IEEE Robotics and Automation Letters (RAL) paper presented at the
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Point Cloud Projective Analysis for Part-based Grasp Planning

Riccardo Monica[1], Jacopo Aleotti[1]

Abstract— **This work presents an approach for part-based grasp planning in point clouds. A complete pipeline is proposed that allows a robot manipulator equipped with a range camera to perform object detection, categorization, segmentation into meaningful parts, and part-based semantic grasping. A supervised image-space technique is adopted for point cloud segmentation based on projective analysis. Projective analysis generates a set of 2D projections from the input object point cloud, labels each object projection by transferring knowledge from existing labeled images, and then fuses the labels by back-projection on the object point cloud. We introduce an algorithm for point cloud categorization based on 2D projections. We also propose a viewpoint aware algorithm that filters 2D projections according to the scanning path of the robot. Object categorization and segmentation experiments were carried out with both synthetic and real datasets. Results indicate that the proposed approach performs better than a CNN-based method for a training set of limited size. Finally, we show part-based grasping tasks in a real robotic setup.**

I. INTRODUCTION

Semantic grasping is the problem of planning appropriate robot grasps on an object in order to perform a task. One way to select task-oriented grasps is to determine the object category, and then to segment the object into a set of meaningful parts. Given the object category it is possible to label each part with semantic information (e.g. a handle). Knowledge of part labels can finally be used to plan grasps on the appropriate object part to perform the assigned task.

This work presents an approach for semantic part-based object grasping based on point clouds. An advantage of working on a point cloud is that object segmentation does not need to model objects using polygon meshes. Moreover, grasp planning is performed directly on the point cloud representing the selected object part to be grasped. The proposed method is based on a technique for object point cloud segmentation that relies on projective analysis [1]. Projective analysis is an image-space supervised learning approach that provides a semantic segmentation of a point cloud into meaningful parts. Given an input object point cloud, segmentation is carried out by first generating a set of 2D projections (images) from multiple virtual viewpoints. Each projection is then segmented by transferring labels from the most similar images in the training set that belong to the same category, using a Bi-class Symmetric Hausdorff distance (BiSH). Finally, labeled images are projected back in 3D space, and consistently merged together to obtain the segmented point cloud. The main advantage of projective analysis is that it works in a low-dimensional 2D space and, therefore, it is applicable for part-based grasping of partially observed objects described by incomplete point clouds.

In summary, this paper provides the following technical contributions. A complete pipeline is proposed for object categorization, segmentation into parts, part labeling, and part-based grasping using point clouds. The method was evaluated on a real robot platform which consists of a robot manipulator equipped with a parallel gripper and a range camera in eye-in-hand configuration. We present a novel point cloud categorization algorithm that exploits the Bi-class Symmetric Hausdorff distance. We also propose a viewpoint-aware algorithm for filtering out 2D object projections that would result in incomplete images, due to occlusion or non-optimal scan path. Experiments of object categorization and segmentation have been carried out with both complete synthetic data (without noise) and point clouds from real scans (including incomplete scans). The proposed method was compared against PointNet [2], a state of the art deep learning model for point cloud classification and part-based semantic segmentation. Results indicate that the proposed approach performs better than the convolutional neural network for a training set of limited size, which we are more interested in, due the absence of large training sets for 3D shape segmentation, and due to the effort involved in finding similar objects to build a dataset from 3D scans. For reproducibility we make our dataset as well as the source code publicly available. Finally, we report experiments of part-based grasp planning on the real robot.

II. RELATED WORK

The closest works to ours that perform object categorization, segmentation, and part-based grasping are [3], [4], [5]. In [3] a method for grasp planning across familiar objects was presented. The approach was based on polygon meshes rather than point clouds, and the object parts were modeled as simple geometric primitives like planes, cylinders and spheres. Antanas et al. [4] proposed a method for semantic part-based grasping using point clouds, however, they assumed that objects were symmetric and that object parts belonged to five semantic labels (top, middle, bottom, handle, usable). In [5] a method for object categorization and grasping by part was presented, based on Reeb graph shape segmentation of polygon meshes. The method did not exploit low level geometrical features to distinguish between classes with the same topology of connected object parts, therefore, for example, it could not discern the differences between a cup and a hammer, or between a table and a four-legged toy.

[1] Riccardo Monica and Jacopo Aleotti are with RIMLab, Robotics and Intelligent Machines Laboratory, Department of Engineering and Architecture, University of Parma, Italy riccardo.monica@unipr.it, jacopo.aleotti@unipr.it

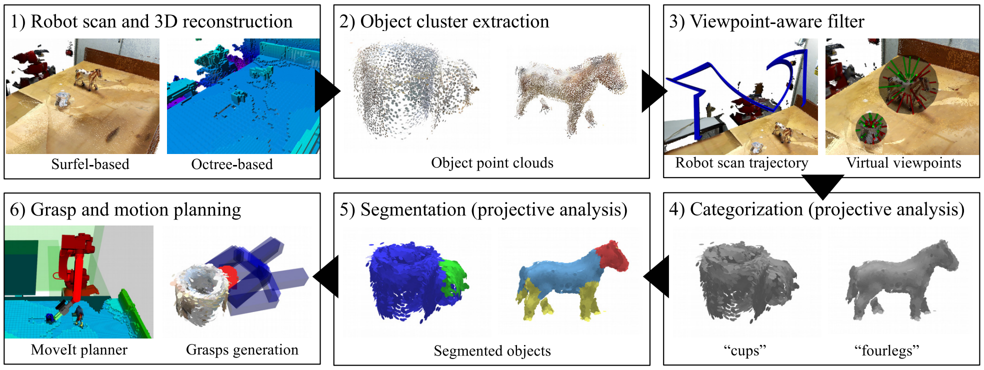
1) Robot scan and 3D reconstruction
Surfel-based
Octree-based
2) Object cluster extraction
Object point clouds
3) Viewpoint-aware filter
Robot scan trajectory
Virtual viewpoints
6) Grasp and motion planning
MoveIt planner
Grasps generation
5) Segmentation (projective analysis)
Segmented objects
4) Categorization (projective analysis)
“cups”
“fourlegs”

Fig. 1. Pipeline of the proposed approach for part-based grasp planning. The pipeline consists of 6 steps (from top left to bottom right).

A large body of previous works focused on detection of grasp affordances. A grasp affordance is a way of grasping an object to achieve a particular function (e.g. pouring, support, or tool-use). These approaches are part-agnostic, in the sense that they tackle the grasp planning problem by looking for object properties that afford an assigned task, which is chosen among a finite set of possible actions. In particular, a first group of works investigated object categorization and detection of grasp affordances [6], [7], [8], [9], [10], [11]. Some papers also included grasp synthesis in simulated or real environments [9], [10], [11]. A second group of works were also object agnostic, as they identified grasp affordances by relying solely on low-level object features, without performing object categorization. Several machine learning algorithms have been adopted to this purpose, and the methods also differ with respect to the input data. A point cloud object representation was adopted in [12], [13], [14], [15], [16], [17], while RGB-D data were used in [18], [19], [20], [21], [11], [22]. Three dimensional data were extracted for stereo vision by Kraft et al. [23], while the method in [24] was evaluated using a synthetic 3D dataset. Finally, 2D RGB images were considered in [25], [26]. In contrast to the works based on detection of part affordances, the goal of our method is to categorize an object, segment it into meaningful parts and assign a semantic label to each part. The robot can then grasp the most appropriate part of the object to perform the desired task. Other previous works explored geometric object segmentation strategies for grasp planning without performing part labeling [27], [28], [29], [30], [31], [32]. In [27], [28], [29], [30] methods were investigated for transferring grasps across similar objects, while in [31], [32] the authors proposed approaches to facilitate robot grasping by segmenting objects using superquadrics or a box-based representation. Finally, it is worth citing works that, although did not perform robot grasp planning, focused on shape segmentation to extract grasping areas [33], [34], [35].

III. METHOD

We consider a tabletop environment that contains multiple objects of different categories. Let $\mathcal{C}$ be the finite set of possible object categories, and $\mathcal{L}$ be the set of all possible labels of the object parts. The proposed grasp planning system, shown in Fig. 1, receives as input a pair $(c^\star, l^\star)$ composed of an object category $c^\star \in \mathcal{C}$ and part $l^\star \in \mathcal{L}$, meaning that the robot task is to grasp an object of class $c^\star$ by its part $l^\star$ (e.g. grasp a cup by the handle). The pipeline of the proposed method consists of six main steps. In the first step the robot manipulator performs a 3D scan of the environment, along a predefined scan path fixed for all experiments, by using an eye-in-hand depth sensor. The range images acquired in the scan phase are used to generate an occupancy octree Ω (OctoMap [36]) with resolution R_{octree}, as well as a surfel map Σ using ElasticFusion [37]. A surfel map is a point cloud where each surfel has attributes such as position, color, normal and radius. In the *object cluster extraction* phase (step 2), clusters of points are extracted from point cloud Σ, so that each cluster T_i represents one of the objects on the table. To this purpose we adopt a standard approach that detects the tabletop as the dominant plane using RANSAC plane fitting, with a threshold th_{plane}. After removing the points in Σ that belong to the tabletop the remaining points are partitioned into clusters T_i using Euclidean clustering, with distance threshold th_{cl}.

The following steps operate by projecting each point cloud T_i onto a set of 2D images, as if observed from different virtual camera viewpoints, located on the surface of a sphere centered on the object. Object categorization and segmentation are carried out by exploiting a dataset of segmented point clouds, which is pre-processed as illustrated in Section III-A. As object point cloud clusters may be incomplete due to partial observation or occlusion, in the *viewpoint-aware filter* phase (step 3) the virtual viewpoints are filtered so that only the 2D projections compatible with the robot scan path remain (Section III-B). Then, each object cluster T_i is classified into its respective object category c_i^T (step 4), and a target object $T_{i^\star}$ is selected to be grasped, so that $c_{i^\star}^T = c^\star$. In order to categorize an object a novel supervised algorithm is proposed that uses the Bi-class Symmetric Hausdorff distance (Section III-C) [1]. It is worth noting that the original work in [1] did not perform object categorization. Also, in [1] shape segmentation was evaluated only on complete point clouds. In the next phase semantic part-based segmentation is performed on the target cluster $T_{i^\star}$ (step 5), to assign a meaningful part label l_j to

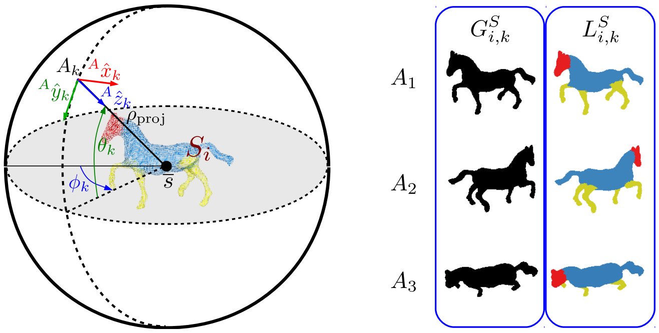

Fig. 2. Left: a segmented point cloud of the dataset (each part is displayed with a different color), and a virtual camera pose A_k, located on the surface of the sphere (parameterized by ϕ_k, θ_k) centered in $\overline{s}$ with radius ρ_{proj}. Camera z-axis $^A\hat{z}_k$ points towards $\overline{s}$. Right: three examples of generated binary images $G^S_{i,k}$ and label images $L^S_{i,k}$, for viewpoints A_1, A_2 and A_3.

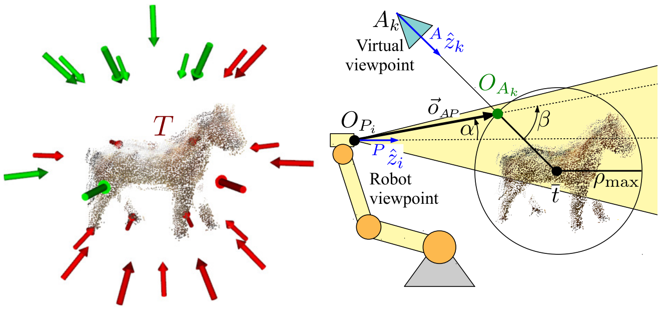

Fig. 3. Left: the virtual camera poses $A_k \in \mathcal{A}$ represented by red and green arrows pointing towards object point cloud T. Virtual camera poses represented by green arrows are compatible with at least one real robot observation pose P_i. Right: A_k is compatible with P_i if O_{A_k} is in the yellow cone (approximate field of view of the robot), if there are no occluding surfels, and if angle β between $^A\hat{z}_k$ and $\vec{o}_{AP}$ is small.

each point $t_j \in T_{i^\star}$. Part-based segmentation is performed by applying the projective analysis approach in [1], named BiSH-PA (Section III-D). For segmentation, only the objects in the dataset of the same class $c^T_{i^\star}$ are considered. Finally, in the *grasp and motion planning* phase (step 6, Section III-E) a set of grasp candidates on the target part $l_j = l^\star$ is generated, and a collision free path for the robot is planned to grasp the object by the target part.

A. Dataset pre-processing

The categorization and the BiSH-PA segmentation algorithms require a dataset $\mathcal{S}{=}\{S_i\}$ of complete object point clouds S_i, whose category $c^S_i{\in}\mathcal{C}$ is known in advance. Each point cloud of the dataset is annotated with a label mapping $\Lambda_i : S_i \rightarrow \mathcal{L}$ so that $\Lambda_i(s_{i,j})$ is the part label $l_{i,j}{\in}\mathcal{L}$ of point $s_{i,j}{\in}S_i$. Dataset S is preprocessed by projecting each point cloud $S_i{\in}\mathcal{S}$ on 2D images from a set of virtual viewpoints $\mathcal{A}{=}\{A_1 \dots A_{K_{\text{vv}}}\}$, of cardinality K_{vv}, uniformly distributed on a sphere and centered at the point cloud centroid $\overline{s}$ (Fig. 2, left). Images are obtained through a perspective projection onto a square image of size w_{g} pixels and focal length f_{g}. Each virtual camera pose A_k has the principal axis $^A\hat{z}_k$ pointing towards $\overline{s}$. Camera poses are parameterized using spherical coordinates, with azimuth angle $\phi_k{\in}\,[0, 2\pi]$, elevation $\theta_k{\in}\,[-\pi/2, \pi/2]$, and radius ρ_{proj}. The distance of the virtual camera to the object centroid ρ_{proj} is computed so that all point cloud projections fit in the image retaining a margin about equal to $w_{\text{g}}/10$ pixels per side, i.e. $\rho_{\text{proj}}{=}1.2\,\rho_{\text{max}}\,2f_{\text{g}}/w_{\text{g}}$ where ρ_{max} is the maximum distance of a point $s_j{\in}S_i$ from $\overline{s}$. Each projection $A_k{\in}\mathcal{A}$ of a dataset point cloud i generates a binary image $G^S_{i,k}(u, v)$, where pixel (u, v) is 0 if the object projects on (u, v), and 1 otherwise. Moreover, for each projection a label image $L^S_{i,k}$ is created, where each pixel contains the part label $L^S_{i,k}(u, v){=}l_{i,j}$ of the point projected on (u, v) (Fig. 2, right).

B. Viewpoint-aware point cloud projections generation

In step 3 of the proposed pipeline, for each object point cloud T a set of 2D images are generated from virtual viewpoints $A_k{\in}\mathcal{A}$, centered on centroid $\overline{t}$ of T (Fig. 3, left). As the object point cloud T may be incomplete, centroid $\overline{t}$ may not correspond to the actual object centroid. Hence, projection distance ρ_{proj} is recomputed as explained in Section III-A using T and $\overline{t}$ instead of S_i and $\overline{s}$. Moreover, some virtual object projections in $\mathcal{A}$ could be incomplete, and they could hinder the segmentation step, because the BiSH-PA algorithm could match a partial object image with a complete image of the object contained in the dataset. Therefore, we introduce a viewpoint-aware filter that takes advantage of the known robot scan path $\mathcal{P}$ to extract a subset $\mathcal{B} \subset \mathcal{A}$ of virtual view poses, which roughly correspond to actual viewpoints of the eye-in-hand sensor (Fig. 3, right).

Scan $\mathcal{P}$ is defined as a set of real sensor viewpoint poses P_i, where each pose is centered in O_{P_i} with principal axis $^P\hat{z}_i$. The algorithm filters out each virtual view pose A_k that is not compatible with any real robot observation pose P_i. Let O_{A_k} be the point on the sphere centered at the object centroid with radius ρ_{max}, and with the same azimuth ϕ_k and elevation θ_k of A_k. Being ρ_{max} approximately equal to the object radius, if O_{A_k} is within the field of view of a sensor view pose P_i, and the orientation of the real sensor view pose is similar to the orientation of the virtual view pose, then A_k is considered compatible with P_i. Moreover, O_{A_k} should be within the sensor range, and there should not be any occluding surfels. In particular, let the observation ray $\vec{o}_{AP}{=}O_{A_k}{-}O_{P_i}$ be the vector from O_{A_k} to O_{P_i}. Pose A_k is compatible with P_i (and it is inserted in $\mathcal{B}$) if the following conditions hold:

1) O_{A_k} is in the sensor field of view at P_i, i.e. angle α between $\vec{o}_{AP}$ and $^P\hat{z}_i$ is lower than threshold α_{max}:

$$\alpha = \arccos \frac{\langle \vec{o}_{AP}, {}^P\hat{z}_i \rangle}{\|\vec{o}_{AP}\|} < \alpha_{\text{max}} \tag{1}$$

2) angle β between $\vec{o}_{AP}$ and the virtual camera principal axis $^A\hat{z}_k$ is lower than a threshold β_{max}:

$$\beta = \arccos \frac{\langle \vec{o}_{AP}, {}^A\hat{z}_k \rangle}{\|\vec{o}_{AP}\|} < \beta_{\text{max}} \tag{2}$$

3) O_{A_k} is within the sensor range: $r_{\text{rmin}} < \|\vec{o}_{AP}\| < r_{\text{rmax}}$.

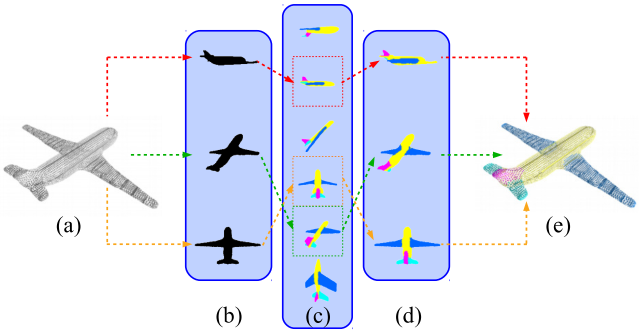

Fig. 4. BiSH-PA segmentation approach [1]: (a) unlabeled point cloud T, (b) projected binary images G_k^T of T, (c) matching of each binary image to a labeled image $L_{i,k}^S$ of an object of the same category in the dataset, (d) label transfer, (e) backprojection and optimization on point cloud T. Each part has a different color.

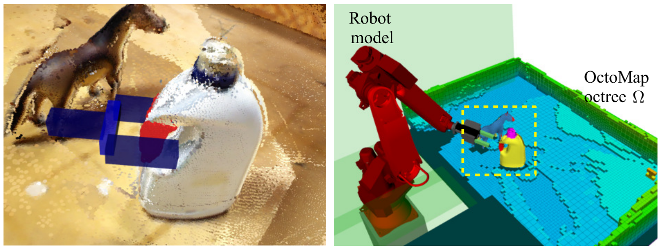

Fig. 5. Left: the point cloud Σ provided as input to GPD. Points on the target part (the jug handle) are highlighted in red. One of the generated grasp poses is displayed in blue. Right: the motion planning environment. The objects T_i in the dashed yellow square are represented by meshes for collision detection, and they have been removed from octree Ω.

TABLE I

EXPERIMENTAL PARAMETERS

Symbol	Value	Symbol	Value
α_{max}	$\pi/6$	β_{max}	$\pi/4$
w_{g}	256 pixels	K_{vv}	40
K_{proto}	2	R_{octree}	2 cm
th_{cl}	2 cm	th_{plane}	2 cm
r_{rmin}	0.5 m	r_{rmax}	2 m

4) there are no occluding surfels in the full point cloud Σ intersecting segment $\vec{o}_{AP}$.

For each viewpoint $A_k{\in}\mathcal{B}$ point cloud T is projected to generate the binary image G_k^T, and an index image $J_k^T(u,v)$ that contains the index j of the projected point $t_j{\in}T$.

C. Object point cloud categorization

The categorization phase determines the category $c^T{\in}\mathcal{C}$ of an object point cloud T as the category of the dataset most similar to T. Categorization exploits the Bi-class Symmetric Hausdorff distance BiSH(G_k, G'_k) between two binary images G_k and G'_k [1]. The BiSH metric cuts both input images into topologically homogeneous horizontal slabs and it accounts for internal holes. Then, the optimal mapping between the slabs of the two images is found. The value of BiSH is computed as the sum of the Bi-class Symmetric Hausdorff distance of each slab. On top of BiSH, we define the Bi-class Symmetric Hausdorff distance $\mathrm{BiSH_s}(\mathcal{G},\mathcal{G}')$ between two sets of binary images $\mathcal{G}{=}\{G_k\}$, $\mathcal{G}'{=}\{G'_\kappa\}$ as:

$$\mathrm{BiSH_s}(\mathcal{G},\mathcal{G}') = \min\left\{ \sum_{G_k\in\mathcal{G}} \min_{G'_\kappa\in\mathcal{G}'} \mathrm{BiSH}(G_k,G'_\kappa)\,, \sum_{G'_\kappa\in\mathcal{G}'} \min_{G_k\in\mathcal{G}} \mathrm{BiSH}(G_k,G'_\kappa) \right\} \qquad (3)$$

Let $\mathcal{S}_c{\subset}\mathcal{S}$ be the subset that contains all point clouds S_i of a single category $c_i^S{=}c$. Thus, the category c^T of point cloud T could be computed as

$$c^T = \operatorname*{argmin}_{c\in\mathcal{C}} \mathrm{BiSH_s}(\mathcal{G}_c, \mathcal{G}_\mathcal{B}(T)) \qquad (4)$$

where $\mathcal{G}_c$ is the set that contains all the binary images $G_{i,k}^S$ obtained from all the object point clouds $S_i{\in}\mathcal{S}_c$ of a single category c in the dataset, and $\mathcal{G}_\mathcal{B}(T)$ is the set of all the binary images of point cloud T from the filtered viewpoint set $\mathcal{B}$. However, (4) is computationally expensive, as it requires computation of the BiSH distance between each projection of T and all dataset images. Therefore, categorization is performed on a reduced dataset of images $\mathcal{G}'_c$, by selecting K_{proto} prototypes point clouds from each category subset $\mathcal{S}_c$. Category prototypes are selected by minimizing the intra-class $\mathrm{BiSH_s}$ distance, using k-means. Hence, category c^T of T is determined as

$$c^T = \operatorname*{argmin}_{c\in\mathcal{C}} \mathrm{BiSH_s}(\mathcal{G}'_c, \mathcal{G}_\mathcal{B}(T)) \qquad (5)$$

D. BiSH-PA algorithm for point cloud segmentation

After categorization, BiSH-PA segments an unlabeled point cloud T (Fig. 4a) of known category c^T using only the point clouds in the dataset of the same category $c_i^S{=}c^T$. A label l_j is assigned to each point $t_j{\in}T$ so that t_j and $s_{i,j}$ are in the same part if $l_j{=}\Lambda_i(s_{i,j})$, where $s_{i,j}$ is a point in the dataset and $\Lambda_i(s_{i,j})$ is the part label of $s_{i,j}$, as defined in Section III-A. In particular, BiSH-PA [1] works as follows:

1) Matching: for each binary image G_k^T of T, projected from a filtered viewpoint $A_k{\in}\mathcal{B}$, the most similar image $G_{i^\star,k^\star}^S$ is found among binary images $\mathcal{S}_c$ of the same category in the dataset (Fig. 4c), using the BiSH distance, i.e. $G_{i^\star,k^\star}^S{=}\mathrm{argmin}_{G_{i,k}^S\in\mathcal{G}_{c^T}} \mathrm{BiSH}(G_k^T, G_{i,k}^S)$

2) Transfer of part labels: BiSH also provides an optimal pixel mapping $(u',v'){=}M(u,v)$ between each image $G_k^T\,(u,v)$ and the closest image $G_{i^\star,k^\star}^S\,(u',v')$ in the dataset. Hence, for each image G_k^T a label image L_k^T is generated by transferring part labels from the label image $L_{i^\star,k^\star}^S$ of the dataset so that $L_k^T(u,v){=}L_{i^\star,k^\star}^S(M(u,v))$ (Fig. 4d).

3) Backprojection: part labels are projected back on T (Fig. 4e) from all the labeled images. Therefore, for each point $t_j \in T$, a set of potentially conflicting labels $\lambda_j{=}\left\{L_k^T(u,v)\,,\,\forall k,u,v \mid j = J_k^T(u,v)\right\}$ may occur.

4) Optimization: graph cut optimization to select the part label $l_j{\in}\lambda_j$ to be assigned to each point $t_j{\in}T$.

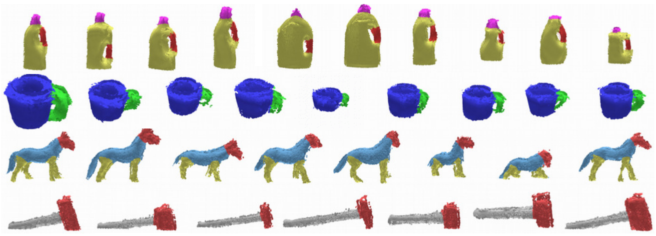

Fig. 6. Annotated point clouds (RD dataset) with a different color for each part.

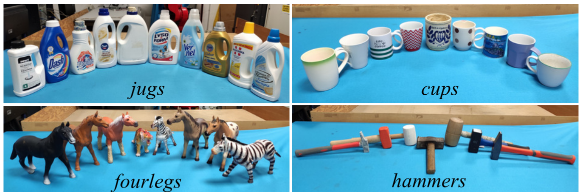

Fig. 7. The RD dataset (34 objects, 4 categories).

E. Part-based grasp planning

In order to generate grasp poses for the target object $T_{i^\star}$, on part $l^\star$, the Grasp Pose Detector (GPD) [38] was adopted. GPD requires as input for collision checking the full point cloud Σ of the environment. Moreover, input points from the target part must be provided to be used in the grasp generation process. Hence, we direct GPD grasp generation toward the target object part by providing as input the points with label $l^\star$ (Fig. 5, left). GPD outputs a set of grasp poses, ordered by score. The MoveIt planner (Fig. 5, right) is used to plan a robot path to reach the grasp poses. The collision environment in MoveIt includes the object point clouds T_i and the OctoMap occupancy octree Ω. Each surfel in T_i is converted to a hexagon polygon with the same radius. Both unknown and occupied voxels in Ω are considered obstacles in MoveIt. Any voxel closer than $2R_{\text{octree}}$ to a point in T_i is set to empty in Ω, to prevent MoveIt from reporting a collision between the object point cloud and the occupancy octree. Path planning is performed for each grasp pose, in decreasing order of score as ranked by GPD, and the first successful grasp is selected to be executed by the robot.

IV. RESULTS

The experimental setup consists of a 6-DOF robot manipulator (Comau SMART SiX) equipped with a parallel gripper and an Orbbec Astra-S 3D camera mounted on the end-effector. The robot forward kinematics was used in ElasticFusion to track the pose of the camera. The method was implemented using the ROS (Robot Operating System) framework, and it was evaluated on an Intel i7-6700 @ 3.40GHz, with a GeForce GTX 980 Ti. The BiSH-PA source code is available at `http://rimlab.ce.unipr.it/Software.html`, under `bish_segmentation`. Fixed parameters are reported in Table I. Virtual camera focal length f_{g} was set equal to the image size w_{g}. Two datasets of complete point clouds segmented into parts were considered for evaluation. Both datasets have a limited size as explained in the introduction. The first dataset comprises a total of 285 high quality models that belong to 15 object categories from the Princeton Segmentation Benchmark (PSB) [39]. As BiSH-PA operates on point clouds, each vertex of the polygonal model was converted into a surfel with the same normal, and radius 1 cm. The second dataset (RD) contains lower quality models generated from 3D scans of real objects using the robot setup. It contains a total of 34 objects, segmented into parts, divided into 4 categories: *jugs*, *cups*, *fourlegs* and *hammers* (Fig. 6 and 7). The RD dataset, annotated using the annotation tool [40], is available at `http://rimlab.ce.unipr.it/Software.html` under `grasping_rd_dataset`.

TABLE II

BiSH-PA RESULTS FOR DIFFERENT PARAMETER SETS (PS) ON THE PSB DATASET

ps	w_{g}	Slabs $G^S_{i,k}/G^T_k$	K_{vv}	Cat. (%)	Segm. (%)	Time (s)
Π_1	256	5/10	40	71.42	90.38	10
Π_2	512	5/10	40	70.00	89.40	34
Π_3	256	10/20	40	71.75	90.71	14
Π_4	256	5/10	24	67.75	87.76	7
Π_5	256	5/10	60	72.22	90.03	20

A. BiSH-PA evaluation

A first evaluation of BiSH-PA was carried out on the PSB dataset using 4-fold cross-validation. Results of categorization accuracy (Cat.), segmentation accuracy (Seg.) and computational time of a single point cloud (Time) are reported in Table II. Classification accuracy is defined as the percentage of correctly classified test point clouds, while segmentation accuracy is the percentage of points of correctly classified objects with part label equal to the ground truth. Experiments were performed with different sets of the following parameters: image size w_g, number of horizontal slabs for $G^S_{i,k}$ and G^T_k, and number of virtual viewpoints K_{vv}. The number of horizontal slabs of the dataset images $G^S_{i,k}$ was set as half the number of slabs of the test images G^T_k, as suggested in [1]. Table II indicates that parameter set Π_1 achieved the best compromise between accuracy and efficiency (about 10 s of computation time) and, therefore, it was selected for all the experiments reported hereafter. In particular, increasing the image resolution (from 256 to 512 pixels in parameter set Π_2) increases the computation time, but it does not improve neither segmentation nor categorization. Moreover, an increased number of horizontal image slabs provides only a slight improvement (parameter set Π_3). Finally, the reduction of viewpoints K_{vv} hampers performance by about 3% (parameter set Π_4), whereas increasing K_{vv} has a limited affect on accuracy, but it results in a higher computation time (parameter set Π_5).

A second round of experiments was performed, on the PSB and RD datasets, to compare BiSH-PA against PointNet [2], a deep learning approach for point cloud categorization and segmentation. For each dataset, evaluation was repeated

TABLE III

Categorization and segmentation accuracy for BiSH-PA, PointNet and PointNet with Transfer Learning

Dataset	Metric	BiSH-PA	PointNet	PointNet-TL
PSB	Cat. (%)	71.42	72.33	**85.31**
PSB	Segm. (%)	**90.38**	79.02	88.16
RD	Cat. (%)	**100.00**	61.00	40.00
RD	Segm. (%)	**93.12**	48.69	39.95

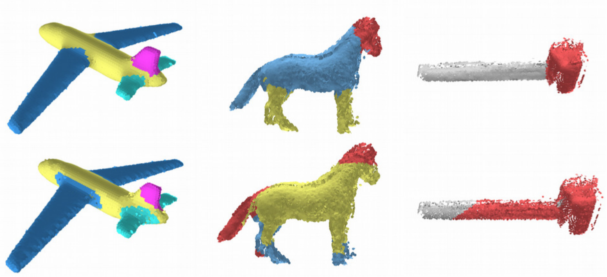

Fig. 8. Example point clouds segmented by BiSH-PA (top row) and PointNet (bottom row). BiSH-PA achieves a slightly better segmentation quality than PointNet on synthetic models of the PSB dataset (airplane model, left column), and a clearly better quality on real scans of the RD dataset (fourleg and hammer models, center and right columns).

10 times using 4-fold cross-validation, and the results were averaged. PointNet was trained for 500 epochs, with initial learning rate 0.001. We also report results achieved by PointNet trained with transfer learning ("PointNet-TL") on the default PointNet dataset (ModelNet40 [41]), that contains synthetic 3D models without noise. Categorization and segmentation accuracy are reported in Table III. Images of the resulting segmentation are displayed in Fig. 8. On the PSB dataset, BiSH-PA categorization accuracy is comparable to PointNet, which is hampered by the small size of the dataset, while PointNet-TL performed better due to transfer learning on ModelNet40, however, a large dataset is needed to achieve this result. BiSH-PA performed better than both PointNet and PointNet-TL in terms of segmentation accuracy on the PSB dataset. On the RD dataset, BiSH-PA outperformed PointNet likely because of the small dataset size and due to the lower quality of the point clouds, that are affected by the sensor noise. Moreover, we can observe that BiSH-PA also outperformed PointNet-TL as ModelNet40 contains synthetic 3D models without noise. Therefore, in real robot tasks BiSH-PA is better suited than a state of the art CNN-based algorithm like PointNet, when using a dataset of limited size.

B. Evaluation of the part-based grasping approach

The proposed viewpoint-aware filter was assessed in terms of 3D scan completeness. A total of 24 part-based grasping trials were performed, six for each object category. In each trial the workspace contained two objects that were scanned by the real robot along a pre-defined path, which allows a complete reconstruction of the environment. After that, each object point cloud cluster was categorized and segmented into parts using the RD dataset minus the point clouds that correspond to the objects currently in the workspace, to avoid bias. Then, a grasping task was planned in simulation (Fig. 9). The category $c^\star$ of the object to be grasped and the target part label $l^\star$ were provided as input to the grasping pipeline. Each trial was evaluated using four levels of 3D scan completeness, that were generated by considering the range data acquired along partial segments of the scan path (25%, 50%, 75% and 100% of the scan path respectively), as shown in Fig. 10. We defined a successful trial whenever an object of the specified category $c^\star$ was grasped from the target part $l^\star$, i.e. when the 3D model of the gripper collided with points that belong to $l^\star$. As reported in Fig. 11, the number of successful trials increased with scan completeness. Also, the success rate was higher when the viewpoint-aware filter was applied on three object categories (*jugs*, *cups* and *fourlegs*). Only three grasping failures occurred (two *jugs* and one *cup*) when the viewpoint-aware filter was applied. Grasping failures may occur for several reasons as reported in Fig. 12. A grasp fails if the object is not observed, which can happen when the scan path is incomplete, regardless of the viewpoint filter. Similarly, grasping failures caused by errors in object classification increase as the scan completeness decreases, however, they are significantly more frequent

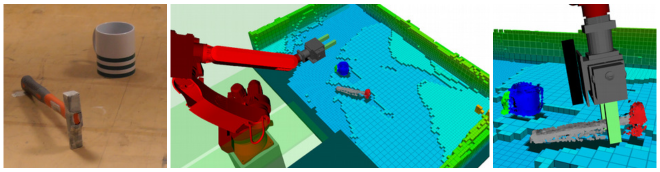

Fig. 9. Left: example scenario with two objects (a cup and a hammer). The motion planning environment (center). The planned part-based grasp of the *hammer* by its *handle* (right).

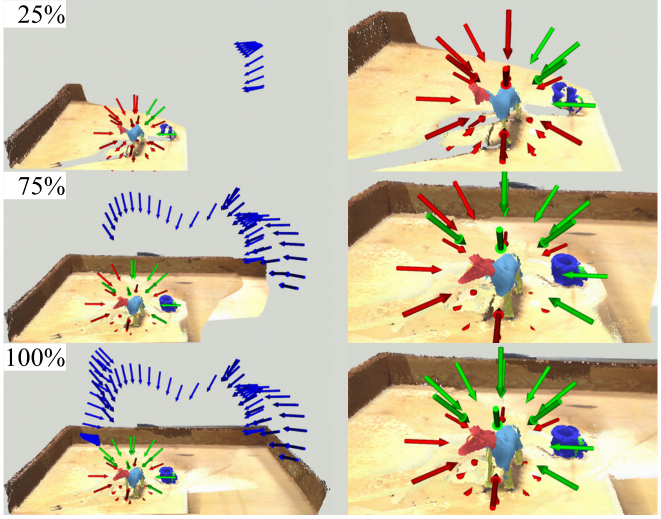

Fig. 10. Examples of virtual viewpoints for three levels of scan completeness of a scanned fourleg object (25%, 75% and 100%, top to bottom). Blue arrows are actual robot viewpoints P_i along the scan path, red and green arrows are virtual camera poses pointing towards the object (set $\mathcal{A}$). Green arrows are the virtual viewpoints compatible with the robot scan path, accepted by the viewpoint-aware filter (subset $\mathcal{B}$). The right column shows zoomed images. As the level of scan completeness increases, more virtual viewpoints (green) are accepted by the viewpoint-aware filter.

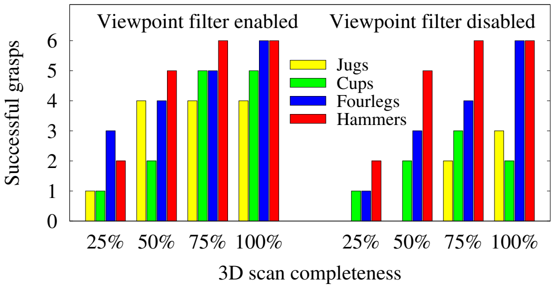

Fig. 11. Successful grasps in simulation with respect to scan completeness.

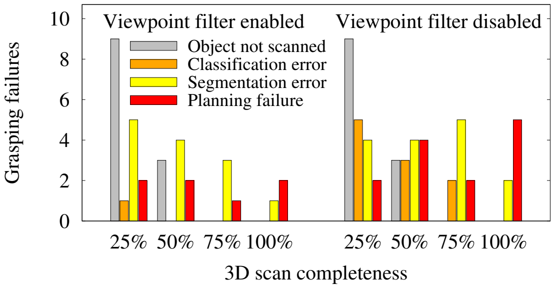

Fig. 12. Number of grasping failures when the viewpoint-aware filter is enabled (left) and disabled (right) with respect to scan completeness.

when the proposed viewpoint-aware filter is disabled. Also, failures may occur when the robot grasps the object from the wrong part, due to incorrect object segmentation, and it can be noticed that if the viewpoint-aware filter is active the performance degrades more gracefully as scan completeness decreases. Finally, failures may be generated in the grasp planning phase, or when planning the motion of the robot arm. Again, motion planning failures are more frequent when the proposed viewpoint-aware filter is disabled, possibly due to lower segmentation quality. Each unsuccessful trial was stopped as soon as the first failure occurred.

Four successful grasping experiments executed on the real robot are reported in Fig. 13. Results are also shown in the accompanying video. After grasping the target object by the selected part, the robot lifts the grasped object. Execution time for each phase of the algorithm, averaged over the four experiments, is reported in Table IV. On average, the computational time required for planning each task is about 35 s. It can be noticed that the viewpoint-aware filter requires a negligible amount of computation time (about 10 ms) compared to the other steps. A semantic object-related task was also performed in simulation, where the handle of the jug is grasped, and then the jug's neck is placed above the cup bowl, properly oriented for pouring (Fig. 14).

V. Conclusion

In this work, an approach for semantic part-based object grasping was presented based on projective analysis. The method allows object categorization, segmentation, and grasp planning without requiring any priors on part shapes. The BiSH-PA algorithm showed better performance compared to a CNN-based state of the art method based on a dataset of limited size containing real 3D scans. In order to deal with incomplete 3D scans a viewpoint-aware filter was proposed that improves segmentation and reduces grasping failures. The proposed method was evaluated in real-world scenarios.

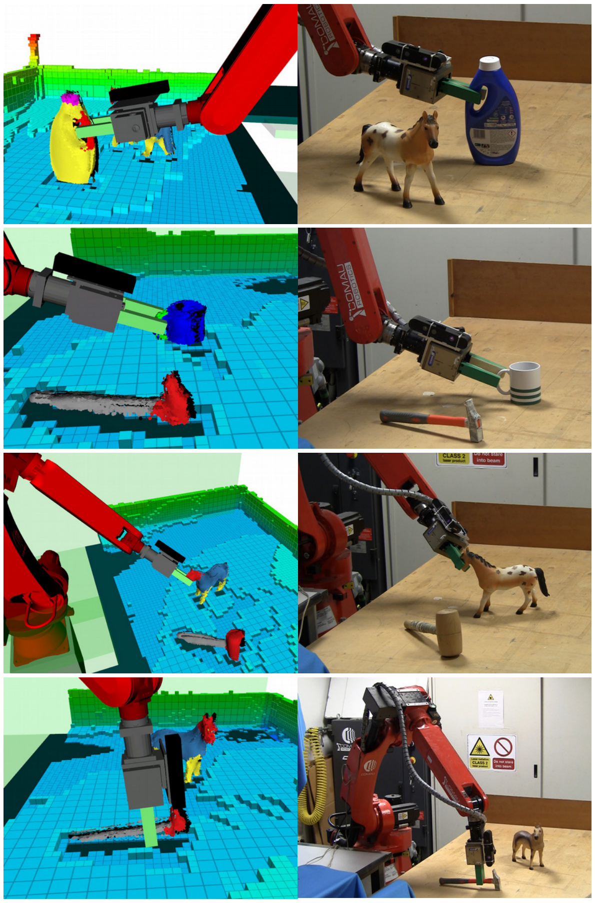

Fig. 13. Four grasping experiments carried out using the proposed part-based grasping pipeline: grasp the *jug* by the *handle*, grasp the *cup* by the *handle*, grasp the *fourleg* by the *head*, grasp the *hammer* by the *handle* (top to bottom). Images show the motion planning environment (left) and the real experimental setup (right).

TABLE IV

Average Execution Time and Standard Deviation

Phase	Time (s)
Object cluster extraction	8.35 ± 1.34
Viewpoint-aware filter	0.01 ± 0.01
Classification and segmentation (BiSH-PA)	6.41 ± 1.25
Grasp planning (GPD)	11.14 ± 4.41
Motion planning (MoveIt)	10.58 ± 3.98
Robot movement	51.54 ± 4.27
Total	88.04 ± 1.92

References

[1] Y. Wang, M. Gong, T. Wang, D. Cohen-Or, H. Zhang, and B. Chen, "Projective analysis for 3D shape segmentation," *ACM Trans. Graph.*, vol. 32, pp. 192:1–192:12, 2013.

[2] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 77–85.

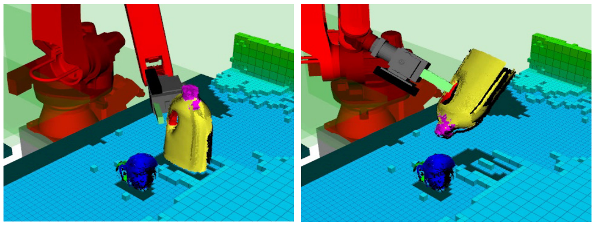

Fig. 14. A pouring experiment. The robot grasps the jug handle (left), and then it moves the jug so that the jug's neck (colored in magenta) goes above the cup bowl (blue part), properly oriented for pouring.

[3] N. Vahrenkamp, L. Westkamp, N. Yamanobe, E. E. Aksoy, and T. Asfour, "Part-based grasp planning for familiar objects," in *IEEE-RAS 16th Intl Conference on Humanoid Robots (Humanoids)*, 2016, pp. 919–925.

[4] L. Antanas, P. Moreno, M. Neumann, R. P. de Figueiredo, K. Kersting, J. Santos-Victor, and L. De Raedt, "Semantic and geometric reasoning for robotic grasping: a probabilistic logic approach," *Autonomous Robots*, vol. 43, no. 6, pp. 1393–1418, Aug 2019.

[5] J. Aleotti, D. Lodi Rizzini, and S. Caselli, "Object Categorization and Grasping by Parts from Range Scan Data," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 4190–4196.

[6] M. Madry, D. Song, and D. Kragic, "From object categories to grasp transfer using probabilistic reasoning," in *IEEE Intl Conference on Robotics and Automation (ICRA)*, 2012, pp. 1716–1723.

[7] M. Tenorth, S. Profanter, F. Balint-Benczedi, and M. Beetz, "Decomposing CAD models of objects of daily use and reasoning about their functional parts," in *IEEE/RSJ Intl Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 5943–5949.

[8] H. O. Song, M. Fritz, D. Goehring, and T. Darrell, "Learning to detect visual grasp affordance," *IEEE Transactions on Automation Science and Engineering*, vol. 13, no. 2, pp. 798–809, 2016.

[9] C. Liu, Wenliang Li, F. Sun, and J. Zhang, "Grasp planning by human experience on a variety of objects with complex geometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 511–517.

[10] M. Kokic, J. A. Stork, J. A. Haustein, and D. Kragic, "Affordance detection for task-specific grasping using deep learning," in *IEEE-RAS 17th Intl Conference on Humanoid Robotics*, 2017, pp. 91–98.

[11] F. Chu, R. Xu, and P. A. Vela, "Learning Affordance Segmentation for Real-World Robotic Manipulation via Synthetic Images," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1140–1147, 2019.

[12] D. Kanoulas, J. Lee, D. G. Caldwell, and N. G. Tsagarakis, "Visual Grasp Affordance Localization in Point Clouds Using Curved Contact Patches," *International Journal of Humanoid Robotics*, vol. 14, pp. 1–21, 2017.

[13] C. Yang, X. Lan, H. Zhang, and N. Zheng, "Task-oriented Grasping in Object Stacking Scenes with CRF-based Semantic Model," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6427–6434.

[14] D. Song, C. H. Ek, K. Huebner, and D. Kragic, "Task-based robot grasp planning using probabilistic inference," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 546–561, 2015.

[15] M. Schoeler and F. Wörgötter, "Bootstrapping the Semantics of Tools: Affordance Analysis of Real World Objects on a Per-part Basis," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 8, no. 2, pp. 84–98, 2016.

[16] R. Detry, J. Papon, and L. Matthies, "Task-oriented grasping with semantic and geometric scene understanding," in *IEEE/RSJ Intl Conference on Intelligent Robots and Systems*, 2017, pp. 3266–3273.

[17] M. Iizuka and M. Hashimoto, "Detection of Semantic Grasping-Parameter using Part-Affordance Recognition," in *Intl Conference on Research and Education in Mechatronics (REM)*, 2018, pp. 136–140.

[18] S. R. Lakani, A. J. Rodríguez-Sánchez, and J. Piater, "Towards affordance detection for robot manipulation using affordance for parts and parts for affordance," *Autonomous Robots*, vol. 43, no. 5, pp. 1155–1172, 2019.

[19] A. Myers, C. L. Teo, C. Fermüller, and Y. Aloimonos, "Affordance detection of tool parts from geometric features," in *IEEE Intl Conference on Robotics and Automation (ICRA)*, 2015, pp. 1374–1381.

[20] S. R. Lakani, A. J. Rodríguez-Sánchez, and J. Piater, "Exercising Affordances of Objects: A Part-Based Approach," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3465–3472, 2018.

[21] K. Chaudhary, K. Okada, M. Inaba, and X. Chen, "Predicting Part Affordances of Objects Using Two-Stream Fully Convolutional Network with Multimodal Inputs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3096–3101.

[22] F. Chu, R. Xu, L. Seguin, and P. A. Vela, "Toward affordance detection and ranking on novel objects for real-world robotic manipulation," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4070–4077, 2019.

[23] D. Kraft, R. Detry, N. Pugeault, E. Baseski, F. Guerin, J. H. Piater, and N. Krüger, "Development of object and grasping knowledge by robot exploration," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 4, pp. 368–383, 2010.

[24] H. Dang and P. K. Allen, "Semantic grasping: planning task-specific stable robotic grasps," *Autonomous Robots*, vol. 37, no. 3, pp. 301–316, 2014.

[25] A. Nguyen, D. Kanoulas, D. G. Caldwell, and N. G. Tsagarakis, "Object-based affordances detection with convolutional neural networks and dense conditional random fields," in *IEEE/RSJ Intl Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5908–5915.

[26] T. Do, A. Nguyen, and I. Reid, "AffordanceNet: An End-to-End Deep Learning Approach for Object Affordance Detection," in *IEEE Intl Conference on Robotics and Automation*, 2018, pp. 5882–5889.

[27] R. Detry, C. H. Ek, M. Madry, and D. Kragic, "Learning a dictionary of prototypical grasp-predicting parts from grasping experience," in *IEEE Intl Conference on Robotics and Automation (ICRA)*, 2013, pp. 601–608.

[28] R. Detry and J. Piater, "Unsupervised learning of predictive parts for cross-object grasp transfer," in *IEEE/RSJ Intl Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 1720–1727.

[29] M. Matl, J. Mahler, and K. Goldberg, "An algorithm for transferring parallel-jaw grasps between 3D mesh subsegments," in *IEEE Conference on Automation Science and Engineering*, 2017, pp. 756–763.

[30] H. Tian, C. Wang, D. Manocha, and X. Zhang, "Transferring Grasp Configurations using Active Learning and Local Replanning," in *Intl Conference on Robotics and Automation*, 2019, pp. 1622–1628.

[31] C. Goldfeder, P. K. Allen, C. Lackner, and R. Pelossof, "Grasp planning via decomposition trees," in *IEEE Intl Conference on Robotics and Automation (ICRA)*, 2007, pp. 4679–4684.

[32] K. Huebner, "BADGr-A toolbox for box-based approximation, decomposition and GRasping," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 367–376, 2012.

[33] S. El-Khoury, A. Sahbani, and V. Perdereau, "Learning the natural grasping component of an unknown object," in *IEEE/RSJ Intl Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 2957–2962.

[34] K. M. Varadarajan and M. Vincze, "Object part segmentation and classification in range images for grasping," in *2011 15th Intl Conference on Advanced Robotics (ICAR)*, 2011, pp. 21–27.

[35] Y. Shiraki, K. Nagata, N. Yamanobe, A. Nakamura, K. Harada, D. Sato, and D. N. Nenchev, "Modeling of everyday objects for semantic grasp," in *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, 2014, pp. 750–755.

[36] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[37] T. Whelan, S. Leutenegger, R. S. Moreno, B. Glocker, and A. Davison, "ElasticFusion: Dense SLAM without a pose graph," in *Proceedings of Robotics: Science and Systems*, Rome, Italy, 2015.

[38] A. ten Pas, M. Gualtieri, K. Saenko, and R. Platt, "Grasp pose detection in point clouds," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1455–1473, 2017.

[39] X. Chen, A. Golovinskiy, and T. Funkhouser, "A Benchmark for 3D Mesh Segmentation," *ACM Transactions on Graphics*, vol. 28, no. 3, 2009.

[40] R. Monica, J. Aleotti, M. Zillich, and M. Vincze, "Multi-label point cloud annotation by selection of sparse control points," in *International Conference on 3D Vision (3DV)*, 2017, pp. 301–308.

[41] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and J. Xiao, "3D ShapeNets: A Deep Representation for Volumetric Shapes," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 1912–1920.