Surface Oriented Traverse for Robust Instance Detection in RGB-D

Ruizhe Wang Gérard G. Medioni Wenyi Zhao

***Abstract*— We address the problem of robust instance detection in RGB-D image in the presence of noisy data, cluttering, partial occlusion and large pose variation. We extract contour points from the depth image, construct a Surface Oriented Traverse (SOT) feature for each contour point and further classify it as either belonging or not belonging to the instance of interest. Starting from each contour point, its SOT feature is constructed by traversing and uniformly sampling along an oriented geodesic path on the object surface. After classification, all contour points vote for an instance-specific saliency map, from which the instance of interest is finally localized. Compared with the holistic template-based and learning-based methods, our method inherits advantages of the feature-based methods in dealing with cluttering, partial occlusion, and large pose variation. Furthermore, our method does not require accurate 3D models or high quality laser scan data as input and takes noisy data from commodity 3D sensors. Experimental results on the public RGB-D Object Dataset and our FindMe RGB-D Dataset demonstrate the effectiveness and robustness of our proposed instance detection algorithm.**

I. Introduction

Accurate and robust instance detection is a fundamental problem in robotics and perception, and it plays a key role in many complex high-level robotic operations, e.g., navigation and object retrieval. With the recent advances in 3D sensing technologies, substantial improvements have been made over the traditional 2D approaches by incorporating an additional depth channel. However, the problems of cluttering, partial occlusion, large pose variation, and low quality input data from commodity 3D sensors still remain challenging.

Traditionally holistic template-based methods [10], [11], [22] and learning-based methods [4], [14], [17], [19], [21], [25], [30] prevail in the field of instance detection from RGB-D. Both types of methods detect the instance using a sliding window approach either with multiple templates or a learned model. Both paradigms, however, have several drawbacks in common. First, it is not trivial to deal with partial occlusion. While the most efficient holistic methods [10], [11], [17], [22] fail, improved methods with visibility reasoning [4], [19], [30] inevitably increase the computational cost. Second, when coping with large pose variation, especially wild camera in-plane rotations, more templates or more training images of the object are required, which inevitably leads to a higher confusion rate with either other objects or the background. Third, most methods assume accurate 3D models available either for direct use or for rendering [4], [10], [11], [19], [22], [25], [30], it is not, however, an easy task to obtain such models for common objects in daily life. While methods with high-quality laser scanners are too expensive, solutions with commodity depth sensors [20] fail to reconstruct small or symmetric objects due to sensor resolution limitations and geometric alignment failures respectively.

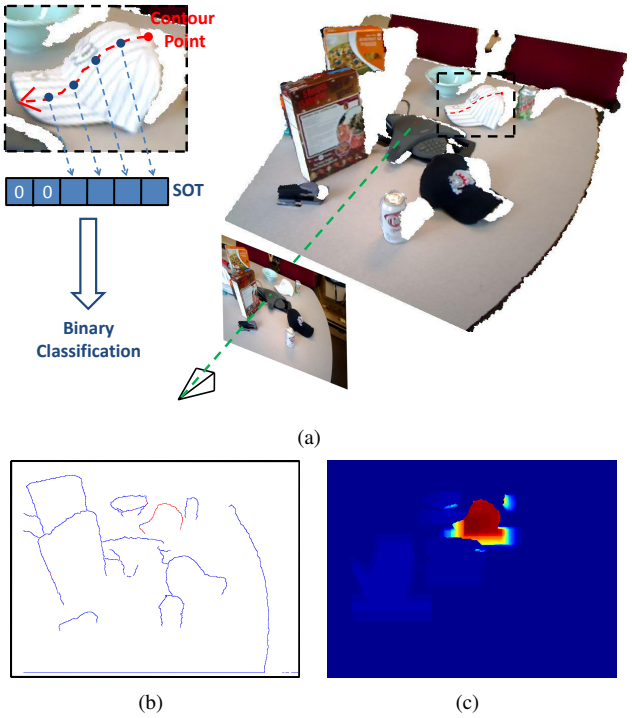


Fig. 1. (a) A 3D scene generated from a RGB-D image and SOT feature is constructed for one example contour point (red dot on the white cap), by traversing and uniformly sampling (blue dots) along an oriented geodesic path on the object surface (red dash line). After construction, SOT feature is passed into a binary classifier. (b) All contour points are classified as either instance (i.e., white cap) contour points (red) or non-instance contour points (blue). (c) An instance-specific saliency map generated by voting all classified contour points.

Feature-based methods for instance detection in intensity images [18] have been successful when dealing with partial-occlusion and large pose variation, mainly due to the invariant nature of the constructed 2D features. More recently, inspired by the success of the 2D feature, several 3D features have been developed [3], [8], [9], [12], [15], [23], [27], [28], [31] and applied towards 3D instance detection [15], [26], [27]. These feature-based instance detection methods usually follow an interest points detection, feature extraction, feature matching, geometric alignment and false rejection paradigm.

R. Wang and G. Medioni are with the Department of Computer Science, University of Southern California, Los Angeles, California, USA. {ruizhewa,medioni}@usc.edu

W. Zhao is with DAQRI Inc., Los Angeles, California, USA. wenyi.zhao@daqri.com

While the discernability of 3D features heavily relies on the quality of 3D data, most feature-based methods have only been evaluated on high quality laser scan data or synthetic data.

We propose a novel Surface Oriented Traverse (SOT) 3D feature to perform the task of robust instance detection in RGB-D image. Unlike traditional methods, we follow an interest points detection (i.e., contour points), feature extraction (i.e., SOT), feature points classification, saliency-map generation and object localization paradigm. First, we extract contour points from the depth image, which are sparse and often reveal the object's true boundary (Figure 1(b)). Second, we construct a SOT feature for each contour point by traversing and uniformly sampling along an oriented geodesic path on the object surface (Figure 1(a)). Third, we address the problem of instance detection in RGB-D as binary classification of all contour points into instance and non-instance ones (Figure 1(b)). Fourth, all classified contour points vote to generate an instance-specific saliency map (Figure 1(c)). Fifth, we finally localize the instance of interest by analyzing the saliency map. Our method performs well in the presence of partial occlusion, cluttering, and large pose variation, and takes noisy data from commodity 3D sensors, instead of accurate 3D models or high quality laser scan data, as input.

Our main contributions are: I. A novel Surface Oriented Traverse (SOT) 3D feature extracted from RGB-D image; II. A new instance detection framework by classifying and voting contour points extracted from depth images; III. Extensive evaluation of proposed method on a public dataset as well as a self-collected dataset. For the remainder of the paper, section II presents the relevant literature. Section III describes our instance detection algorithm in details. Section IV presents the experimental evaluation results while section V ends with a conclusion and future work.

II. Related Work

Object detection in RGB-D Data. For template-based methods, Hinterstoisser *et al.* [10] use an efficient representation of multimodal templates from both the RGB and depth images and scan a test RGB-D image with all templates in a sliding window fashion in real time. They further improve their method by a geometry and texture verification post-processing step [11]. In a recent work [22], the multimodal templates are discriminatively trained and a cascade scheme is proposed to speed up the detection process such that 10-30 objects can be detected simultaneously in real time. Though extremely fast, the holistic template-based methods usually fail in the case of partial occlusion.

For learning-based methods, Lai *et al.* [17] concatenate the Histogram of Gradients (HOG) features extracted from the RGB and depth images and train a linear Support Vector Machine (SVM) to detect the object. Pepik *et al.* [21] extend the Deformable Part Models (DPM) to include both estimate of viewpoint and 3D parts that are consistent across viewpoints. Kim *et al.* [14] find the optimal location of 3D objects by exploring the compatibility between segmentation hypotheses of the object in the image and the corresponding 3D map. Song and Xiao [25] train an Exemplar-SVM classifier for each rendered depth image of a 3D CAD model and detect it in the test image by sliding a detection window in 3D space. Meger *et al.* [19] locate objects in 3D that adapts visual appearance models using explicit visibility analysis. Bonde *et al.* [4] use soft label Random Forest to learn discriminative shape features of an object and emphasize features on the visible region to handle occlusion. Tejani *et al.* [30] employ a Latent-Class Hough Forests and handle clutter and occlusion during the inference process. Stiene *et al.* [29] detect objects in range images by classifying entire contours represented by the Eigen-Curvature Scale Space (Eigen-CSS).

While the problems of cluttering and partial occlusion have been addressed by the recent works of [10], [11], [22] and [4], [19], [25], [30] respectively, all of them assume an accurate 3D model as input, which is not easily accessible for common objects in daily life.

3D Features. Many 3D features, especially in the robotic community, have been developed recently. Stein and Medioni [28] propose SPLASH for efficient structural indexing of a 3D scene for object recognition. Johnson *et al.* [12] introduce the Spin Image for efficient object recognition. They recognize objects in cluttered scenes by matching geometrically consistent Spin Images. Frome *et al.* [8] extend the 2D Shape Context [3] and develop the 3D Shape Context (3DSC) feature for object recognition. They introduce a Representative Descriptor Method at the recognition stage. Knopp *et al.* [15] develop the SURF [2] feature to handle the 3D case and perform object recognition by a Hough transform procedure. Rusu *et al.* [23] introduce the Fast Point Feature Histogram (FPFH) 3D feature to perform the task of shape registration. The Signatures of Histograms of OrienTations (SHOT) feature [31] balances the trade-off between robustness and descriptiveness. The Rotational Projection Statistics (RoPS) feature [9] is obtained by rotationally projecting the neighboring points of a feature point onto 2D planes. It is used for object recognition again by checking geometric consistency among matchings. Steder *et al.* [27] construct the Normal Aligned Radial Feature (NARF) from contour points extracted from the range scan.

III. Surface Oriented Traverse (SOT) Feature

We first describe how to extract contour points in depth images (Section III-A). Then we propose our method to construct SOT for each contour point (Section III-B), followed by its parameters and characteristics (Section III-C) and a detailed comparison with a similar 3D feature (Section III-D). Finally we discuss how to classify contour points, how to vote for an instance-specific saliency map, and how to further localize the instance of interest in it (Section III-E).

A. Extract Contour Points from Depth Images

As shown in Figure 2, the contour points, or edges, extracted from intensity images by thresholding the gradient magnitude at each pixel, contain a lot of false boundary

pixels, i.e., pixels inside the object and from the background (Figure 2(a)). The contour points extracted from the depth image (Figure 2(b)), on the other hand, are much sparser and reveal objects' true boundaries.

Given a depth image $\mathcal{D}$ that maps a pixel $\mathbf{u} = (u, v)^T \in \mathbb{R}^2$ to its depth value $\mathcal{D}(\mathbf{u})$, the contour points are defined as points each of whose 8-connected neighborhood $\mathcal{N}_{\mathbf{u}}^8$ has at least one pixel $\mathbf{v}$ whose depth value $\mathcal{D}(\mathbf{v})$ is larger than $\mathcal{D}(\mathbf{u})$ by a threshold $D_{\mathcal{T}}$,

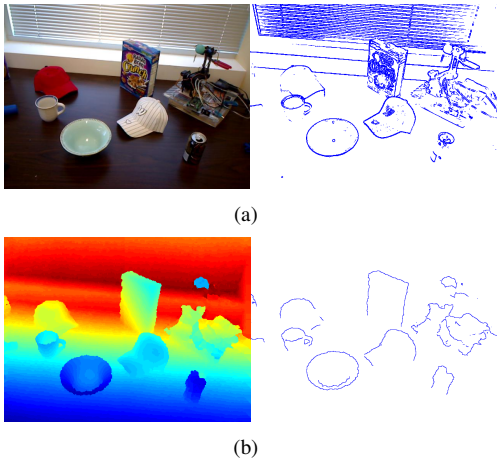
$$\mathcal{C} = \{\mathbf{u} \in \mathcal{U} | \exists \mathbf{v} \in \mathcal{N}_{\mathbf{u}}^8,\ \mathcal{D}(\mathbf{v}) - \mathcal{D}(\mathbf{u}) > D_{\mathcal{T}}\}, \quad (1)$$

where $\mathcal{U}$ represents the set of all possible 2D pixel locations, and $D_{\mathcal{T}}$ represents the largest gap allowed on a smooth surface. Throughout all our experiments (Section IV), we use a structured-light based Kinect sensor [1]. While the noise level increases quadratically with distance, we set the depth gap threshold $D_{\mathcal{T}}$ at depth z to be $15 \times q(z)$ where $q(z)$ is the quantization step at depth z [24]. For time-of-flight based 3D sensors with a well-known flying-pixel issue on object boundaries, an interesting discussion on contour points extraction can be found in [27].

B. The SOT Feature

To ease the understanding of the SOT feature construction process, we demonstrate how to extract SOT features on two objects, i.e. a cube (Figure 3(a)) and a ball (Figure 3(b)). For each extracted contour point $\mathbf{u}$, we construct its SOT feature by traversing and uniformly sampling along an oriented geodesic path on the object surface. There are two key components in the construction process, i.e., the *orientation of sampling* and how to sample in *a view-invariant way*. To be more specific, for a contour point $\mathbf{u}$, its SOT feature $\mathcal{F}_{SOT}(\mathbf{u})$ is constructed as follows (Figure 3):

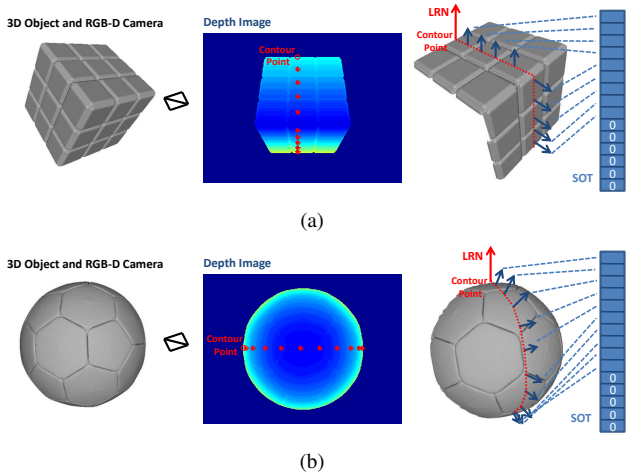
I. We decide the *orientation of sampling* $\vec{\mathbf{g}}_{\mathbf{u}} \in \mathbb{R}^2$ at contour point $\mathbf{u}$ by applying a Sobel filter on depth image $\mathcal{D}$, i.e., by calculating gradient of the depth image at $\mathbf{u}$. It is worth mention that since the object of interest always lies in the foreground, the calculated gradient $\vec{\mathbf{g}}_{\mathbf{u}}$ naturally points towards the object inside.



(a)

(b)

Fig. 2. (a) Color image and contour points extracted by thresholding the maximum gradient magnitude of the R,G,B channels at each pixel (b) Depth image and contour points extracted according to Equation 1.



(a)

(b)

Fig. 3. (a) 3D cube and SOT feature. (b) 3D ball and SOT feature. From left to right: 3D Object and the RGB-D camera; Depth image of the RGB-D camera and SOT feature constructed for one contour point (red circle), by uniformly sampling along the 2D contour normal line (red stars); The corresponding oriented geodesic path on object surface in 3D with blue arrows representing normals at sample locations. (Textures, though used in SOT, are not displayed here for better visualization.)

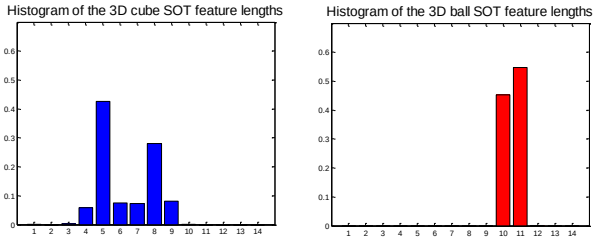
II. We sample in *a view-invariant way* by obtaining a 3D Local Reference Normal (LRN) $\hat{\mathbf{n}}_{LRN} \in \mathbb{R}^3$ at $\mathbf{u}$. LRN is calculated by using Principal Component Analysis (PCA) [13] on all points within radius δ_R of $\mathbf{u}$ in 3D.

III. Starting from $\mathbf{u}$, we traverse along an oriented line defined by $\vec{\mathbf{g}}_{\mathbf{u}}$ in the RGB-D image. We approximate the traversed geodesic distance by accumulating distances between consecutive depth pixels in 3D. Upon reaching pixel $\mathbf{v}$ after traversing a distance of δ_{step} on object surface, we calculate the *shape descent* with respect to LRN at $\mathbf{u}$ as the dot product $d_{\mathbf{u}\leftarrow\mathbf{v}} = (\vec{\mathbf{p}}_{\mathbf{v}} - \vec{\mathbf{p}}_{\mathbf{u}})^T \hat{\mathbf{n}}_{LRN}$ and *normal compatibility* with respect to LRN at $\mathbf{u}$ as dot product $c_{\mathbf{u}\leftarrow\mathbf{v}} = \hat{\mathbf{n}}_{\mathbf{v}}^T \hat{\mathbf{n}}_{LRN}$, where $\vec{\mathbf{p}}_{\mathbf{v}}$ indicates the corresponding 3D point of pixel $\mathbf{v}$. Besides geometry, we also incorporate texture information and store *running average*, starting from the last sample location, of R, G, B colors as $\bar{R}_{\mathbf{v}}$, $\bar{G}_{\mathbf{v}}$ and $\bar{B}_{\mathbf{v}}$ respectively. Running average is used to obtain a more robust estimate of texture. Finally at the current sample location $\mathbf{v}$, $d_{\mathbf{u}\leftarrow\mathbf{v}}$, $c_{\mathbf{u}\leftarrow\mathbf{v}}$, $\bar{R}_{\mathbf{v}}$, $\bar{G}_{\mathbf{v}}$ and $\bar{B}_{\mathbf{v}}$ are written into the SOT feature vector $\mathcal{F}_{SOT}(\mathbf{u})$.

IV. Repeat step III until total number of γ samples are obtained or until reaching a depth gap larger than $\mathcal{D}_T$, i.e., reaching the other side of the object boundary. In the latter case, the remaining parts of $\mathcal{F}_{SOT}(\mathbf{u})$ are filled with zeros.

C. Parameters and Characteristics of SOT

SOT feature has 4 parameters δ_R, $\mathcal{D}_T$, δ_{step} and γ. The first two parameters control the robustness of the LRN estimation and define the maximum depth gap that SOT can traverse through respectively. δ_{step} represents the sample distance on object surface and γ defines the length of SOT feature vector $\mathcal{F}_{SOT}(\mathbf{u})$. It is worth mention that $\delta_{step} \times \gamma$ defines the maximum length we can traverse along an object surface. Hence a small $\delta_{step} \times \gamma_{step}$ fails to completely



(a) (b)

Fig. 4. (a) Histogram of the lengths of SOT features extracted from multiple rendered depth images of the 3D cube at different view points. (b) Histogram of the lengths of SOT features extracted from multiple rendered depth images of the 3D ball at different view points.

traverse the object surface to capture enough discriminative information for classification, while a large $\delta_{step} \times \gamma_{step}$ leaves many zeros in the back of feature vector and is not computationally efficient. Parameter values are set in Section IV.

SOT feature has 3 main characteristics:

I. It is invariant to in-plane rotations since we are using image gradient at each contour point as the traverse orientation and all geometry information are sampled according to the view-invariant LRN. It is also invariant to scale changes, i.e., camera distance to the object center, since we are approximating the absolute geodesic distance while traversing. Its view-invariant property against changes in other view directions, however, depends on the shape of the underlying object. That being said, we find in our experiments that this shape related view dependency of SOT on each object is well generalized by the classification algorithm.

II. It is computationally efficient as essentially we are traversing along a 2D line and sampling a limited number of locations. To be more specific, the complexity to construct SOT for one contour point is $\mathcal{O}(N^{\frac{1}{2}})$, where N is the total number of image pixels, since a 2D line at most traverse $\mathcal{O}(N^{\frac{1}{2}})$ pixels and at each pixel only several arithmetic operators are involved. Hence complexity for constructing SOT features for all contour points in one image is $\mathcal{O}(N^{\frac{3}{2}})$. In practice, however, the complexity is $\mathcal{O}(N^{\frac{1}{2}}\gamma)$ considering that contour points from depth images are sparse and $\gamma << N^{\frac{1}{2}}$.

III. SOT encodes the object's global uniqueness. Besides the *shape descent*, *normal compatibility* and *running average of colors* along the oriented path, SOT also reveals the total length of the traversed path before reaching the object boundary on the other side, which is another intrinsic feature for different objects. As shown in Figure 4, when setting the same δ_{step}, most 3D ball's SOT features have either 9 or 10 steps, while most 3D cube's SOT features have lengths between 5 and 8.

D. Comparison with NARF

The NARF feature [27] is extracted from range images and also incorporates contour information. However, our SOT feature is substantially different and a detailed comparison is given which illustrates not only the differences between SOT and NARF but also the differences between SOT and other 3D features in general.

I. NARF features are computed on interest points, and while the extraction of interest points involves information of object boundaries, the extracted interest points do not necessarily lie on the contours depending on the support size δ selected. SOT features, on the contrary, are calculated on all contour points.

II. NARF feature, like other 3D features, is calculated based on the information collected in a local neighborhood, i.e., sphere, around the interest point. SOT feature, on the contrary, is obtained by sampling along a path starting from the contour point.

III. NARF feature uses depth information only. SOT feature, on the contrary, is multimodal and uses texture as well. While most commodity 3D sensors come with a calibrated RGB camera, a multimodal solution is proven to outperform a unimodal one in the context of instance detection [10], [16].

E. Classification, Saliency Map and Object Localization

Classification. A Random Forest is trained [5] to classify all constructed SOT features. We use a Random Forest with 30 decision trees, and employ Gini impurity as the splitting criterion. The tree depth is limited to 20 to prevent potential over-fitting. Training images are collected by taking images around instance of interest at different elevations (Section IV). At the testing stage, each SOT feature $\mathcal{F}_{SOT}(\mathbf{u})$ is passed into the Random Forest and the corresponding contour point $\mathbf{u}$ is assigned a posterior probability $p(\mathcal{I}|\mathbf{u})$ of belonging to instance $\mathcal{I}$ after classification.

Saliency Map. A classified contour point $\mathbf{u}$ votes for the instance saliency map by the following rules:

I. It votes the same saliency $\exp(p(\mathcal{I}|\mathbf{u}))\mathcal{D}(\mathbf{u})$ to all its local neighbors $\mathcal{N}_{\mathbf{u}}^{L}$, which is decided by the following rules II and III. $\mathcal{D}(\mathbf{u})$ is used to compensate for sparsity incurred by distance. A higher $p(\mathcal{I}|\mathbf{u})$ produces a more salient region $\mathcal{N}_{\mathbf{u}}^{L}$.

II. Depth value of pixels in $\mathcal{N}_{\mathbf{u}}^{L}$ are restricted to be within $\mathcal{D}_{max}$ and $\mathcal{D}_{min}$ which represent the maximum and minimum depth values during traverse respectively. Pixels with depth values outside the range are not regarded as belonging to object and do not get the vote.

III. Pixels in $\mathcal{N}_{\mathbf{u}}^{L}$ are restricted to be within a radius of $\|\mathbf{u}-\mathbf{m}\|_2$ where $\mathbf{m} \in \mathcal{R}^2$ is the 2D middle sample location during traverse. Pixels outside the radius are not regarded as belonging to object and do not get the vote.

Finally, for the saliency map $\mathcal{S}$, its value at pixel $\mathbf{v}$ is calculated as the sum of voting from all contour points, i.e., $\mathcal{S}(\mathbf{v}) = \sum_{\mathbf{u}\in\mathcal{C}} \mathbb{1}(\mathbf{v} \in \mathcal{N}_{\mathbf{u}}^{L}) \exp(p(\mathcal{I}|\mathbf{u}))\mathcal{D}(\mathbf{u})$ where $\mathbb{1}()$ is the indicator function. Examples of generated saliency maps are shown in Figure 1 and Figure 8.

Object Localization. After reconstructing the saliency map, we localize the instance of interest from it by iteratively selecting and removing high salient regions. To know the maximum size that the object may occupy in the RGB-D image in advance, we store, for each pixel $\mathbf{v}$, half of the

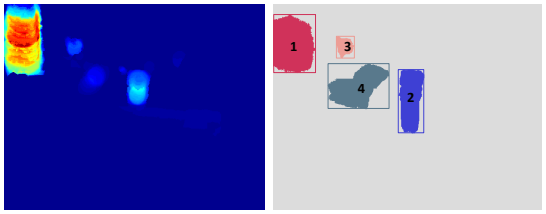


Fig. 5. Left: Saliency map; Right: 4 segmentations of the instance of interest from the saliency map ranked according to the saliency value.

maximum traverse distance of all SOT features that vote for $\mathbf{v}$, i.e., $\mathcal{D}(\mathbf{v}) = \max_{\mathbf{u} \in \mathcal{C}} \mathbb{1}(\mathbf{v} \in \mathcal{N}^L_{\mathbf{u}}) \frac{\mathcal{L}_{SOT}(\mathbf{u})}{2}$, where $\mathcal{L}_{SOT}(\mathbf{u})$ represents the geodesic length of SOT feature $\mathcal{F}_{SOT}(\mathbf{u})$. $\mathcal{D}$ indicates, starting for each pixel, the approximate maximum geodesic distance to safely traverse and stay on the object surface. Our object localization algorithm works as follows:

I: Retrieve pixel $\mathbf{z}$ of the maximum saliency, i.e., $\mathbf{z} = \arg\max_{\mathbf{v}} \mathcal{S}(\mathbf{v})$.

II: Set $\mathbf{z}$ as the root node of a tree and perform Breadth First Search (BFS) starting from $\mathbf{z}$ to propagate segmentation to the entire tree. The propagation stops at a pixel if a depth gap larger than $D_{\mathcal{T}}$ is reached or its geodesic distance to the root node is larger than $\mathcal{D}(\mathbf{z})$, i.e., when we are no longer on the object.

III: All pixels of the tree rooted at $\mathbf{z}$ is one localization, i.e., segmentation, of the instance. We remove the tree from $\mathcal{S}$ by setting saliency of all tree pixels as 0, and repeat step I until $\mathcal{S}$ is all zero.

As shown in Figure 5, our localization method provides several segmentations of the instance from the saliency map, and similar to [14], our segmentation of the object is in the 3D scene, instead of simply a 2D bounding box. The saliency score of each segmentation is assigned as the maximum saliency value within it. Example localization results are shown in Figure 8 and Figure 10. For evaluation purposes, in order to be consistent with other methods, we still assign a bounding box to each segmentation. Bounding boxes of small areas or low saliency scores are pruned. And a final Non-Maximum Suppression (NMS) is performed on all bounding boxes.

IV. Experiments

For experiments, we evaluate our instance detection method on a public RGB-D Object Dataset [16] (Section IV-B) and our proposed FindMe RGB-D Dataset (Section IV-C). Both datasets are collected with a commodity structured-light 3D sensor, i.e., Kinect [1], and no accurate 3D models of the objects are provided. We compare our SOT-RF (i.e., Random Forest on SOT) method with two popular instance detection algorithms, i.e., a template-based method LINEMOD [10] and a learning-based method HOG-SVM [16], [17] that employs linear SVM [6] on concatenated RGB and Depth HOG features [7].

A. Data Preprocessing

The depth image provided by the Kinect sensor contains noisy data with holes. We preprocess the Kinect data in three ways: 1) Apply a recursive median filter on depth image to fill holes as proposed in [16]; 2) Smooth the depth image and extract 3D surface normal per pixel as proposed in [10]; 3) Smooth the estimated 2D contour normals, i.e., sampling orientations, with a Gaussian filter to obtain more robust estimation at each contour point.

B. RGB-D Object Dataset

The RGB-D Object Dataset [16] contains 8 individual scenes for evaluating instance detection algorithms, where each scene is a single video sequence consisting of multiple RGB-D frames. These 8 scenes contain approximately 20 objects belonging to the larger RGB-D Object Dataset. To get a more robust evaluation, we use the 10 objects (Figure 6) where each of them appears in at least 2 scenes. Each object, in the RGB-D Object Dataset, is represented by 3 sequences of RGB-D frames taken around the object at different heights.

For SOT-RF and HOG-SVM, positive training samples are from the instance's RGB-D images while negative training samples come from other objects' RGB-D images as well as those randomly sampled from the background scenes. For LINEMOD, to generate multiple templates for each RGB-D image, we construct a meshed points cloud from each RGB-D image and render it in different camera pose combinations $[-15^o, 0^o, 15^o] \times [0.9m, 1.0m, 1.15m, 1.30m, 1.50m]$, with the former representing variation of in-plane rotation and the latter covering various distances between the camera and the instance. Thus, we generate 15 templates per RGB-D image, and by using approximately 300 RGB-D images for each object, we obtain a total of roughly 5000 templates for each object.

Parameters of SOT-RF are set as below: $\delta_R = 10mm$, $\delta_{step} = 5mm$ for small objects (soda can, flashlight and coffee mug), $\delta_{step} = 15mm$ for large objects (cereal box, cap and bowl), $\gamma = 20$ for all objects. Parameters are not specifically tuned and work in a wide range.

For each object, we detect it in those test scenes it appears. For each RGB-D image, we do not assume that the instance

Fig. 6. 10 objects from the RGB-D Object Dataset used for evaluation. Top row: cereal box 1, cereal box 4, soda can 3, soda can 6, bowl 2; Bottom row: cap 1, cap 4, flashlight 3, flashlight 5, coffee mug 1.

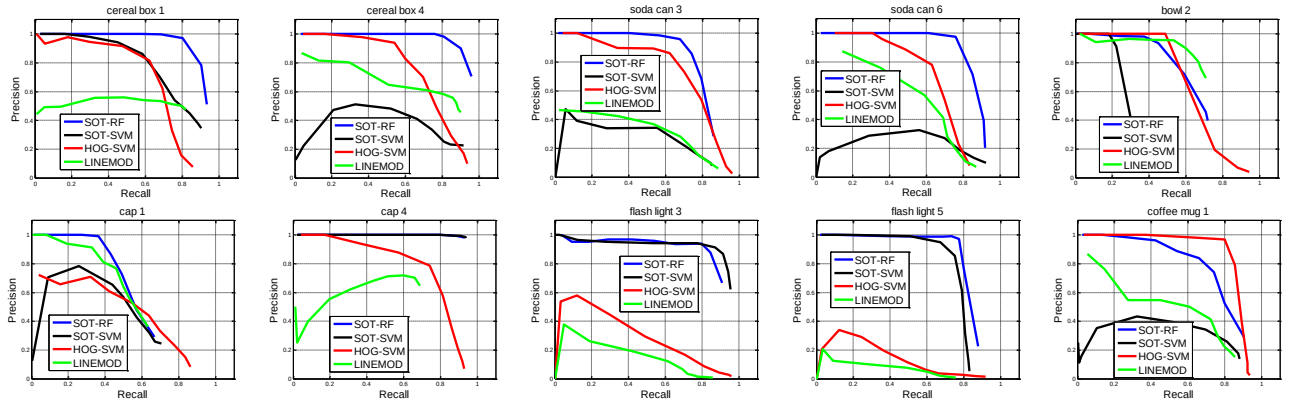
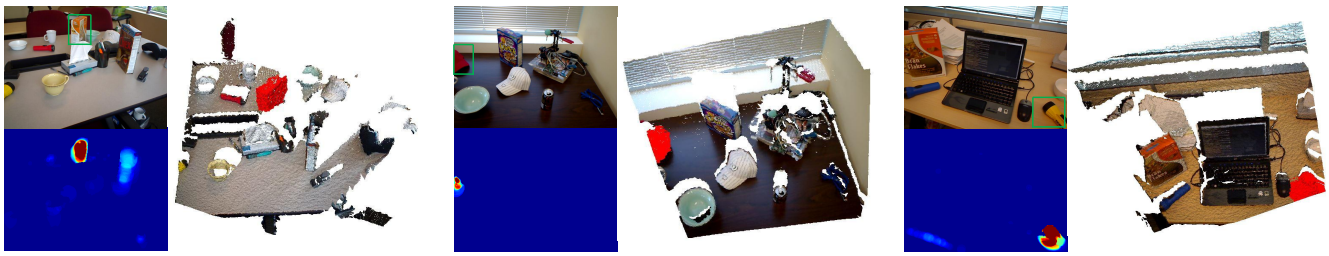


Fig. 7. Precision-recall curves comparing performance of SOT-SVM+RF, SOT-SVM, HOG-SVM and LINEMOD on 10 objects in the RGB-D Object Dataset.



(a) (b) (c)

Fig. 8. (a) Cereal box 1 detection in 3D scene. (b) Cap 4 detection in 3D scene. (c) Flashlight 5 detection in 3D scene. For each image, top left displays the RGB image as well as the bounding box in green, bottom left shows the corresponding saliency map and right visualizes the 3D object detection result (in red). HOG and LINEMOD fail to localize these objects due to severe partial-occlusions.

is unique and we keep all bounding boxes instead of only the one with the highest score. To demonstrate the necessity of using a more powerful, yet computationally expensive, Random Forest as our learning algorithm, we compare with a baseline method SOT-SVM which employs a linear SVM to classify SOT features.

The precision-recall curves of different methods on the 10 objects are shown in Figure 7. SOT-RF clearly outperforms SOT-SVM which indicates insufficiency in generalization power of linear SVM. SOT-RF outperforms HOG-SVM and LINEMOD on 8 out of 10 objects. For objects (flashlight), when SOT-RF outperforms HOG-SVM and LINEMOD by a large margin, it is primarily due to the fact that HOG-SVM fails to generalize a reasonable model (the flashlight looks substantially different in each view especially along the azimuth direction when placed on table) while for LINEMOD, many of its templates are easily confused with edges from the background. For objects (cereal box, cap and soda can), where SOT-RF outperforms HOG-SVM or LINEMOD by a small margin, it is mainly because that HOG-SVM and LINEMOD are holistic methods and fail to deal with partial occlusions, while SOT-RF is feature-based and performs well in those scenarios (Figure 8). SOT-RF method performs slightly worse than HOG-SVM on concave shape objects (bowl and coffee mug) mainly because that the SOT feature fails to traverse through gap on object surface incurred by its concavity and does not capture enough global discriminative information.

C. FindMe RGB-D Dataset

To fully test the performance of our SOT-RF method in the presence of cluttering, partial occlusion and large pose variation, we collect the FindMe RGB-D Dataset of 10 objects. Data is collected similarly as the RGB-D Object Dataset. We select 5 objects (cereal box, football, food bag, ninja toy and tea can) and detect them in 4 different scenes. **Scene 1: Clean.** We spread 5 objects over the table to prevent partial-occlusions and collect data in a controlled camera motion (Figure 10(a)). **Scene 2: Large pose variation:** We collect the same data of 5 objects with a freely moving camera, i.e. large in plane rotations and varying distance to the table. **Scene 3: Partial-occlusion:** We put the other 5 objects on the same table to create a cluttered scene where objects might be partially occluded, and collect data in a controlled camera motion. **Scene 4: Partial-occlusion and large pose variation:** We collect the same data of cluttered scene with a freely moving camera (Figure 10(b)). For each scenario, approximately 500 frames are collected and evaluated.

For a fair comparison with LINEMOD in the scenes with wild camera motion, we render each RGB-D image in 60 different combinations $[0^o : 30^o : 330^o] \times [0.9m, 1.0m, 1.15m, 1.30m, 1.50m]$, i.e. we fully cover the in-plane rotation variations. We collect approximately 20,000

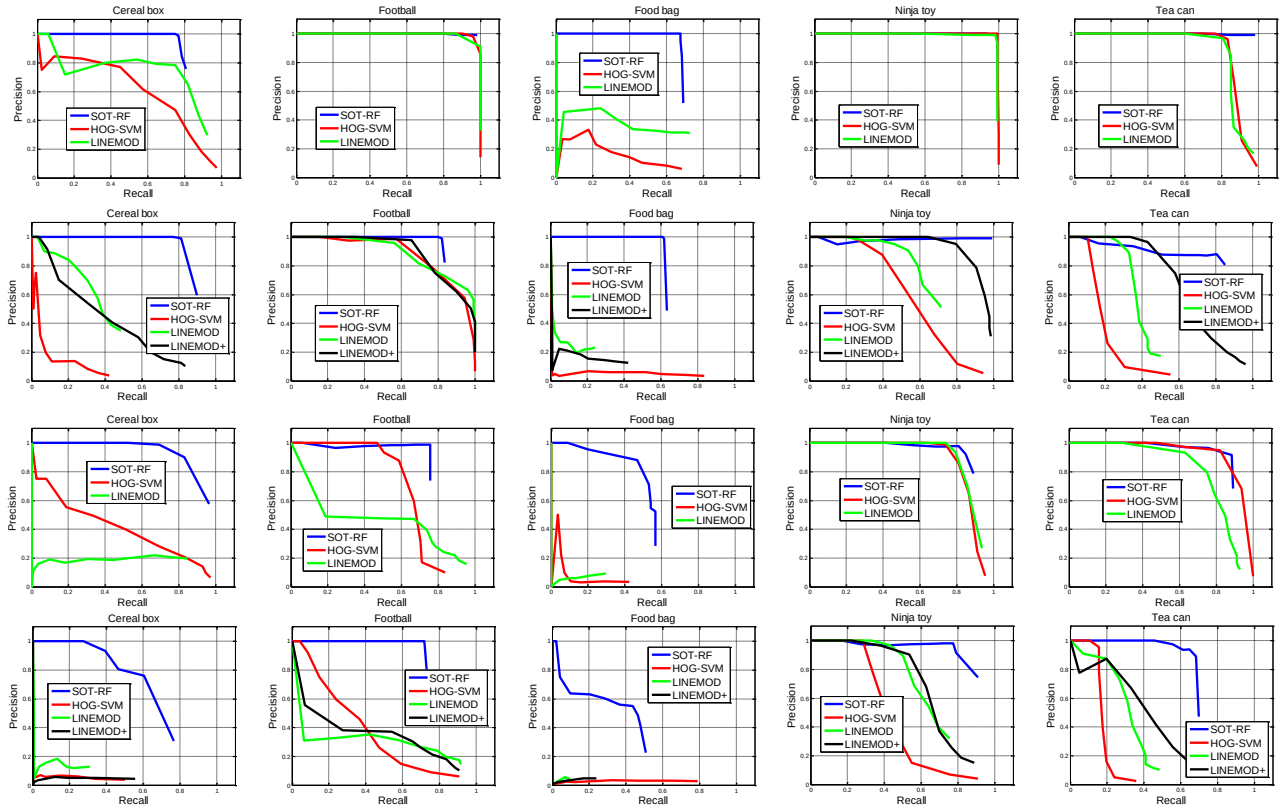
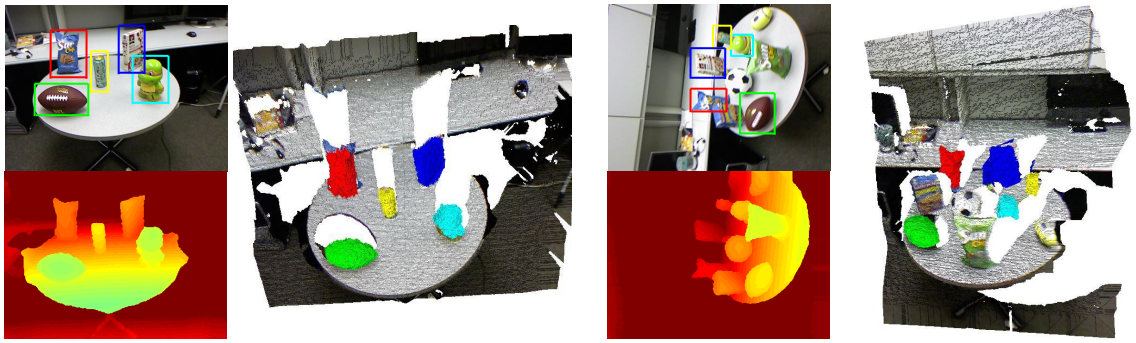


Fig. 9. From top row to bottom row: precision-recall curves comparing SOT-SVM+RF, HOG-SVM, LINEMOD and LINEMOD+ on 5 objects in scenarios 1 to 4 of FindMe RGB-D Dataset.



(a) (b)

Fig. 10. (a) Example detection result of scenario 1. (b) Example detection result of scenario 4. For each example, top left displays the RGB image with the detected 2D bounding boxes, bottom left shows the depth image and right visualizes the 3D scene as well as 3D detection results in different colors.

templates for each object and denote this method as LINEMOD+. LINEMOD+ is evaluated in scene 2 and scene 4 to overcome all different object poses seen by a freely moving camera. For SOT-RF, we set $\delta_{step} = 12mm$ and $\gamma = 20$ for all objects while other parameters remain the same.

Figure 9 shows the comparative results. For scene 1, all 3 methods exhibit similar high performance on food bag, ninja toy and tea can, as there is no occlusion and the camera moves in a controlled manner. HOG-SVM and LINEMOD, however, fail to detect thin side of the cereal box as well as non-rigidity of the food bag even in a clean set-up. Though performance of HOG-SVM and LINEMOD remains competitive on specific objects when either free camera motion or partial-occlusions are added, none of them reaches the same performance level of our SOT-RF method when partial-occlusions and free camera motion are combined as in scene 4. Also adding more templates to LINEMOD does not guarantee an increase in performance, as incorporating more templates causes a higher chance of confusion with either background or other objects. To sum up, our feature-based SOT-RF method can correctly localize all 5 objects in a cluttered scene even in a novel view (Figure 10(b)).

On average our current CPU implementation preprocess

data in $135ms$, extract contour points and construct SOT features in $12ms$, perform binary classification in $110ms$, generate saliency map and localize the object in $37ms$. This leads to a total time of $135ms + 12ms + 5 \times (110ms + 37ms) = 882ms$ to detect 5 objects in a single RGB-D image. Since each SOT feature is constructed, classified and used for voting independently from each other, we believe that a GPU implementation will easily run in real time.

D. Discussion and Limitations

Specifically, our proposed SOT feature can be directly employed on service robots, designed for visually impaired and possibly mounted on a helmet, to help them localize common items of interest in their surroundings. In this scenario, near-real-time 3D object instance detection in cluttered, noisy RGB-D images, captured via head mounted sensors in a free egocentric motion, is essential to be of utility to visually impaired people. One future application of SOT features thus includes integration into a wearable visual aid system that combines scene/context recognition, robust real-time 3D object detection/recognition, and auditory feedback cues, to help guide visually impaired users to desired items around them on cluttered office desks or pantry shelves, similar to the complex evaluation tests presented here with the RGB-D Object Dataset and the proposed FindMe RGB-D Dataset.

As a limitation, currently our SOT feature works well on objects of well shaped boundaries and distinguishable textures, while applies poorly to thin objects or objects with deep concavity, since SOT fails to traverse through the object surface to collect enough discriminative features. We believe that, however, our pixel-wise method is complementary to the traditional holistic methods and combining both will lead to a better detection performance.

V. Conclusion

We have proposed a novel Surface Oriented Traverse (SOT) 3D feature and a solid framework to address robust instance detection in RGB-D. Our method takes noisy data from commodity sensors as input and performs well in the presence of cluttering, partial occlusion, and large pose variation.

References

[1] "Microsoft kinect for windows," http://www.xbox.com/en-US/kinect.

[2] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *ECCV*. Springer, 2006, pp. 404–417.

[3] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 4, pp. 509–522, 2002.

[4] U. Bonde, V. Badrinarayanan, and R. Cipolla, "Robust instance recognition in presence of occlusion and clutter," in *ECCV*. Springer, 2014, pp. 520–535.

[5] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[6] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, vol. 1. IEEE, 2005, pp. 886–893.

[8] A. Frome, D. Huber, R. Kolluri, T. Bülow, and J. Malik, "Recognizing objects in range data using regional point descriptors," in *ECCV*. Springer, 2004, pp. 224–237.

[9] Y. Guo, F. Sohel, M. Bennamoun, M. Lu, and J. Wan, "Rotational projection statistics for 3d local surface description and object recognition," *International journal of computer vision*, vol. 105, no. 1, pp. 63–86, 2013.

[10] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit, "Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes," in *ICCV*. IEEE, 2011, pp. 858–865.

[11] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," in *ACCV 2013*. Springer, 2013, pp. 548–562.

[12] A. E. Johnson and M. Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 21, no. 5, pp. 433–449, 1999.

[13] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2005.

[14] B.-s. Kim, S. Xu, and S. Savarese, "Accurate localization of 3d objects from rgb-d data using segmentation hypotheses," in *CVPR*. IEEE, 2013, pp. 3182–3189.

[15] J. Knopp, M. Prasad, G. Willems, R. Timofte, and L. Van Gool, "Hough transform and 3d surf for robust three dimensional classification," in *ECCV*. Springer, 2010, pp. 589–602.

[16] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view rgb-d object dataset," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1817–1824.

[17] K. Lai, L. Bo, X. Ren, and D. Fox, "Detection-based object labeling in 3d scenes," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 1330–1337.

[18] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, vol. 2. IEEE, 1999, pp. 1150–1157.

[19] D. Meger, C. Wojek, J. J. Little, and B. Schiele, "Explicit occlusion reasoning for 3d object detection." in *BMVC*. Citeseer, 2011, pp. 1–11.

[20] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*. IEEE, 2011, pp. 127–136.

[21] B. Pepik, M. Stark, P. Gehler, and B. Schiele, "Teaching 3d geometry to deformable part models," in *CVPR*. IEEE, 2012, pp. 3362–3369.

[22] R. Rios-Cabrera and T. Tuytelaars, "Discriminatively trained templates for 3d object detection: A real time scalable approach," in *ICCV*. IEEE, 2013, pp. 2048–2055.

[23] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Robotics and Automation (ICRA), 2009 IEEE International Conference on*. IEEE, 2009, pp. 3212–3217.

[24] J. Smisek, M. Jancosek, and T. Pajdla, "3d with kinect," in *Consumer Depth Cameras for Computer Vision*. Springer, 2013, pp. 3–25.

[25] S. Song and J. Xiao, "Sliding shapes for 3d object detection in depth images," in *ECCV*. Springer, 2014, pp. 634–651.

[26] B. Steder, G. Grisetti, M. Van Loock, and W. Burgard, "Robust online model-based object detection from range images," in *Intelligent Robots and Systems (IROS), 2009 IEEE/RSJ International Conference on*. IEEE, 2009, pp. 4739–4744.

[27] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard, "Point feature extraction on 3d range scans taking into account object boundaries," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 2601–2608.

[28] F. Stein and G. Medioni, "Structural indexing: Efficient 3-d object recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 125–145, 1992.

[29] S. Stiene, K. Lingemann, A. Nuchter, and J. Hertzberg, "Contour-based object detection in range images," in *3D Data Processing, Visualization, and Transmission, Third International Symposium on*. IEEE, 2006, pp. 168–175.

[30] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim, "Latent-class hough forests for 3d object detection and pose estimation," in *ECCV*. Springer, 2014, pp. 462–477.

[31] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *ECCV*. Springer, 2010, pp. 356–369.