Model-Free Segmentation and Grasp Selection of Unknown Stacked Objects*

Umar Asif, Mohammed Bennamoun, and Ferdous Sohel

School of Computer Science & Software Engineering,
The University of Western Australia, Crawley, Perth, WA, Australia
umar.asif@research.uwa.edu.au,
{mohammed.bennamoun,ferdous.sohel}@uwa.edu.au

Abstract. We present a novel grasping approach for unknown stacked objects using RGB-D images of highly complex real-world scenes. Specifically, we propose a novel 3D segmentation algorithm to generate an efficient representation of the scene into segmented surfaces (known as surfels) and objects. Based on this representation, we next propose a novel grasp selection algorithm which generates potential grasp hypotheses and automatically selects the most appropriate grasp without requiring any prior information of the objects or the scene. We tested our algorithms in real-world scenarios using live video streams from Kinect and publicly available RGB-D object datasets. Our experimental results show that both our proposed segmentation and grasp selection algorithms consistently perform superior compared to the state-of-the-art methods.

Keywords: 3D segmentation, grasp selection.

1 Introduction

Grasping unknown objects in real-world environments is still a challenging task for visual perception and autonomous robot manipulation [3]. There are two main problems in a general grasping pipeline. **First**, an accurate segmentation of an object of interest is required to separate it from the environment. This is an intrinsically challenging problem, especially when the scene contains a large variety (textured, non-textured, planar and non-planar) of stacked objects with no information about their sizes, shapes or, visual appearance (e.g., color, or texture). The difficulty is compounded with the additional challenges of real-world environments (e.g., variable lighting conditions, shadows, clutter, and inherent sensor noise). Many state-of-the-art approaches handle this task either through recognition-based methods (e.g., [11,6,18,34,30,2]) or learning-based methods (e.g., [27,20]). While the former methods restrict the system to a fixed number of objects (whose models were previously stored or learned), the latter are also not suitable due to their high computational runtime when dealing with unknown objects in unknown environments (e.g., home or workplace).

* Electronic supplementary material - Supplementary material is available in the online version of this chapter at http://dx.doi.org/10.1007/978-3-319-10602-1_43. Videos can also be accessed at http://www.springerimages.com/videos/978-3-319-10601-4

D. Fleet et al. (Eds.): ECCV 2014, Part V, LNCS 8693, pp. 659–674, 2014.

Second, the selection of the most appropriate grasp from nearly an infinite number of possible grasps is a great challenge in the case of unknown stacked objects. Many approaches (e.g., [8,29]) use 2D or 3D object models to determine feasible grasps. Other approaches (e.g., [12,19,5]) assume that the objects to be grasped belong to a particular set of shape compositions and therefore reduce the number of possible grasps by fitting shape primitives (e.g., boxes, cylinders, or superquadrics). However, these shape abstractions are not accurate in the case of stacked and occluded objects. Learning-based approaches (e.g., [17,32,24,3]) handle the case of unknown objects using 2D and 3D features. However, all these approaches consider simple situations where the objects are isolated on a table. In the case of stacked and occluded objects, it is a very challenging task to generate the most appropriate grasp from a noisy partial view of the object (particularly from a single camera viewpoint).

We present a purely vision-based approach which addresses both the aforementioned problems in a model-free manner. **To handle the first problem**, we present a novel 3D segmentation algorithm (Sec. 3.1-3.2) which segments the scene into surfels and object hypotheses. **For the second problem**, we present a novel algorithm (Sec. 3.3) which generates potential grasps (specifically two-finger pinch-grasps) using surfels and automatically selects the most appropriate grasp for the objects in the scene. We tested our algorithms using live video streams from Microsoft Kinect and challenging object datasets, where we demonstrate that our algorithms improve both the segmentation and grasp selection performance for unknown stacked and occluded objects (Sec. 4). In summary, the contributions of this paper are:

1. Our proposed 3D segmentation algorithm segments unknown stacked and occluded objects compared to the methods in [24,28], which only segment objects isolated from each other. Our algorithm is completely **model-free** compared to the methods in [27,20] which require learning and parameter tuning on specific datasets. Hence, our algorithm is capable of separating a large variety of objects without any prior information about the objects or the scene (e.g., prior information about the table plane, or background). Our segmentation results are superior compared to the state-of-the-art methods (Sec. 4.2)
2. Our proposed grasp selection algorithm generates potential grasps and automatically selects the most appropriate grasp for unknown stacked and occluded objects. To the best of our knowledge, this is the first work to deal with unknown stacked objects in a complete **model-free** way. Our grasp selection results outperform the state-of-the-art grasp-selection methods (Sec. 4.3).
3. We evaluate the performance of our algorithms in real-world environments using our in-house mobile robot with a 7-DOF arm to grasp unknown stacked objects (see video in the supplementary material).

2 Related Work

Vision-based object grasping approaches can be divided into three main categories [3], grasping of known, familiar, and unknown objects. **For grasping**

known objects, active-exploration-based methods (e.g., [25]), or learning-based techniques (e.g., [31,22]) assume that a detailed 2D or 3D model of the object is available which is matched to the current scene for segmentation and pose estimation. Based on the model and the estimated pose, a large number of grasp hypotheses are generated and their quality is evaluated to compute the most appropriate grasp. Methods (e.g., [4]) use analytical approaches (by modeling the interaction between a gripper and an object using tactile sensing [23]) to generate force-closure grasps and rank them by optimizing the parameters of a dexterous hand [35]. Others use learning through demonstration approaches (e.g., [9]) to efficiently associate grasps to known objects. Recently, methods (e.g., [12,19]) evaluate the shape of the object by fitting shape primitives (e.g., boxes, cylinders, cones, and superquadrics) thereby reducing the number of potential grasps. However, all these methods require a prior knowledge about the objects, which restricts their application in real-world environments containing unknown objects.

For grasping familiar objects, methods (e.g.,[32,22,7]) learn relationships between visual features and grasp quality on a set of training objects. This knowledge is then used to grasp resembling objects. For instance, in [32], grasp hypotheses were learned based on a set of local 2D image features (edges, textures and color) using synthetic data to grasp novel objects. The method of [22] used a support vector machine (SVM) to predict the grasp quality based on the hand configuration and the parameters of the superquadric representation of the objects. In the work of [7], grasp selection was learned on a set of simple geometrical shapes and applied to grasp novel objects. However, most of these methods were trained using synthetic models and implemented in simulation. **Grasping unknown objects** is acknowledged to be a difficult problem particularly in the presence of a large variety of objects (i.e., with different sizes, shapes, textures, and layouts) and scenes (i.e., with clutter, indistinguishable background, and different lighting conditions). In this context, [21] used 2D contours of objects to approximate their centers of mass for grasp selection. The method in [28] used a selection algorithm to determine grasping points on the top-surface of an object. The method in [24] used a bottom-up segmentation algorithm and an SVM to learn the grasp selection of unknown objects. Recently, an early cognitive system presented in [14] was applied to the problem of grasping unknown objects based on edge and texture features. However, all of these methods consider the simple scenario of objects isolated from each other, which undermines the segmentation problem. This is not true when dealing with unknown real-world situations (e.g., Fig. 1B). Considering the important requirements for the next generation of service robots [14]), the grasping capabilities need to be implemented and evaluated in complex situations (i.e., scenes with unknown stacked and occluded objects).

In this paper, we present a purely vision-based grasping approach which has the following advantages, compared to the other methods: **First**, it is a fully model-free approach (i.e., no prior information about the objects or the scene is required), in contrast to the methods (e.g., [3,17,16,24,14]) which learn grasp

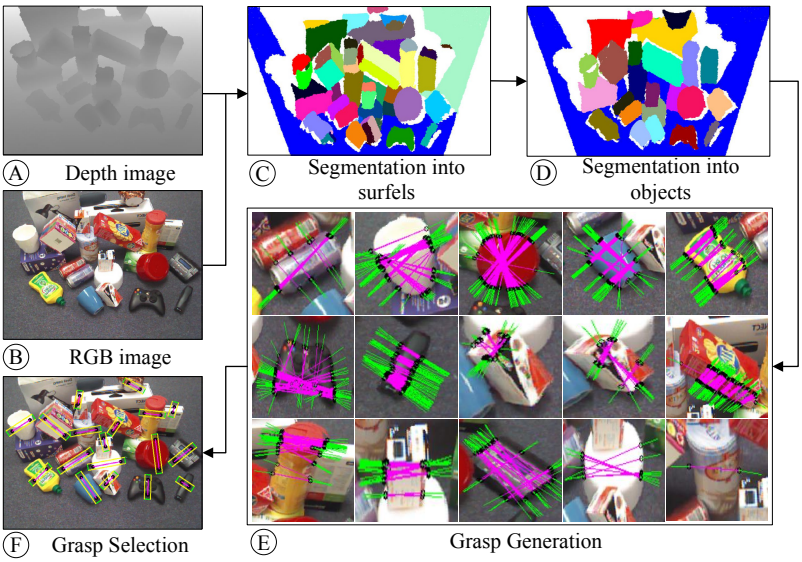

Fig. 1. Overview of our proposed grasping approach. First, we take a depth image (A) and a color image (B) as input and generate two segmentations: mid-level surfels (C), and high-level object hypotheses (D). Next, we use surfels to generate grasp hypotheses (E) and subsequently select the most appropriate grasp (F).

selection on training data using specific datasets. This allows our approach to generalize to different real-world environments (e.g., kitchen, or a workplace). **Second**, our approach successfully handles a large variety of unknown objects in unknown layouts compared to the methods (e.g., [17,16,24,28,14]) which considered only simple situations.

3 Proposed Grasping Approach

As illustrated in Fig. 1, **first**, we produce a 3D segmentation of the scene into mid-level surfaces (known as surfels), and high-level object hypotheses. **Second**, we use these segmentations to generate potential grasp hypotheses. Our overall approach addresses two key problems: **i)** accurate segmentation of unknown stacked and occluded objects and **ii)** selection of the most appropriate grasp in a model-free manner. **For the first problem**, we propose a new 3D segmentation algorithm which produces a layered representation of the scene in terms of mid-level and high-level information (i.e., surfels, and objects respectively). On the mid-level, our algorithm (see Sec. 3.1) follows an optimization procedure using simulated annealing to partition a given point cloud into surfels. On the high-level, our algorithm (see Sec. 3.2) combines perceptually similar surfels into object hypotheses using graph-cuts. **For the second problem**, we propose a novel algorithm (see Sec. 3.3) which generates potential grasp hypotheses and

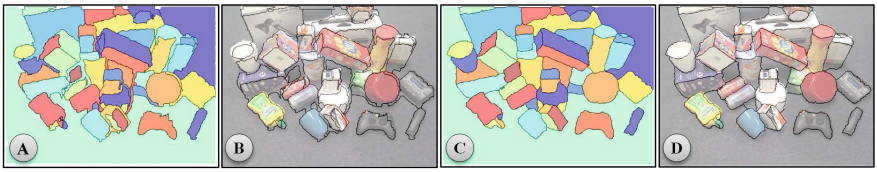

Fig. 2. Segmentation into surfels and their refinement. A-B: initial partitioning used for initialization, C-D: partitioning after 5 iterations of simulated annealing.

subsequently evaluates them to select the most appropriate grasp in a model-free manner.

3.1 Segmentation into Surfels

As illustrated in Fig. 2, our segmentation proceeds in two stages: **i)** initial segmentation (see Fig. 2A) and **ii)** its iterative refinement (see Fig. 2C). **In the first stage**, our algorithm produces an initial partitioning of the point cloud in terms of surfels. **In the second stage**, the algorithm iteratively refines these surfels (by exchanging boundary points between neighbors) so that their boundaries accurately conform to the physical object boundaries in the scene.

Initial Segmentation. The algorithm starts with a clustering technique in which structurally homogeneous points (i.e., those which have close 3D proximities, and similar local- and global-orientations of their surface normals) are grouped into distinct clusters. Our clustering algorithm is similar to the two-pass binary image labeling technique described in [33]. It has been, however, modified to label point cloud using our distance metric D_c . We define a comparator function $\mathbb{C}$, which compares a point i with its neighboring point j as given below:

$$\mathbb{C}(i,j) = \begin{cases} true & if(D_c < \delta_{D_c}). \\ false & otherwise \end{cases} \tag{1}$$

If $\mathbb{C}(i,j) = true$, then a common label is assigned to both the points i and j, otherwise a new label is created by incrementing the largest assigned label by one. Our distance metric D_c is a linear combination of three terms: 3D proximity, co-planarity, and global-orientation disparity. D_c can be written as:

$$D_c = \sqrt{||\mathbf{p}(i) - \mathbf{p}(j)||^2 + ||\beta_n(i) - \beta_n(j)||^2 + ||\gamma_n(i) - \gamma_n(j)||^2}, \tag{2}$$

where, $||.||$ represents the normalization within the local neighborhood. The terms $\mathbf{p}$, β_n, and γ_n represent the 3D coordinates, surface normal local orientation, and surface normal global-orientation of a query point respectively. For a point i, the orientation $\beta_n(i)$ is determined by the average dot product between its surface normal $\boldsymbol{n}_i$ and the normals of all the points in a local neighborhood $\boldsymbol{\Omega}$ of radius σ as:

$$\beta_n(i) = \frac{\sum_{j=1}^{size(\boldsymbol{\Omega})} acos(dot(\boldsymbol{n}_i, \boldsymbol{n}_j))}{size(\boldsymbol{\Omega})}. \tag{3}$$

The global-orientation $\gamma_n(i)$ is determined by the $atan2$ angle between the surface normal $\boldsymbol{n}_i$ and a vector $\boldsymbol{v}_c$ (drawn from the camera reference frame to the corresponding point i in the point cloud) as:

$$\gamma_n(i) = atan2(||cross(\boldsymbol{n}_i, \boldsymbol{v}_c)||, dot(\boldsymbol{n}_i, \boldsymbol{v}_c)). \tag{4}$$

Our algorithm proceeds as follows: the first point of the point cloud is assigned a unique label. For the remaining points, each unlabeled point i is compared to its left, left-top, and top neighbors using Eq. 1. If multiple neighbors with different labels satisfy the criterion in Eq. 1, the label of one of the neighbors is assigned to the current point i and equivalences are set for all of these neighbors. Otherwise, a new label is assigned to the point i by incrementing the largest assigned label by one. This procedure is iteratively repeated until all points are labelled. In the second pass, the labelled image is scanned again, and all equivalent regions are assigned the same region label using the union-find algorithm as in [33]. This produces a partitioning into regions whose boundaries are subsequently refined using an optimization technique as described in the following.

Refinement. We introduce an objective function, which when optimized, enforces structural homogeneity (i.e., similar normal orientations) of the points within each surfel. The optimization is based on a simulated annealing algorithm which proceeds as follows: The algorithm uses the initial segmentation as an initial solution and iteratively updates the solution by exchanging boundary points between neighboring surfels to generate a new partitioning. The algorithm terminates if the objective function does not increase or when the maximum number of iterations is reached (if the algorithm is allowed to run for a fixed number of iterations). In order to converge to a solution close to the global optimum, it is important to start with a good partition. We use our initial segmentation as a first rough partitioning for the simulated annealing algorithm. In our experiments, we found that there are several advantages of this initialization which include: **i)** the initial segmentation is automatic (i.e., our clustering algorithm automatically segments the point cloud into structurally distinct clusters). **ii)** Our initial segmentation is perceptually accurate because structurally homogeneous points are grouped into distinct clusters. Since the initialization is close to the optimal solution, this speeds up the convergence of the simulated annealing algorithm. Let K be the number of surfels we obtain from the initial segmentation. We define $S = \{s_1, s_2, \ldots, s_K\}$ to describe the initial partitioning where, each surfel s_k is represented by a cluster of 3D points in the point cloud as shown in Fig. 1C.

Objective Function Our objective function is defined to evaluate the homogeneity as a distribution of local orientations (ϕ, ψ, and α) of the surface normals of the points within each surfel s_k. The orientations are measured in the same manner as in [30]. Mathematically, the ϕ, ψ, and α values of a point $i \in s_k$ are computed

with respect to its surfel's centroid s_{k_c} as:

$$\phi(i) = acos(dot(\mathbf{n}_i, \boldsymbol{v}_n)), \psi(i) = acos(dot(\mathbf{n}_{s_k}, \boldsymbol{v}_n)), \alpha(i) = atan2(\rho, \tau), \\ \rho = dot(cross(\mathbf{n}_{s_k}, cross(\boldsymbol{v}_n, \mathbf{n}_i)), \tau = dot(\mathbf{n}_{s_k}, \mathbf{n}_i) \tag{5}$$

where $\boldsymbol{v}_n$ is a normalized vector between $\mathbf{p}(i)$ and surfel centroid s_{k_c}. In order to define our objective function, we first define three feature histograms Φ, Ψ, and Ω based on the distribution of ϕ, ψ, and α values of the points in s_k respectively. Mathematically, Φ, Ψ, and Ω are written as:

$$\Phi_{s_k} = ||hist(\{\phi_i\})||, \Psi_{s_k} = ||hist(\{\psi_i\})||, \Omega_{s_k} = ||hist(\{\alpha_i\})||, \forall i \in s_k, \tag{6}$$

where, the term $||hist||$ represents the normalized histogram. For a given initial solution S, our objective function $C(S)$ is given by:

$$C(S) = \sum_{k=1}^{K} \left(\sqrt{\frac{1}{n_b} \sum_{j=1}^{n_b} (\Phi_{s_k}^j)^2} + \sqrt{\frac{1}{n_b} \sum_{j=1}^{n_b} (\Psi_{s_k}^j)^2} + \sqrt{\frac{1}{n_b} \sum_{j=1}^{n_b} (\Omega_{s_k}^j)^2} \right), \tag{7}$$

where $\Phi_{s_k}^j$, $\Psi_{s_k}^j$, and $\Omega_{s_k}^j$ represent the number of entries in the j^{th} bin of the corresponding histograms. The term n_b represents the total number of bins of the corresponding histogram. In our experiments, we divide the ϕ and ψ values in 50 bins ranging from 0 to 180 deg, and the α values in 50 bins ranging from $-\pi$ to π. The simulated annealing algorithm updates S iteratively as follows: at each iteration, the algorithm generates a new partitioning S_{new} based on the previous one (i.e., S_{old}) by moving the boundary points between the neighboring regions. A boundary movement proceeds as follows: first, the algorithm places a rectangular patch of $\sigma \times \sigma$ pixels around a query boundary point p and computes its distance with respect to each surfel center s_k which lies within the patch and is adjacent in the point cloud (i.e., the minimum pairwise distance between p and the points of s_k is less than a distance threshold $\delta_{adj} = 5mm$), as:

$$\zeta(p, s_k) = \sqrt{||c_p - \bar{c}_{s_k}||^2 + ||\beta_p - \bar{\beta}_{s_k}||^2}, \tag{8}$$

where, $\bar{c}_{s_k}$, and $\bar{\beta}_{s_k}$ represent the mean of the CIELab color, and β values of all points within the surfel s_k respectively. Next, the algorithm assigns the label of the surfel with the minimum ζ value to the query boundary point p. This procedure is repeated for all boundary points of each surfel in S to generate S_{new}. The new partitioning (S_{new}) is evaluated using the objective function in Eq. 7 and is accepted as valid if it increases $C(S)$. This iterative refinement is repeated until either of the two following termination conditions is met: **i**) the change in the cost between two successive iterations is less than a threshold ε, or **ii**) the total number of iterations exceeds a predefined number N_{max}. The final partitioning, S^*, is subsequently used to produce high-level object hypotheses (Sec. 3.2). Our proposed initial segmentation algorithm has several advantages including: **i)** simulated annealing refines the surfels iteratively and produces a valid segmentation at the end of each iteration. This is advantageous because

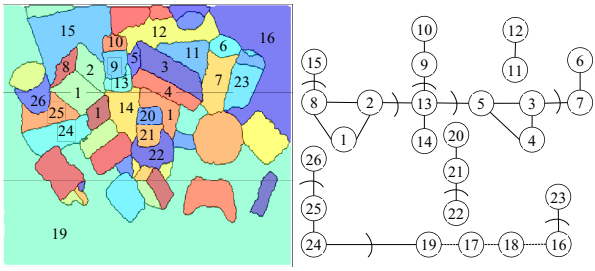

Fig. 3. The merging process of surfels based on graph-cut. Left: the segmentation into surfels. Right: our undirected graph structure. Cuts (arcs) are placed between the nodes which do not satisfy convex shape relationship.

we can stop the algorithm at any iteration and still achieve a valid segmentation, compared to the graph-based or region-growing methods, where one has to wait until all the cuts have been added to the graph, or until the growing is completely performed. **ii)** Our algorithm successfully recovers boundary errors (caused by missing depth information or smooth variations in the normal directions of touching surfaces) from a single RGB-D image of the scene. This is advantageous in situations where only a single partial view of the environment is available.

3.2 Perceptual Grouping into High-Level Object Hypotheses

The next key contribution is the merging of the surfels (i.e., whether two surfels correspond to the same object instance) to generate high-level object hypotheses in a model-free manner (i.e, without any prior knowledge of the objects in the scene). Our perceptual grouping algorithm follows a graph-based merging approach in which an undirected adjacency graph (built on surfels using adjacency relations such as 3D proximity) is reduced in a greedy manner. As illustrated in Fig. 3, the nodes of the graph represent the surfels and an edge between any pair represents the adjacency between the corresponding surfels. The cuts are placed on the edges which do not satisfy the merging criterion. Our merging criterion is based on the shape relationship (i.e., convex or concave) between two adjacent surfels. To estimate the shape relationship between two adjacent surfels s_k, and s_j, we construct a straight line between a point from s_k (i.e., the point which is maximally distant above/below the s_j plane) and a point from s_j (i.e., the point which is maximally distant above/below the s_k plane). As illustrated in Fig. 4, this line lies below both the surfels when they belong to the same object instance (e.g., two sides of a box shown in Fig. 4A). When the surfels belong to different object instances (e.g., box placed on a cylindrical container as shown in Fig. 4C), this line lies above either of the two surfels. Mathematically, we compute the projected distance d_p of the midpoint of the line from each surfel plane (see Fig. 4). If $d_p < 0$ for both the surfels, we report that the two surfels have a

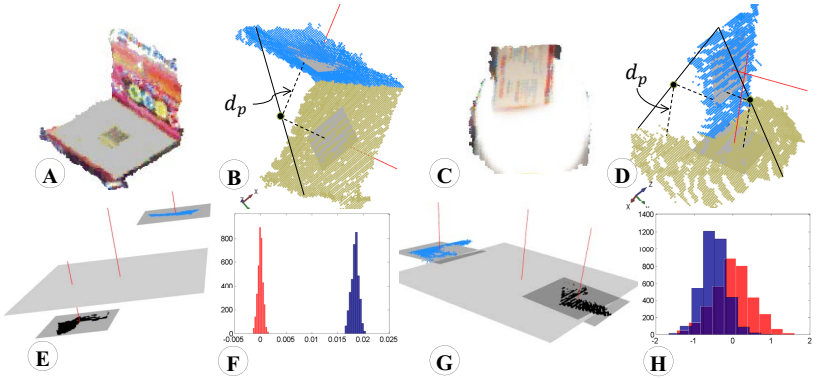

Fig. 4. Top row: Shape relationship analysis between two adjacent surfels. (A-B): when surfaces belong to the same object instance, the line lies below both surfels. (C-D): when one surface protrudes from its supporting surface, the respective line lies above either of the two surfaces. Bottom row: Co-planarity between two surfaces. (E-F): when surfaces are not aligned, their distance histograms do not intersect. (G-H): the distance histograms of co-planar surfaces intersect.

convex-shape, otherwise a concave-shape relationship. We iteratively check this condition for a set of 5 lines (in our experiments) and merge the two surfels if all lines satisfy the convex-shape relationship criterion. This procedure efficiently combines surfels which belong to distinct object instances however, ignores those which are not adjacent in the 3D Cartesian space (e.g., in Fig. 3, surfels 16, 17, 18, and 19 belong to the floor surface but are not connected in the 3D Cartesian space). To efficiently handle this problem, we detect surfels which are co-planar and do not have a convex relationship with any surfel (e.g., see edges represented by the dotted lines in Fig. 3-Right), and merge them subsequently. To measure co-planarity between two surfels, we check if they have similar mean normal directions and are aligned. The alignment is checked by building a histogram of the projected distances of each surfel points from the combined plane (i.e., the plane which best fits the 3D points of the union of the two surfels) and computing their intersection. As shown in Fig. 4F, the histograms of two non-aligned surfels do not intersect. On the other hand, the histograms of a co-planar pair intersect (as shown in Fig. 4H).

3.3 Grasp Synthesis

We apply our layered representation of the scene (i.e., surfels and object hypotheses) to perform grasp synthesis of unknown objects in a model-free manner. Our proposed algorithm proceeds in two steps. **i)** grasp generation, and **ii)** grasp selection. **For grasp generation**, the algorithm generates pairs of grasping points using reflection symmetry (i.e, co-linearity of the 3D points and similar orientations of their surface normals in the 2D image plane and the 3D Cartesian space) between points along the boundaries of the surfel. **For grasp selection**,

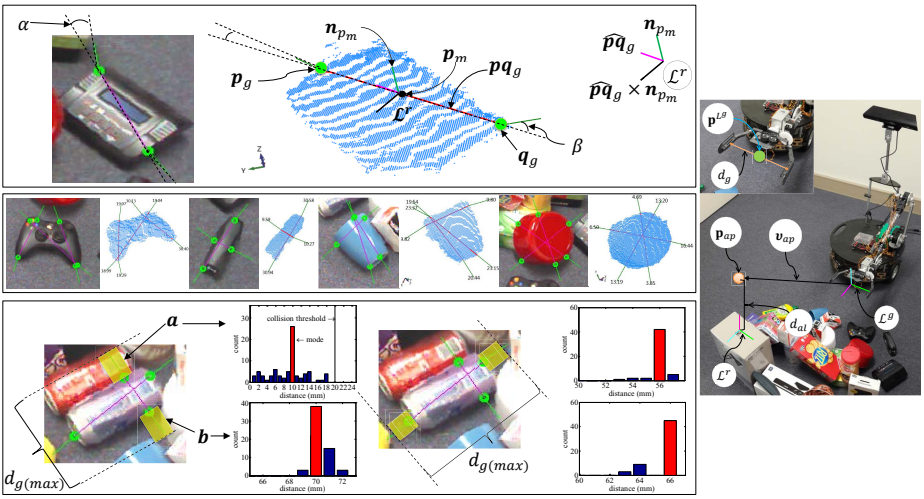

Fig. 5. Grasp synthesis using surfels. Left (top row): shows the angles α and β in the 2D image plane and the 3D Cartesian space respectively. Left (middle row): shows grasping points which satisfy our reflection symmetry criterion. Left (bottom row): shows the patches a and b used to extract shape descriptors. Their corresponding distance histograms show mode values (highlighted in red) smaller and larger than a threshold ($20mm$) for the cases when majority points are collision (e.g., a) and void (e.g., b) respectively. Right: illustration reference frames L^g and L^r, the approach point $\boldsymbol{p}_{ap}$, and the approach vector $\boldsymbol{v}_{ap}$.

the algorithm evaluates the grasp hypotheses based on the shape descriptors (a distance histogram representing the distances between a grasping point and all points in a rectangular region) of their corresponding grasping points to select the most appropriate grasp.

Grasp Generation. Our general grasp notation G for a surfel s is defined as:

$$\boldsymbol{G}_s = \{\boldsymbol{p}_g, \boldsymbol{q}_g, L^r, \boldsymbol{p}_{ap}\} \tag{9}$$

As illustrated in Fig. 5, $\boldsymbol{p}_g$ and $\boldsymbol{q}_g$ are two grasping points located on the boundary of the surfel s. L^r is a local reference frame whose origin is the midpoint $\boldsymbol{p}_m$ of the vector $\boldsymbol{p}_g\boldsymbol{q}_g$ (i.e., straight line connecting the two grasping points). The x, z and y axes of L^r correspond to the direction vector $\boldsymbol{p}_g\hat{\boldsymbol{q}}_g$, surface normal $\boldsymbol{n}_{p_m}$ and $\boldsymbol{p}_g\hat{\boldsymbol{q}}_g \times \boldsymbol{n}_{p_m}$ respectively. $\boldsymbol{p}_{ap}$ is a point located at a distance d_{al} from L^r along $+n_{p_m}$ (positive direction of the normal) and is used as the approach point to align the gripper for the corresponding grasp (see Fig. 5-right). The grasping points are determined by finding two points from the surfel boundaries, which satisfy the reflection symmetry criteria in both the 2D image plane and the 3D Cartesian space (i.e., their surface normals minimize the angles α, and β as shown in Fig. 5-left-top). The algorithm stores all pairs of $(\boldsymbol{p}_g, \boldsymbol{q}_g)$ whose corresponding angles α, and β, are within ± 2 (as shown in Fig. 5-left-middle). This

procedure is repeated for each surfel and the generated grasp hypotheses are subsequently ranked as explained in the following.

Grasp Selection. For each object of the segmented scene, its generated grasp hypotheses are first ranked in the order of increasing distance to the mean position of their corresponding surfel. This encourages the grasping of the object from a point that is close to its center of gravity. Next, the ranked grasp hypotheses are evaluated in terms of the shape descriptors of their corresponding grasping points. The extraction of our shape descriptor proceeds in two steps: i) a rectangular patch is constructed in the 2D image plane such that its major and minor axes are co-linear and perpendicular to the the line $\boldsymbol{p}_g\boldsymbol{q}_g$ respectively (see Fig. 5-left-bottom). ii) Next, each neighboring point (i.e, the point which lies in the rectangular patch and does not belong to the corresponding surfel) is iteratively classified as void or collision (i.e., if its distance to the corresponding grasping point is greater or less than a distance threshold respectively). If several points in the patch are classified as collision (i.e., the mode of patch's distance histogram is less than a fixed threshold), the corresponding grasp hypothesis is discarded. This facilitates the selection of grasp hypotheses, whose corresponding grasping points have sufficient void space for the gripper to fit its fingers without collision with the neighboring surfaces. After the evaluation of the grasp hypotheses based on their shape descriptors, the highest ranked grasp is selected as the most appropriate for grasp execution.

Grasp Execution. This section describes our grasp execution procedures. Once the most appropriate grasp is selected by the grasp selection algorithm, inverse kinematics is used to drive the gripper to the grasp location. Inverse kinematics represent the following mapping:

$$G_s = (\boldsymbol{p}_g, \boldsymbol{q}_g, L^r, \boldsymbol{p}_{ap}) \rightarrow (L^g, d_g). \tag{10}$$

As illustrated in Fig. 5-right, L^g denotes the gripper reference frame calibrated with respect to the camera. The orientation of L^g is such that $\boldsymbol{z}^{L^g}$(z-axis of L^g frame) is parallel to the grippers fingers, $\boldsymbol{x}^{L^g}$ connects the fingers and $\boldsymbol{y}^{L^g} = \boldsymbol{z}^{L^g} \times \boldsymbol{x}^{L^g}$, and the origin $\boldsymbol{p}^{L^g}$ is placed between the two fingers. The term d_g corresponds to the distance between the fingers. The grasp execution algorithm proceeds as follows: the gripper is set to a pre-grasp configuration (i.e., $d_g = d_{g(max)}$), and is moved to the approach point, $\boldsymbol{p}_{ap}$ (i.e., $\boldsymbol{p}^{L^g} = \boldsymbol{p}_{ap}$), along the shortest directed path $\boldsymbol{v}_{ap}$. Next, the gripper orientation is set to align with the object surface (i.e., $\boldsymbol{z}^{L^g} = \boldsymbol{n}_{\boldsymbol{p}_m}$), and $\boldsymbol{x}^{L^g} = \boldsymbol{p}_g\hat{\boldsymbol{q}}_g$, and the gripper is translated along the direction $\boldsymbol{n}_{\boldsymbol{p}_m}$ until it reaches the grasp-position (i.e., $\boldsymbol{p}^{L^g} > \boldsymbol{p}_m$). Finally, the fingers move from the pre-grasp configuration to the grasp-configuration (i.e., $d_g \leq |\boldsymbol{p}_g\boldsymbol{q}_g|$) and the grasp-execution concludes when the joints settle in a static configuration. There are several advantages of our grasp selection algorithm which include: i) Using surfels to generate pairs of grasping points facilitates the determination of appropriate pairs and reduces

outliers (i.e., grasping points which belong to other object instances). This also reduces the computational complexity by analyzing a fixed number of object surfaces compared to the methods (e.g., [14,17]), which search for grasping points in the entire image space or the entire point cloud. ii) The reflection symmetry criterion reduces the number of grasp hypotheses to a small number of appropriate grasps from which the best grasp is automatically selected using our ranking procedure (based on the shape descriptors of the corresponding grasping points). This is computationally more efficient compared to the methods which use shape abstractions to evaluate a nearly infinite number of possible grasp hypotheses [12].

4 Experiments

We evaluated the performance of our algorithms in terms of segmentation accuracy (Sec. 4.2) and grasp selection accuracy (Sec. 4.3). To quantify the performance and to comprehensively compare with the state-of-the-art methods, we used three popular object datasets which are publicly available. We also tested the performance on live video streams from Microsoft Kinect to validate the suitability of our algorithms for robotic applications (see video in the supplementary material). All experiments were done on a multi-core i7 machine without any GPU support.

4.1 Datasets

We used two datasets for the evaluation of our segmentation performance. The first dataset is the Washington RGB-D object dataset (WRGBD) [15], which provides 8 different video sequences of office and kitchen environments. Each video sequence in the dataset is a series of RGB-D images captured using a Kinect sensor from different viewpoints. The variable characteristics of the scenes such as different illumination settings, variable viewpoints, and a large variety of objects, make WRGBD a very challenging dataset for object segmentation purposes. The ground truth of the WRGBD dataset is available in two forms: 2D bounding boxes around salient objects in each scene and, labeled point clouds for each video sequence. The second dataset is the Object Segmentation Database (OSD) [26]. OSD contains RGB-D images of table-top scenes in which a large variety of objects are stacked over each other in several layouts. This is a great challenge for the segmentation algorithms to separate distinct objects particularly when they occlude each other. The ground truth for this dataset is available in the form of manually annotated segmentation masks for each object in the scene. For the evaluation of our grasp selection algorithm, we used the Cornell grasping dataset [13]. This dataset contains 1035 images of 280 different objects in different layouts, each annotated with several ground-truth positive and negative grasping rectangles. A grasping rectangle is an oriented rectangle in the image plane, which defines the orientation of a parallel gripper with respect to the image plane [13].

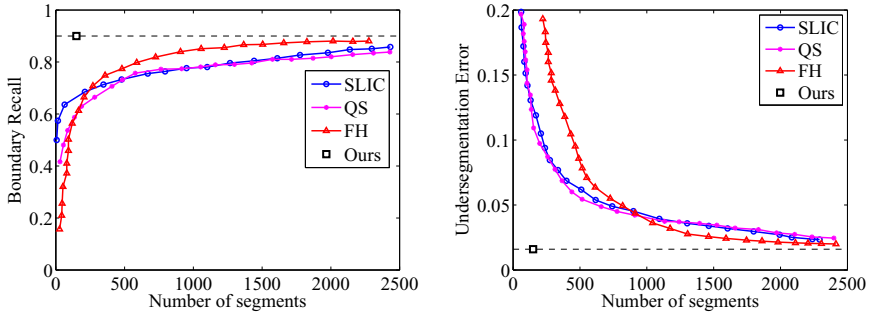

Fig. 6. 3D Boundary recall (left) and 3D under-segmentation errors (right) of our algorithm compared to the other methods on the OSD dataset.

Table 1. Object segmentation results on WRGBD dataset

	percentage of objects detected						
Object category	Soda Can	Coffee Mug	Cap	Bowl	Flashlight	Cereal Box	Average
Method in [20]	90.3%	84.2%	88.0%	85.5%	98.3%	95.4%	90.3%
Ours	**93.5**%	**87.3**%	**91.7**%	**90.5**%	**98.5**%	**98.2**%	**93.3**%

4.2 Evaluation of Object Segmentation

To quantitatively measure segmentation quality (i.e., how accurately the boundaries of its segments conform to the physical object boundaries), we used two standard metrics namely: 3D boundary recall, and 3D under-segmentation error as suggested in [37]. We compared our results with three state-of-the-art segmentation methods namely: hierarchical graph-based (GBH) [10], SLIC [1] and Quickshift (QS) [36]. The boundary recall and under-segmentation results are shown in Fig. 6-left and Fig. 6-right respectively. The results show that the performance of our algorithm is not dependent on the number of segments. On the contrary, other methods (i.e., [10,1,36]) improve as the number of segments (specified by the user) increase. Furthermore, our algorithm achieved the best boundary recall and under-segmentation results compared to the other methods. These improvements are credited to our proposed optimization-based segmentation refinement procedure (see Sec. 3.1 for details), which results in superior conformity of the surfel boundaries to the physical boundaries compared to state-of-the-art methods. Table. 1 shows our object segmentation results on the WRGBD. As shown in the table, our approach achieved the best performance in the four selected object categories and, on average, attained at least a 3.0% higher average precision compared to the method in [20]. On the OSD dataset, our algorithm achieved the best performance in the precision scores compared to the methods in [27] and [20] as shown in Table 2. These improvements are credited to our proposed merging criteria (see Sec. 3.2 for details), which efficiently combines perceptually similar surfels into distinct object hypotheses.

Table 2. Object segmentation results on OSD dataset

Algorithm	Precision (%)	Recall (%)
Ours	**94.41**	**96.20**
[27] two SVMs	89.98	97.05
[27] one SVM	93.75	95.48
[20]	62.14	70.93

Table 3. Evaluation of Grasp selection performance

Algorithm	Distance (%)	Orientation (%)
Ours	$\mathbf{77.6 \pm 2.5}$	$\mathbf{61.5 \pm 1.8}$
[13]	75.3 ± 2.9	60.1 ± 3.2
[24]	70.3 ± 6.3	58.5 ± 7.2
[28]	68.6 ± 10.5	52.9 ± 8.8

4.3 Evaluation of Grasp Selection

We compared our grasp selection results (on the Cornell grasping dataset) with the methods in [28,24]. For our evaluation, we compared the top-ranked rectangle for each method with the set of ground-truth rectangles for each image. We present our results using distance and orientation metrics as suggested in [13]. Table. 3 shows the results of our proposed grasp selection algorithm compared with the other methods. Our proposed algorithm outperforms the other segmentation-based grasp selection methods [24,28] by up to 7.3% for the distance metric and 3% for the orientation metric. Our results also show a superior performance (i.e., an improvement of 2.3% for the distance and 1.4% for the orientation metrics respectively) compared to the learning-based method in [13]. These improvements are credited to the use of our proposed reflection symmetry criteria to generate appropriate grasp hypotheses on surfel boundaries which are subsequently evaluated using our proposed shape descriptor to automatically select the most appropriate grasp for an unknown object in a model-free manner (see Sec. 3.3 for detail).

5 Conclusion

We successfully addressed the challenging problems of the segmentation and grasp selection of unknown stacked and occluded objects without the use of any prior information (model-free) of the objects or the environment. This was accomplished by the introduction of a novel 3D segmentation algorithm, which efficiently handles the case of stacked and occluded objects. We subsequently presented a novel grasp selection algorithm, which generates appropriates grasps hypotheses using surfel boundaries and automatically selects the most appropriate grasp in a model-free manner. Our segmentation and grasp-selection results show superior performance compared to the state-of-the-art methods.

Acknowledgment. This work was supported by Australian Research Council grants (DP110102166, DE120102960).

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. PAMI 34(11), 2274–2282 (2012)

2. Asif, U., Bennamoun, M., Sohel, F.: Real-time pose estimation of rigid objects using rgb-d imagery. In: ICIEA, pp. 1692–1699 (2013)
3. Bohg, J., Kragic, D.: Learning grasping points with shape context. Robotics and Autonomous Systems 58(4), 362–377 (2010)
4. Borst, C., Fischer, M., Hirzinger, G.: Grasp planning: How to choose a suitable task wrench space. In: ICRA, vol. 1, pp. 319–325 (2004)
5. Bowers, D.L., Lumia, R.: Manipulation of unmodeled objects using intelligent grasping schemes. IEEE Transactions on Fuzzy Systems 11(3), 320–330 (2003)
6. Collet, A., Martinez, M., Srinivasa, S.S.: The moped framework: Object recognition and pose estimation for manipulation. IJRR 30(10), 1284–1306 (2011)
7. Curtis, N., Xiao, J.: Efficient and effective grasping of novel objects through learning and adapting a knowledge base. In: IROS, pp. 2252–2257 (2008)
8. Dogar, M.R., Hsaio, K., Ciocarlie, M., Srinivasa, S.: Physics-based grasp planning through clutter (2012)
9. Ekvall, S., Kragic, D.: Integrating object and grasp recognition for dynamic scene interpretation. In: ICAR, pp. 331–336 (2005)
10. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. IJCV 59(2), 167–181 (2004)
11. Fenzi, M., Dragon, R., Leal-Taixé, L., Rosenhahn, B., Ostermann, J.: 3d object recognition and pose estimation for multiple objects using multi-prioritized ransac and model updating. In: Pattern Recognition, pp. 123–133 (2012)
12. Huebner, K., Ruthotto, S., Kragic, D.: Minimum volume bounding box decomposition for shape approximation in robot grasping. In: ICRA, pp. 1628–1633 (2008)
13. Jiang, Y., Moseson, S., Saxena, A.: Efficient grasping from rgbd images: Learning using a new rectangle representation. In: ICRA, pp. 3304–3311 (2011)
14. Kootstra, G., Popović, M., Jørgensen, J.A., Kuklinski, K., Miatliuk, K., Kragic, D., Krüger, N.: Enabling grasping of unknown objects through a synergistic use of edge and surface information. IJRR 31(10), 1190–1213 (2012)
15. Lai, K., Bo, L., Ren, X., Fox, D.: A large-scale hierarchical multi-view rgb-d object dataset. In: ICRA, pp. 1817–1824 (2011)
16. Le, Q.V., Kamm, D., Kara, A.F., Ng, A.Y.: Learning to grasp objects with multiple contact points. In: ICRA, pp. 5062–5069 (2010)
17. Lenz, I., Lee, H., Saxena, A.: Deep learning for detecting robotic grasps. arXiv preprint arXiv:1301.3592 (2013)
18. Li, X., Guskov, I.: 3d object recognition from range images using pyramid matching. In: ICCV. pp. 1–6 (2007)
19. Miller, A.T., Allen, P.K.: Graspit! a versatile simulator for robotic grasping. IEEE Robotics & Automation Magazine 11(4), 110–122 (2004)
20. Mishra, A.K., Shrivastava, A., Aloimonos, Y.: Segmenting "simple" objects using rgb-d. In: ICRA. pp. 4406–4413 (2012)
21. Morales, A., Sanz, P.J., Del Pobil, A.P., Fagg, A.H.: Vision-based three-finger grasp synthesis constrained by hand geometry. Robotics and Autonomous Systems 54(6), 496–512 (2006)
22. Pelossof, R., Miller, A., Allen, P., Jebara, T.: An svm learning approach to robotic grasping. In: ICRA, vol. 4, pp. 3512–3518 (2004)
23. Platt, R.: Learning grasp strategies composed of contact relative motions. In: 7th IEEE-RAS International Conference on Humanoid Robots, pp. 49–56 (2007)
24. Rao, D., Le, Q.V., Phoka, T., Quigley, M., Sudsang, A., Ng, A.Y.: Grasping novel objects with depth segmentation. In: IROS, pp. 2578–2585 (2010)

25. Recatalá, G., Chinellato, E., Del Pobil, Á.P., Mezouar, Y., Martinet, P.: Biologically-inspired 3d grasp synthesis based on visual exploration. Autonomous Robots 25(1-2), 59–70 (2008)
26. Richtsfeld, A., Morwald, T., Prankl, J., Zillich, M., Vincze, M.: Segmentation of unknown objects in indoor environments. In: IROS, pp. 4791–4796 (2012)
27. Richtsfeld, A., Mörwald, T., Prankl, J., Zillich, M., Vincze, M.: Learning of perceptual grouping for object segmentation on rgb-d data. Journal of Visual Communication and Image Representation 25(1), 64–73 (2014)
28. Richtsfeld, M., Vincze, M., et al.: Grasping of unknown objects from a table top. In: Workshop on Vision in Action: Efficient Strategies for Cognitive Agents in Complex Environments (2008)
29. Rosales, C., Porta, J.M., Ros, L.: Global optimization of robotic grasps. In: Proceedings of Robotics: Science and Systems VII (2011)
30. Rusu, R.B., Bradski, G., Thibaux, R., Hsu, J.: Fast 3d recognition and pose using the viewpoint feature histogram. In: IROS, pp. 2155–2162 (2010)
31. Saxena, A., Driemeyer, J., Kearns, J., Osondu, C., Ng, A.Y.: Learning to grasp novel objects using vision. In: ISER (2006)
32. Saxena, A., Driemeyer, J., Ng, A.Y.: Robotic grasping of novel objects using vision. IJRR 27(2), 157–173 (2008)
33. Shapiro, L., Stockman, G.C.: Computer vision. In: Computer Vision 2001, pp. 69–75. Prentice Hall (2001)
34. Sun, M., Bradski, G., Xu, B.-X., Savarese, S.: Depth-encoded hough voting for joint object detection and shape recovery. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 658–671. Springer, Heidelberg (2010)
35. Tegin, J., Ekvall, S., Kragic, D., Wikander, J., Iliev, B.: Demonstration-based learning and control for automatic grasping. Intelligent Service Robotics 2(1), 23–30 (2009)
36. Vedaldi, A., Soatto, S.: Quick shift and kernel methods for mode seeking. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 705–718. Springer, Heidelberg (2008)
37. Veksler, O., Boykov, Y., Mehrani, P.: Superpixels and supervoxels in an energy optimization framework. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part V. LNCS, vol. 6315, pp. 211–224. Springer, Heidelberg (2010)