Segmentation of Cluttered Scenes through Interactive Perception

Karol Hausman, Christian Bersch, Dejan Pangercic, Sarah Osentoski, Zoltan-Csaba Marton, Michael Beetz
{hausman, pangercic, marton, beetz}@cs.tum.edu,
christian.bersch@googlemail.com, sarah.osentoski@us.bosch.com

I. Introduction

For robot to perform its tasks competently, robustly and in the right context it has to understand the course of its actions and their consequences. For example, imagine the robot being tasked with the clean up of the breakfast table. The robot is confronted with a heavily cluttered scene and has to be able to tell waste, dirty, clean and valuable objects apart. The robot shall be equipped with the knowledge that will, for instance, stop it from throwing away an expensive item. Herein proposed approach elevates robot's perception skills in that it utilizes its capabilities to interact with the clutter of objects. This allows for better segmentation and finally also better object recognition by means of constraining the recognition to a region or regions of interest.

Similar to Katz et al. [1] and Bergstrom et al. [2], we propose a system that uses a robot arm to induce motions in a scene to enable effective object segmentation. Our system employs a combination of the following techniques: i) estimation of a contact point and a push direction of the robot's end effector by detecting the concave corners in the cluttered scene, ii) feature extraction using features proposed by Shi and Tomasi and tracking using optical flow, and iii) a novel clustering algorithm to segment the objects.

Segmentation of rigid objects from a video stream of objects being moved by the robot has been addressed by Fitzpatrick [3] and Kenney et al. [4]. In contrast, our arm motion is not pre-planned but adapts to the scene, we make use of the 3D data to segment the object candidates from the background and we use a novel clustering approach for the segmentation of textured objects.

Overview of the whole system is shown in Fig. 2. The system will be demostrated live during the workshop.

II. Estimation of Contact Point and Push Direction

Since most commonly encountered household items have convex outlines when observed from above, our system uses local concavities in the 2D contour of an object group as an indicator for boundaries between the objects. The robot separates objects from each other by pushing its end effector in between these boundaries.

A. Contact Points from Concave Corners

We restrict the problem of finding a contact point to the table plane. Our algorithm employs 2D-image processing techniques to select contact point candidates. The table plane is estimated from the depth-camera's point cloud data using RANSAC and separated from the object points. The remaining cloud points are projected into a virtual camera view above the table. Since the projected cloud points are sparse, we employ standard morphological operators and 2D-contour search to identify a closed region, R, corresponding to the group of objects.

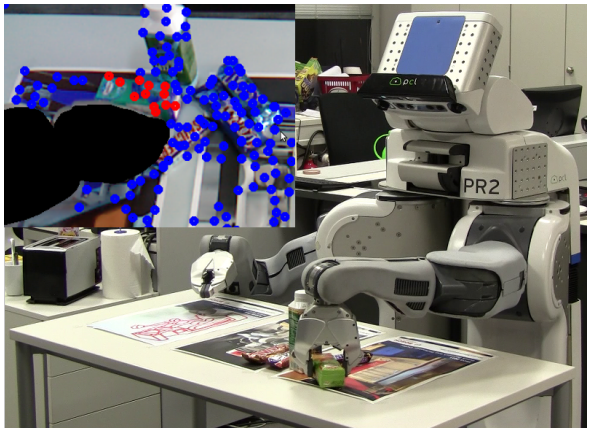

Fig. 1. Top: PR2 robot successfully picking-up the object after segmenting in it in clutter using herein proposed object segmentation algorithm.

This region's outer contour is then searched for strong local directional changes by applying a corner detector and subsequently the corners that are placed at local concavities are selected.

B. Push Direction and Execution

The push direction at a corner is set to be parallel to the eigenvector corresponding to the larger eigenvalue of the Shi-Tomasi covariance matrix. Intuitively, the dominant eigenvector will align with the dominant gradient direction. However, at a corner with two similar gradient responses in two directions, the eigenvector becomes the bisector. As only corners with roughly equal eigenvalues are chosen as potential contact point candidates, the eigenvector of each contact point candidate will bisect the angles of the contour at the corner location.

III. Object Segmentation using Feature Trajectories

Once the robot′s end effector touches the objects, the resulting object motions are used to discriminate between the different items on the table. Feature points are tracked in the scene and the resulting feature point trajectories are

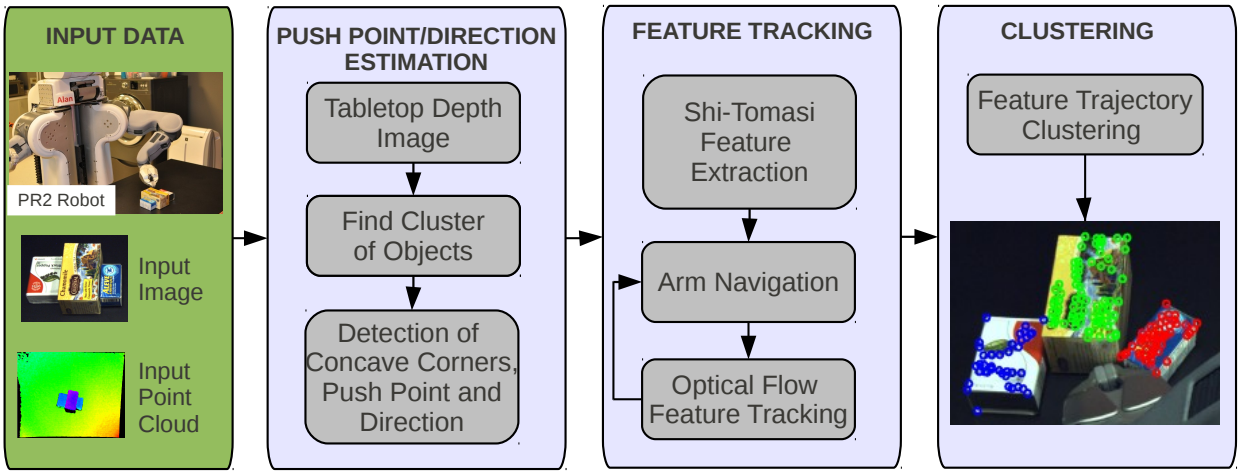

Fig. 2. The system proposed in the paper consists of three main nodes: a node for estimating the initial contact point and the push direction, a node that extracts 2D-features and tracks them while it moves the robot arm in the push direction, and finally an object clustering node that assigns the tracked features to objects.

clustered. The clustering is based on the idea that features corresponding to the same objects must follow the same translations and rotations.

A. Feature Trajectory Generation using Optical Flow

We take advantage of the objects' texture properties by extracting $i = 1...N$ Shi-Tomasi features at the pixel locations $\{\mathbf{p}_{i,0}\}_{i=1}^{N}$ from the initial scene at time $t = 0$, i.e. before an interaction with the robot took place. The feature locations correspond to responses of the Shi-Tomasi feature detector. When the robot's end effector interacts with the object, a Lucas-Kanade tracker is used to compute the optical flow of the sparse feature set. Using the optical flow, each feature's position $\mathbf{p}_{i,t}$ is recorded over the image frames at time $t = 0...T$ while the robot is interacting with the objects. That is, for each successfully tracked feature i, a trajectory $S_i = \{\mathbf{p}_{i,t}\}_{t=0}^{T}$ is obtained.

B. Randomized Feature Trajectory Clustering with Rigid Motion Hypotheses

After calculating the set of all feature trajectories $\mathcal{S} \equiv \{S_i\}_{i=1}^{N}$, the goal is to partition this set such that all features belonging to the same object are assigned the same object index $c_i \in \{1, .., K\}$, where the number of objects K is not known *a priori*.

We take advantage of the rigid body property of objects and assume that each subset of the features trajectories $\mathcal{S}$ belonging to the same object k are subjected to the same sequence of rigid transformation $A_k \equiv \{\mathbf{A}_{k,t}\}_{t=0}^{T-1}$, i.e. we cluster features with respect to how well rigid transformations can explain their motions. As the objects only move on the table plane, we restrict a possible rigid transformation $\mathbf{A}$ to be composed of a 2D-rotation $\mathbf{R}$, a 2D-translation $\mathbf{t}$ and a scaling component s, i.e. $\mathbf{A} = s \cdot [\mathbf{R}|\mathbf{t}]$. The scaling component compensates for the changes in size of the projected objects in the camera image. The actual scaling is not linear due to the perspective view, however, the error resulting from this linearization is small as the objects are displaced only in small amounts.

Algorithm 1: Randomized feature trajectory clustering

Input: Set of feature trajectories $\mathcal{S} \equiv \{S_i\}_{i=1}^{N}$ where $S_i = \{\mathbf{p}_{i,t}\}_{t=0}^{T}$
Output: object cluster count K, object cluster assignments $\mathbf{c} = [c_i]_{i=1}^{N}$ where $c_i \in \{1, .., K\}$
for $m := 1$ *to* M **do**
  $k_m := 1$, $\mathcal{S}_m := \mathcal{S}$
  while $|\mathcal{S}_m| \geq 2$ **do**
    draw 2 random trajectories $S_u, S_v \in \mathcal{S}_m$
    generate sequence of rigid transformations: $A_{k_m} \equiv \{\mathbf{A}_{k_m,t}\}_{t=0}^{T-1}$ from (S_u, S_v)
    for S_j *in* $\mathcal{S}_m$ **do**
      sum squared residuals w.r.t to A_{k_m}: $r_{k_m,j} := \sum_{t=0}^{T-1} \|\mathbf{p}_{j,t+1} - \mathbf{A}_{k_m,t}\mathbf{p}_{j,t}\|_2^2$
      if $r_{k_m,j} < THRESHOLD$ **then**
        $\mathcal{S}_m := \mathcal{S}_m \setminus \{S_j\}$
    $k_m := k_m + 1$
  $K_m := k_m$
  for S_i *in* $\mathcal{S}$ **do**
    Assign each trajectory to best matching rigid transformation sequence: $c^*_{m,i} := \text{argmin}_{\{1,..,k_m,..,K_m-1\}} r_{k_m,i}$, where $r_{k_m,i} := \sum_{t=0}^{T-1} \|\mathbf{p}_{i,t+1} - \mathbf{A}_{k_m,t}\mathbf{p}_{i,t}\|_2^2$
Select best overall matching set of rigid transform sequences: $m^* := \text{argmin}_m \sum_{k_m=1}^{K_m} \frac{\sum_i r_{k_m,i} \cdot \mathbf{1}_{[c^*_{m,i}=k_m]}}{\sum_i \mathbf{1}_{[c^*_{m,i}=k_m]}}$
Return: $K := K_{m^*}$, $\mathbf{c} := \left[c^*_{m^*,i}\right]_{i=1}^{N}$

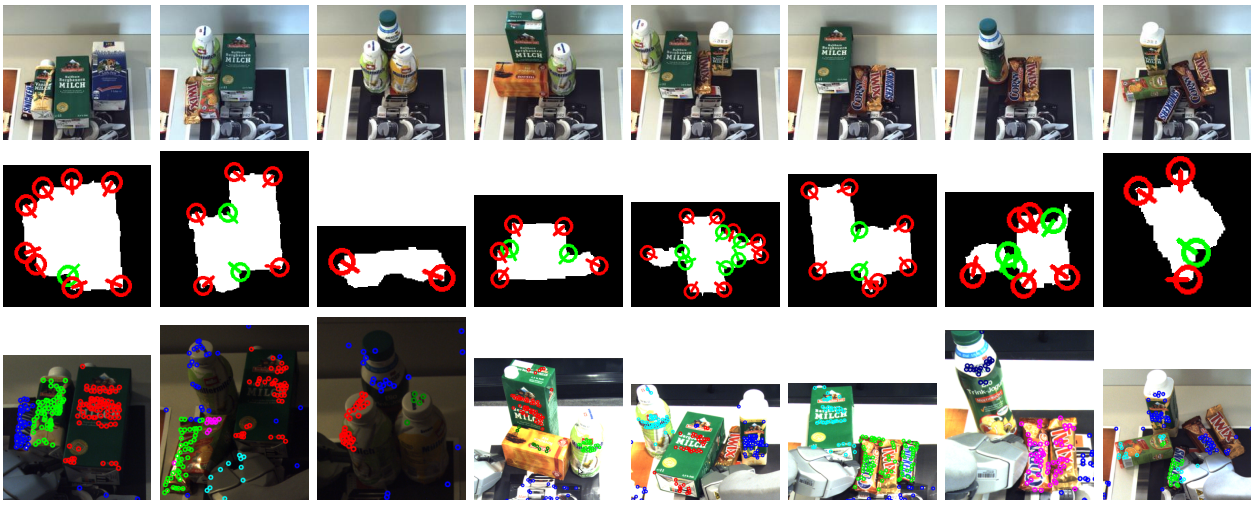

Fig. 3. Test scenes 1 to 8 from left to right. Top row: original scenes, middle row: contact point estimation, bottom row: segmentation after the first push cycle. Please note, that successfully segmented objects were removed from the scene and the contact point estimation and segmentation cycle were repeatedly executed.

The clustering algorithm we propose is outlined in Alg. 1, and combines a divisive clustering approach with RANSAC-style model hypothesis sampling. At the core of the algorithm (lines 4–12), we randomly draw 2 tracked features u,v and estimate a sequence of rigid transformations A_1 from their optical flow motions as first model hypothesis. The feature trajectories S_i that can be explained well by A_1 are considered "model inliers" and are removed from set of feature trajectories. From the remaining set, again 2 features are drawn to create a second model hypothesis A_2 and all inliers are removed. This process repeats until there are not enough features left to create a new model hypothesis. This process results in K hypotheses.

IV. Experiments

Our system was deployed on Willow Garages PR2 robot. Depth images were taken from a Kinect sensor mounted on the robots head and the PR2s built-in 5-megapixel camera was used for capturing images for feature extraction and tracking.

A. Segmentation of Objects in Cluttered Scenes

We evaluated our system on eight tabletop scenes with the cluttered background shown in Fig. 3. For each scene, the original setup of objects, the detected contact point candidates and push directions, and the feature clusters after the first push cycle are shown in the respective row. Across all runs using corner-based pushing 89% of all objects were segmented successfully.

The segmentation of the scenes took 1.047 seconds on average to compute, which also demonstrates that our algorithm is suitable for real world settings.

B. Grasping

We also ran a grasping experiment on the scene 8 (Fig.3). In this experiment, we use low-quality image from the Kinect for the segmentation and an associated point cloud for the calculation of the object pose. The accompanying video[1] is showing the above mentioned experiment.

C. Open Source Code

We provide the software[2] and documentation[3] as an open source. In the workshop we plan to demostrate the segmentation of textured objects using Kinect sensor and manually interaction with the objects.

V. Future Work

The results show applicability of our system for objects of various sizes, shapes and surface. Future work includes integrating our approach with other object segmentation techniques in order to account for textureless objects and to further improve the segmentation rate. We also plan to integrate an arm motion and a grasp planner which will enable the robot to perform robust grasping and deal with even more complex scenes.

References

[1] D. Katz and O. Brock, "Interactive segmentation of articulated objects in 3d," in *Workshop on Mobile Manipulation at ICRA*, 2011.

[2] N. Bergström, C. H. Ek, M. Bjrkman, and D. Kragic, "Scene understanding through interactive perception," in *In 8th International Conference on Computer Vision Systems (ICVS)*, Sophia Antipolis, September 2011.

[3] P. Fitzpatrick, "First contact: an active vision approach to segmentation," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2003.

[4] J. Kenney, T. Buckley, and O. Brock, "Interactive segmentation for manipulation in unstructured environments," in *Proceedings of the 2009 IEEE international conference on Robotics and Automation*, ser. ICRA'09, 2009.

[1] http://youtu.be/4VVov6E3iiM

[2] http://ros.org/wiki/pr2_interactive_segmentation

[3] http://ros.org/wiki/pr2_interactive_segmentation/Tutorials