicate the angular difference between two normal directions. $D_{depth}(p_1,p_2)$ indicate the depth distance between two point. We give plane label to the connected component with enough pixels and choose the one with the largest amount of pixels to be main plane.

On account of the noise of depth sensor, the contours of planes are not smooth enough to achieve satisfied segmentation performance. Therefore, we add a plane refinement function to smooth our result as [2].



(a) (b)

Fig. 1. (a) Top-left figure shows cluttered backgrounds. Top-right shows six target objects. Bot-left shows the result of main plane segmentation. Bot-right shows candidate cluster segmentation result. (b) shows three subsequent frames with target object tracked. Red curve indicates the tracking target

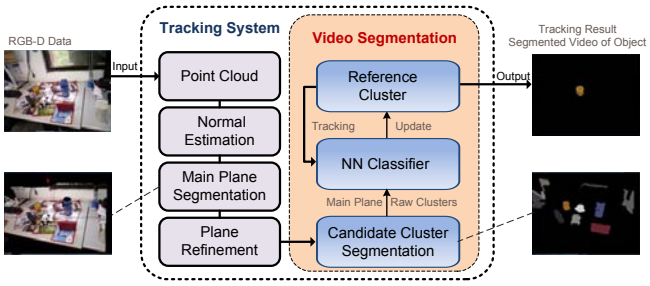


Fig. 2. Details of the proposed video segmentation and object tracking system

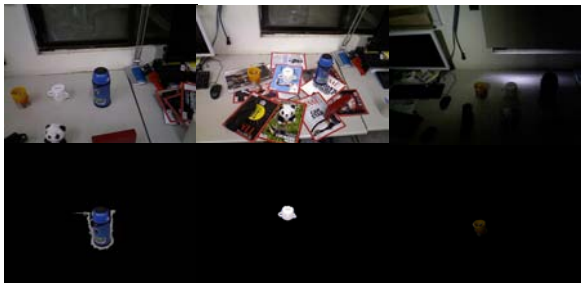


Fig. 3. Results of three sequences: bottle in original scene, bank in cluttered backgrounds and pen container in low illumination.

B. Candidate Cluster Segmentation

In this stage, we only retain the pixels on the main plane, others will be discarded to avoid redundant computation since they are not on the target plane. Before tracking, we should clustered the unlabel pixels [2]. We use (3) to cluster them.

$$C(p_1,p_2)=\begin{cases}\text{true}, & \text{if } ((D_{point-point}<\text{thresh}_{point-point}) \\ & \&\&(L(p_1),L(p_2)\notin plane\_labels)) \\ \text{false}, & \text{otherwise}\end{cases} \tag{2}$$

where $D_{point-point}(P_1,P_2)=\|P_1-P_2\|_2$ imply the difference in 3D position between these two pixels. Finally, we cut out each clustered that is close to the main plane and assume that they are physically meaningful objects.

C. Video Segmentation

After getting the location and area information of physically meaningful clusters, we track each cluster in subsequent frames like Fig. 1(b). Each candidate cluster is put into Nearest Neighbor (NN) classifier. And NN classifier calculates the difference between each cluster in current frame and the reference cluster in previous frame. A cluster with minimum difference is choosed to be the new reference cluster. The reason we choose NN classifier is it's low complexity, while the feasibility of real-time is the main concern in our system. The distance function of NN classifier is shown as follows.

$$d(C_i,C_R)=d_c(C_i,C_R)+d_s(C_i,C_R) \tag{3}$$

where $d_c(C_i,C_R)=(|l_i-l_R|+|a_i-a_R|+|b_i-b_R|)/3$ is the distance in L*a*b color space and $d_s(C_i,C_R)=\sqrt{(X_i-X_R)^2+(Y_i-Y_R)^2+(Z_i-Z_R)^2}$ is the distance of 3D position. As the result, our system is capable of tracking the single object in real-time video ,and it can work even in difficult situations like low luminance and cluttered backgrounds. We have tried many types of feature. We found including both L*a*b and location information achieve better performance.

III. Experiment Results

We implement our system on a 3.07GHz CPU and use input sensor generating RGB-D data with 640*480 resolution. We use nine sequences, including three situations (original, cluttered backgrounds, low illumination) with three objects (bank, bottle, pen container) as test dataset (40 frames in each sequence (Fig. 3)).

TABLE I
THE EXPERIMENT RESULTS WITH THREE OBJECTS AND SITUATIONS

Sequences	Bottle			Bank			Pen Container		
Situations	Original Scene	Cluttered BG	Low Light	Original Scene	Cluttered BG	Low Light	Original Scene	Cluttered BG	Low Light
PSNR	24.07	31.23	23.61	30.69	29.57	30.83	20.60	25.67	23.47
Precision	62.28	88.70	57.73	89.39	95.32	88.14	96.63	91.34	93.63
Recall	99.96	94.26	99.90	93.85	88.19	95.39	68.62	78.02	70.90
F-Score	76.75	91.40	73.18	91.57	91.62	91.62	80.35	84.16	80.69
Pixel Error	27.44	2.61	30.28	4.61	4.81	4.43	6.57	4.10	8.28
Tracking Rate	87.18	100.00	80.00	100.00	97.50	97.50	100.00	100.00	100.00

We compared our results with golden results by human labeling (Table. I). The PSNR, precision, recall, and F-score is calculated by the results of segmentation. The 2D pixel error of center points and tracking rate is computed by the results of tracking. We compute PSNR by segmented binary mask of our system and [5]. The precision is defined as the pixel ratio of true positive region and all positive region, while the recall is the ratio of true positive region and all correct result. And F-score $=2\times\text{precision}\times\text{recall}/(\text{precision}+\text{recall})$. The 2D pixel error $D=\|C_t-C_g\|_2$ is the distance between the centers of tracking result C_t and golden result C_g, and tracking rate is the portion of the frames that $D\leq 20$ in all frames.

IV. Discussion and Conclusion

We have presented a system to exploit RGB-D information with Point Cloud Library for video segmentation and automatic object tracking. It's also able to segment targeted objects on the main plane into clusters in real-time. Additionally, we have shown that our approach achieve good tracking rate but in high operation speed(about 0.3s for each frame). Most importantly, the entire system is able to segment objects and tracking them without any human participation which highly promotes the practicability of it as a consumer application.

References

[1] R. Bogdan Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," *IEEE ICRA*, Shanghai, China, May 9-13, 2011.

[2] A. Trevor, S. Gedikli, R. Rusu, H. Christensen, "Efficient Organized Point Cloud Segmentation with Connected Components," *3rd Workshop on SPME*, Karlsruhe, Germany. May, 2013.

[3] X. Bai and G. Sapiro, "A Geodesic Framework for Fast Interactive Image and Video Segmentation and Matting," *IEEE ICCV*, Rio de Janeiro, Portugal, Oct 14-21, 2007.

[4] X. Bai, J. Wang, D. Simons, G. Sapiro, "Video SnapCut: Robust Video Object Cutout Using Localized Classiers," *ACM SIGGRAPH*, New Orleans, louisiana, Aug 3-7, 2009.

[5] V. Gulshan, C. Rother, A. Criminisi, A. Blake, A. Zisserman, "Geodesic Star Convexity for Interactive Image Segmentation," *IEEE CVPR*, San Francisco, California , June 13-18, 2010.