2016 IEEE International Conference on Robotics and Automation (ICRA)
Stockholm, Sweden, May 16-21, 2016

Grasp Detection for Assistive Robotic Manipulation

Siddarth Jain and Brenna Argall

Abstract— **In this paper, we present a novel grasp detection algorithm targeted towards assistive robotic manipulation systems. We consider the problem of detecting robotic grasps using only the raw point cloud depth data of a scene containing unknown objects, and apply a geometric approach that categorizes objects into geometric shape primitives based on an analysis of local surface properties. Grasps are detected without *a priori* models, and the approach can generalize to any number of novel objects that fall within the shape primitive categories. Our approach generates multiple candidate object grasps, which moreover are semantically meaningful and similar to what a human would generate when teleoperating the robot—and thus should be suitable manipulation goals for assistive robotic systems. An evaluation of our algorithm on 30 household objects includes a pilot user study, confirms the robustness of the detected grasps and was conducted in real-world experiments using an assistive robotic arm.**

I. Introduction

There are millions of people worldwide who often have difficulties performing Activities of Daily Living (ADL). A survey [1] of people with motor impairments identified reaching, gripping and picking up objects from the shelf and floor as the most important tasks that people would like to do but cannot. Object retrieval is one of the most essential components for performing ADL. The lack of ability to retrieve objects often results in lost independence and the individual requires supervision or some other form of assistance. This causes a significant expense on the health care system and caregivers.

Assistive robotic manipulators can improve the lives of people with motor limitations by enabling them to perform pick-and-place tasks or object retrieval, or even assist with personal hygiene and feeding. However, these robotic arms have high degrees of freedom (DoF) which makes them challenging to control with the available control interfaces.

Researchers offer solutions that make the control of robotic manipulators fully or partially autonomous and propose shared-control systems in which robotics autonomy is used to aid users to perform manipulation tasks [2]–[5]. These works simplify the control problem and assume that the robot knows either the user's intent (e.g. single goal) or predefined goals in the environment (e.g. multiple objects placed at known fixed poses). Goals are the locations where a robotic gripper could be placed (*grasp poses*) to perform some sort of interaction with the object. In some works, the user specifies the intended goal using interfaces, such as a GUI [2], [3] or a laser pointer [6], or the system predicts the goal based on the users control signals and the object's position. Others use markers or fiducials on objects to facilitate object and grasp pose detection [7] and thus deal only with labeled goals. In daily life tasks and real-world scenarios, environment and goals change significantly. This paper aims to address the problem of unknown goals—grasp pose detection on novel objects—and presents a perception algorithm to autonomously detect object grasps, with a particular focus on assistive robotic manipulation systems.

Siddarth Jain and Brenna Argall are both jointly affiliated with the Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208, USA and the Rehabilitation Institute of Chicago, Chicago, IL 60611 sidd@u.northwestern.edu, brenna.argall@northwestern.edu

Fig. 1. Test set of 30 household objects used to evaluate our algorithm.

Robotic grasp detection is a challenging perception problem. Over the past decades, several vision-based algorithms have been developed that compute grasps that are stable and suitable for use by an autonomous robotic system. However, such grasps may not be *predictable* to users in a human-robot collaboration scenario. It is important that the human collaborator can infer the actual target of the robot out of many possible choices. Our approach generates multiple grasps to handle the multiple ways in which a single object can be grasped, and the grasps moreover are similar to what a human would generate when teleoperating the robot. This similarity should make them suitable to be provided as manipulation goals in shared-control systems.

The algorithm detects grasps from point cloud depth data of a scene containing unknown objects, and can handle noisy and partial views of the objects. We leverage geometric information for the automatic categorization of objects into shape primitives, and have defined strategies for grasp detection based on the primitive selection. The approach does not require an object recognition or training phase, and provides grasp detection in real-time. Our contributions are validated on 30 household objects (Fig. 1) in real-world experiments using an assistive robotic arm and includes a pilot user study.

The following section provides a review of the related literature. Section III describes the system setup, and Section IV the algorithm in detail. Experimental results are presented in Section V. In Section VI we conclude and provide directions for future research.

978-1-4673-8026-3/16/$31.00 ©2016 IEEE

II. Related Work

The problem of robot grasping has been widely studied in the literature and is an active research field. One camp address grasping using analytical approaches [8], [9] that leverage force-closure and form-closure to assure a stable grasp configuration. Such approaches assume the availability of contact points on the object, and rely on well-defined 3D models of the object, which however generally are not available in real-world scenarios. Another line of research is database-driven (for a survey, see [10]), which also assumes the availability of complete 3D models of the objects and uses a pre-computed grasp dataset [11]–[13]. Some systems use image data for grasp detection and generally require a manually-labeled grasps dataset to learn models for grasp detection [14], [15]. Recently, approaches make use of deep learning methods [16], which require large training datasets and generally are not suitable for real-time applications.

Other approaches approximate an object with shape primitives, which is similar to our work. Miller *et al.* [17] use shape primitives for grasp selection but also require known models of the object. Yamanobe and Nagata [18] assign various primitives by hand. Huebner *et al.* [19] envelop 3D data points as constellation of box primitives for object grasping by parts, however the decomposition step is time-expensive. Methods [17], [19] furthermore rely on simulation software for the grasp generation, which does not always correlate well with physical robot grasp performance.

There exist approaches that use point cloud data for grasp detection. Rusu *et al.* [20] annotate point cloud data with geometric primitive labels using geometric features and use mesh generation for grasp detection in simulation. Our work does not require learning or mesh generation and directly operates on the input point cloud data. Jain *et al.* [6] target grasping household objects with an assistive robot which does not rely on object models and detects a single overhead grasp in point clouds of objects. The approach relies on a top-view of the object (captured via laser scanner) and for the user to indicate which object using a laser pointer.

Concerning the interaction with a human, there exist works that study legible and predictable goal-directed motion (motion that communicates its intent to a human observer) from the robot [21], [22]. Research also studies human grasp postures using demonstrations [23], for human-like robotic grasping. Equally important is for the robot to have an accurate understanding of the intent of the human. Our approach strives to satisfy both of these aims by generating candidate grasps that are similar to what users would themselves generate, and by selecting between the candidate grasps based on inferred user intent. Our approach also is unique in that it generates multiple candidate grasp poses. We anticipate that in collaborative human-robot scenarios, detecting a single object grasp will be insufficient—as environment change, the human user's choice of grasp location also will vary.

III. System Description

The hardware used for implementation and testing of the proposed framework is the MICO robotic arm (Kinova Robotics, Canada), a 6-DoF manipulator with a 2 finger gripper. The MICO is the research edition of the JACO arm which is used commercially within assistive domains. The sensory input to our framework is 3D point cloud data of the scene from a Kinect RGB-D sensor (Microsoft, United States), where each point is represented by a tuple $\langle$x, y, z$\rangle$, its 3D position in camera coordinates. In the current setup, the robot is mounted on a table and the Kinect is mounted on a stand at the rear of the robotic arm, 1.5*m* above the ground (Fig. 2). The sensory input is transformed into the base frame of the robot R_f, in which the Z-axis is perpendicular to the ground (Z_R), the Y-axis is front-to-back (Y_R) and the X-axis is left-to-right (X_R). Notably, the point cloud data consists only of the points on surfaces facing the sensor—that is, it actually is a 2.5D depth data cloud—and hence does not provide complete geometric information.

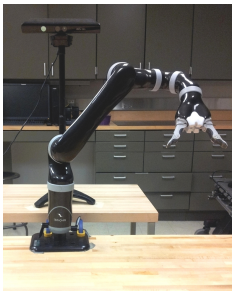

Fig. 2. MICO robot arm.

We preprocess the input point cloud data, with the goal of segmenting the scene into individual graspable objects (point cloud clusters). This involves first finding surfaces where objects could be located (and placed) for pick-and-place manipulations. (Many household objects in domestic environments are supported by flat planar surfaces like tabletops, kitchen counters, etc.). We compute a planar fit using Random Sample Consensus (RANSAC) to generate model hypotheses and extract the surface that provides support for the objects. We then find the individual objects O_j in the scene by performing Euclidean clustering on the points above the planar support.[1] Preprocessing steps then are performed on each object cluster O_j—namely, radius outlier removal to filter out sparse points which are considered to be noise and surface smoothing using Moving Least Squares (MLS). The result is object set $\boldsymbol{O}$.

IV. Grasp Detection

We now present our algorithm for grasp detection. The algorithm generates, for every observed object point cloud cluster O_j, a set of possible grasps $\mathcal{G}_{o_j}$. Each grasp $g^k_{o_j} \in \mathcal{G}_{o_j}$ is a $\mathbb{R}^{4\times4}$ transformation matrix (position and orientation of the robot gripper in base frame R_f) that can readily be used as a motion planning goal to grasp object O_j.[2]

The first step (Sec. IV-A) of the grasp detection process is to model a simplified version of the object's geometry by automated categorization to 3D geometric primitive. The geometric primitive does not need to match the object exactly, and need only approximate the 2.5D depth data cloud well enough within some error bounds. Next, for each primitive type we define a set of grasping strategies (Sec. IV-B), with the dual aims that grasps be similar to what the

[1]Note that segmentation is not the main focus of this work. The above method is chosen for its simplicity and efficacy.

[2]The remainder of this section describes operation on a single object $O_j \in \boldsymbol{O}$, and so for simplicity we drop the notation j.

human user would generate when teleoperating the robot and to limit the huge number of possible grasps on the object.

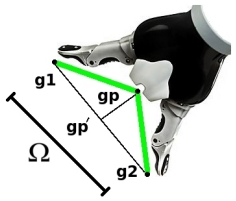

Fig. 3. Grasp points.

We represent a grasp (Fig. 3) using three points $\boldsymbol{g_p}$, $\boldsymbol{g_1}$ and $\boldsymbol{g_2}$, each in $\mathbb{R}^{3\times1}$. Point $\boldsymbol{g_p}$ provides translational information (end-effector position), and $\boldsymbol{g_1}$ and $\boldsymbol{g_2}$ provide orientation information for a $\mathbb{R}^{4\times4}$ transformation matrix. A point $\boldsymbol{g'_p}$ is calculated as an offset for the fingers. Points $\boldsymbol{g_1}$ and $\boldsymbol{g_2}$, and the offset, are specific to the robot—or more particularly, to the width Ω of the end-effector gripper—while $\boldsymbol{g_p}$ is computed solely from geometric primitive information.

A. Geometric Primitive Selection

To select the primitive which best approximates the object geometry, we analyze the surface of the point cloud cluster. In particular, we analyze the principal components of a covariance matrix created from the nearest neighbors (in a local neighborhood of size N) for each point in cluster O.

Specifically, for a given point $\boldsymbol{p}_\star \in O$, we compute the centroid $\bar{\boldsymbol{p}}$ of its N nearest neighbors,

$$\bar{\boldsymbol{p}} = \frac{\sum_{i=1}^{N} \boldsymbol{p}_i}{N}, \quad \boldsymbol{p}_i \in O \tag{1}$$

and the covariance matrix $C \in \mathbb{R}^{3\times3}$,

$$C = \frac{1}{N}\sum_{i=1}^{N} \xi_i \cdot (\boldsymbol{p}_i - \bar{\boldsymbol{p}}) \cdot (\boldsymbol{p}_i - \bar{\boldsymbol{p}})^T \tag{2}$$

where weight ξ_i for point $\boldsymbol{p}_i$ is calculated as in [24]. The eigenvalues $\lambda_j \in \mathbb{R}$ and eigenvectors $\vec{\boldsymbol{v}}_j \in \mathbb{R}^3$ are defined by the following relation,

$$C \cdot \vec{\boldsymbol{v}}_j = \lambda_j \cdot \vec{\boldsymbol{v}}_j, \;\; j \in \{0,1,2\} \tag{3}$$

and form an orthogonal frame that corresponds to the principal components of the set of neighborhood points.

The eigenvalue λ_j quantifies the variation around $\boldsymbol{p}_\star$ along the direction of eigenvector $\vec{\boldsymbol{v}}_j$. The smallest eigenvalue (λ_0) corresponds to variation along the surface normal vector $\vec{\boldsymbol{n}}$ (i.e. eigenvector $\vec{\boldsymbol{v}}_0$) of the surface patch defined by the neighborhood points.

Eigenanalysis (Principal Component Analysis) can be used to evaluate the local surface curvature around $\boldsymbol{p_\star}$. Specifically, eigenvalues λ_j of the covariance matrix C can be used as approximations of the local surface variation around $\boldsymbol{p}_\star$. The *surface variation* $\sigma_{\boldsymbol{p}_\star}$ at point $\boldsymbol{p}_\star$, calculated [25] as

$$\sigma_{\boldsymbol{p}_\star} = \frac{\lambda_0}{\lambda_0 + \lambda_1 + \lambda_2} \tag{4}$$

provides an approximation to the change in curvature.[3] The surface variation $\sigma_{\boldsymbol{p}_i}$ is calculated for every point $\boldsymbol{p}_i \in O$.

[3]Many alternative analyses of surface curvature require a mesh representation (rather than just sampled points). For example, Dyn *et. al* [26] define the Gaussian and Mean curvatures of a smooth surface using a triangular mesh, which however do not provide a good enough representation here as there is strong correlation between estimated values. Other curvature formulations such as shape index are sensitive to noise and cannot be estimated from a set of sampled points directly.

We furthermore define a metric Δ that is used to determine the best 3D geometric primitive,

$$\Delta = \frac{M^*}{M} \tag{5}$$

M^*: Number of points $\boldsymbol{p}_i \in O$ for which $\sigma_{\boldsymbol{p}_i} \leq \delta$
M : Number of points $\boldsymbol{p}_i \in O$

where δ is an empirically-determined threshold (here $\delta = 0.01$). [4] Specifically, many objects found on tables in typical household environments are cylindrical, spherical or box-like (that can be decomposed into planes). Higher values of Δ suggest that most of the object's points have low surface variation, and thus can be decomposed into planes and approximated by a box-like primitive. Small values of Δ suggest higher surface variation, and that the object can be approximated by a spherical or a cylindrical geometric primitive. The thresholds on Δ used for primitive classification (empirically determined) are presented in Table I.

TABLE I
CLASSIFICATION FOR BEST FIT PRIMITIVE

Spherical	$0.0 \leq \Delta \leq 0.10$
Cylindrical	$0.10 < \Delta \leq 0.40$
Box-like	$0.40 < \Delta \leq 1.0$

While many objects are well-represented by spherical, cylindrical and box-like geometries, many others are not—and, for example, are best represented as multiple shape primitives, or are irregularly shaped. The inability to handle such geometries is a limitation of the current algorithm, which we intend to address in future work.

B. Grasp Generation for Primitives

Next a hypothesis, of the geometric model parameters for the selected 3D primitive, is generated using RANSAC. The obtained model then is refined, using a linear least-squares fit for the plane and non-linear Levenberg-Marquardt optimization for the cylinder and sphere.

We define strategies for detecting Top, Side and/or Pinch Grasps for each geometric primitive. Our algorithm generates multiple candidate grasps—because we expect the human's preference on where to grasp the object to change with environmental context. Since the aim also is for these grasps to be easily predicted by the human operators, we limit the full (infinite) scope of possible grasps to semantic groups (e.g. left side, right side, top) and generate a single grasp for each group. Our system implementation (Sec. V) selects a single grasp for execution from the multiple candidate grasps by considering the human's teleoperation signals as an indication of their intent.

Our additional aim is for the generated grasps to be similar to what a human would generate during teleoperation. Balasubramanian *et al.* [23] show that humans tend to grasp

[4]Geometric edges skew the interpretation of Δ. Edges (points with extremely high curvature values) are identified by analyzing the distribution of curvatures $\sigma_{\boldsymbol{p}_i}$—points in the upper quartile of this distribution are not included in the Δ computation.

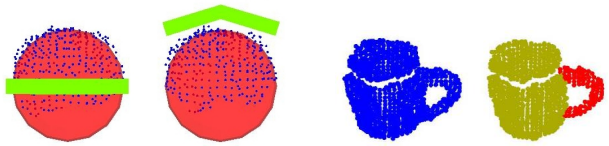

Fig. 4. *Left:* Grasp Generation for Spherical Objects. Detected grasps (green) for spherical object point cloud data (blue) and the best fit 3D shape primitive (red). *Right:* Clusters of points on the cylindrical shape surface (yellow) are marked as protrusions (red).

with wrist orientations that are orthogonal to the object's principal axis and its perpendiculars. Such grasps also generally are more stable than those that do not value orthogonality (e.g. generated using randomized planners). Thus, we design our strategies to incorporate orthogonality by generating grasps along the object centroid, whenever possible, which also minimizes torque during grasp execution.

In the following subsections, $\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{z}}$ refer, respectively, to unit vectors along the base frame axes X_R, Y_R and Z_R (Sec. III).

1) Grasp Generation for Spherical Objects: The shape parameters for the spherical primitive consist of the sphere center $\boldsymbol{s_c}$ and radius s_r. We compute a Top Grasp and a Side Grasp (Pinch is not feasible) by calculating $\boldsymbol{g_p}$, $\boldsymbol{g_1}$ and $\boldsymbol{g_2}$ (Fig. 3), as described in Table II. Figure 4 (left) shows the grasps (green) generated for spherical objects.

TABLE II
GRASP GENERATION FOR SPHERICAL OBJECTS

Side Grasp	*Top Grasp*
$\boldsymbol{g_p} = \boldsymbol{s_c} + s_r \cdot \hat{\boldsymbol{y}}$	$\boldsymbol{g_p} = \boldsymbol{s_c} + s_r \cdot \hat{\boldsymbol{z}}$
$\boldsymbol{g_{1,2}} = \boldsymbol{g'_p} \pm \frac{\Omega}{2} \cdot \hat{\boldsymbol{x}}$	$\boldsymbol{g_{1,2}} = \boldsymbol{g'_p} \pm \frac{\Omega}{2} \cdot \hat{\boldsymbol{x}}$

2) Grasp Generation for Cylindrical Objects: The shape parameters for the cylindrical primitive consist of the axis vector $\vec{\boldsymbol{c}}_{\boldsymbol{a}}$, a point on the cylinder axis $\boldsymbol{c_p}$ and the cylinder's radius c_r. We project the inliers of the cylindrical model onto the axis vector and then compute the mean of the projected points to determine the centroid $\boldsymbol{c_c}$ of the cylinder. The height c_h of the cylindrical object is determined by computing the minimum and maximum bounds of the projected points along the axis $\vec{\boldsymbol{c}}_{\boldsymbol{a}}$. An illustration of this approach for different cylindrical objects is presented in Figure 5.

Next, we look for possible spurious protrusions (including handles) from the object. Protrusions may hinder the grasp execution, and moreover in the case of handles we anticipate a potential handover to human partners. Therefore, objects are grasped on the opposite side of a protrusion. We employ a model-free approach to find protrusions rather than fitting specific geometric shapes—since the geometry varies for different kinds of objects, and we are grasping on the opposite side in any case. We examine the vicinity of the cylinder model's surface for clusters of points, and mark such

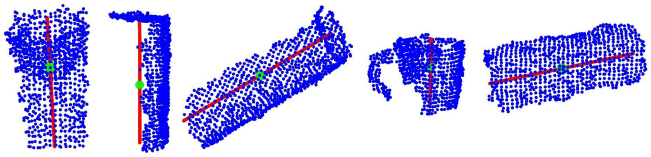

Fig. 5. Cylindrical objects center (green) and height (red line) computation.

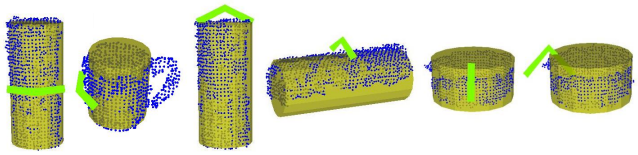

Fig. 6. Grasp Generation for Cylindrical Objects. Detected grasps (green) for cylindrical object point cloud data (blue) and the best fit 3D shape primitive (yellow).

clusters as protrusions (Fig. 4, right). We also determine if the cylindrical object is in an upright or lateral position.

Points $\boldsymbol{g_1}, \boldsymbol{g_2}$ and $\boldsymbol{g_p}$ then are computed as described in Table III, where $\hat{\boldsymbol{c}}_{\perp}$ is a unit vector that is perpendicular to both axis vector $\vec{\boldsymbol{c}}_{\boldsymbol{a}}$ and $\hat{\boldsymbol{z}}$.

In detail, if the cylinder is in a *lateral position* we compute a Top Grasp only. If it is *upright* and with *no protrusion*, we compute Side and Top Grasps. For upright cylinders *with a protrusion*, the Side Grasp is computed on the opposite side of the protrusion. To do this, we find direction vector $\vec{\boldsymbol{c}}_{\boldsymbol{pr}}$ along the protrusion cluster centroid and its projection on the cylinder axis. We then calculate unit vector $\hat{\boldsymbol{c}}_{\perp pr}$, that is perpendicular to both $\vec{\boldsymbol{c}}_{\boldsymbol{pr}}$ and axis vector $\vec{\boldsymbol{c}}_{\boldsymbol{a}}$.

For the case in which the radius $c_r > \Omega/2$, Side and Top Grasps are not physically feasible for the robot gripper. The algorithm instead generates a Pinch Grasp.

Figure 6 shows the various grasps for cylindrical objects.

3) Grasp Generation for Box-like Objects: For box-like objects we model planar components. Specifically, the algorithm performs principal component analysis on the point cloud cluster O to obtain the eigenvectors $\vec{\boldsymbol{v}}_0$ and $\vec{\boldsymbol{v}}_1$, and the centroid $\boldsymbol{b_c}$. Next, the object is decomposed into planar components. We use RANSAC to extract planar models Π from cluster O iteratively, until the remaining number of points in O falls below threshold ρ (here set to 10% of the initial cluster size of O).

For each of the identified planes we compute its centroid $\boldsymbol{p_c}$ and the normal vector $\vec{\boldsymbol{n}}_{\boldsymbol{p}}$. Next, we find planes whose normals are in the direction of the major eigenvector $\vec{\boldsymbol{v}}_1$ to generate Side Grasps on the object and the planes whose normal is in the direction of $\hat{\boldsymbol{z}}$ to generate Top Grasp. In both cases, we compute $\boldsymbol{g_p}$, $\boldsymbol{g_1}$ and $\boldsymbol{g_2}$ for grasp generation as described in Table IV. Figure 7 shows the various grasps for box-like objects.

TABLE IV
GRASP GENERATION FOR BOX-LIKE OBJECTS

$\boldsymbol{g_p} = \boldsymbol{p_c}$
$\boldsymbol{g_{1,2}} = \boldsymbol{g'_p} \pm \frac{\Omega}{2} \cdot \hat{\boldsymbol{v}}_0$

TABLE III

GRASP GENERATION FOR CYLINDRICAL OBJECTS

Side Grasp (upright)	*Top Grasp (upright)*	*Top Grasp (lateral)*	*Side Grasp (protrusion)*	*Pinch Grasp*
$\boldsymbol{g_p} = \boldsymbol{c_c} + c_r \cdot \hat{\boldsymbol{y}}$	$\boldsymbol{g_p} = \boldsymbol{c_c} + \frac{c_h}{2} \cdot \hat{\boldsymbol{z}}$	$\boldsymbol{g_p} = \boldsymbol{c_c} + c_r \cdot \hat{\boldsymbol{z}}$	$\boldsymbol{g_p} = \boldsymbol{c_c} + c_r \cdot \hat{\boldsymbol{c}}_{pr}$	$\boldsymbol{g_p} = \boldsymbol{c_c} + c_r \cdot \hat{\boldsymbol{y}} + \frac{c_h}{2} \cdot \hat{\boldsymbol{z}}$
$\boldsymbol{g_{1,2}} = \boldsymbol{g'_p} \pm \frac{\Omega}{2} \cdot \hat{\boldsymbol{x}}$	$\boldsymbol{g_{1,2}} = \boldsymbol{g'_p} \pm \frac{\Omega}{2} \cdot \hat{\boldsymbol{x}}$	$\boldsymbol{g_{1,2}} = \boldsymbol{g'_p} \pm \frac{\Omega}{2} \cdot \hat{\boldsymbol{c}}_{\perp}$	$\boldsymbol{g_{1,2}} = \boldsymbol{g'_p} \pm \frac{\Omega}{2} \cdot \hat{\boldsymbol{c}}_{\perp pr}$	$\boldsymbol{g_{1,2}} = \boldsymbol{g'_p} \pm \frac{\Omega}{2} \cdot \hat{\boldsymbol{y}}$

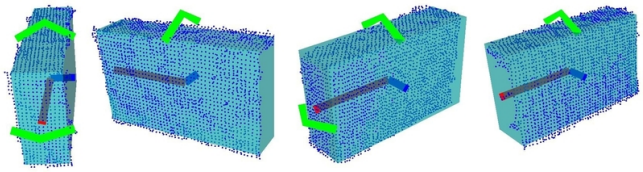

Fig. 7. Grasp Generation for Box-like objects. Detected grasps (green) for box-like object point cloud data (blue) and the best fit 3D shape primitive (cyan).

V. EVALUATION AND RESULTS

We performed three experiments to characterize the performance of our algorithm. Our test set consisted of 30 household objects that cover various categories, shapes, textures and sizes of objects (Fig. 1).[5] More importantly, most object categories in the test set belong to the list of important objects identified for robotic retrieval tasks by motor-impaired patients [27]. Objects were presented in multiple *pose configurations*, consisting of 0°, 45°, -45°, 90° offsets. Experiments were carried out on an Intel Core i7 Quad-Core processor PC with 12GB of RAM and running Ubuntu 12.04, the Robot Operating System (ROS Hydro) and Point Cloud Library (PCL 1.7). Grasp executions were carried out by the MICO robotic arm (6+1 DoF).

Shape Primitive Selection: To evaluate shape primitive selection (Sec. IV-A), we ran the algorithm on each object in the test set, with all test pose configurations and lateral and upright positions when possible. Each pose configuration was evaluated for 10 runs of the algorithm. The algorithm selected the best shape primitive for the objects with very few errors. Table V summarizes the results.[6]

Grasp Similarity to Human Teleoperated Grasp: To evaluate the similarity between the algorithm-generated grasps to those generated by a human when teleoperating the robot, a pilot study was performed in which three users (who were all lab members) teleoperated the MICO robot using a 3-axis joystick interface to grasp the test objects. The test objects were presented one by one on a table in front of the robot in one of the above pose configurations (selection was random and balanced). The users were not provided with any information about the algorithmic approach and were instructed to teleoperate the robot to grasp the object in a way they thought would be stable enough to lift the object from the table.

Before each trial the algorithm computed a set $\mathcal{G}$ of candidate grasps for the presented object O. As the user teleoperated the robot, the algorithm maintained a confidence score for each grasp $g \in \mathcal{G}$, based on: (1) the Euclidean distance between the robot end-effector position and the grasp position, and (2) the end-effector orientation (roll, pitch and yaw) alignment with the grasp pose.

The most confident grasp g^* generated by the algorithm was then compared to the human-selected grasp g_h, defined as the final robot end-effector pose on the object with the robot gripper closed. For each trial we calculated the difference between poses g^* and g_h—as the difference in roll δR, pitch δP and yaw δY angles (in degrees), and the Euclidean distance δT in position (in cm).

Table VI provides results for each of the trials, and also reports whether the human selected a Side (S), Top (T) or

TABLE V

EVALUATION OF BEST SHAPE PRIMITIVE FIT ON THE TEST OBJECTS

Key: #: Number of. %: Percent correct.

Object	#Pose	#Cylinder	#Sphere	#Box-like	%
Al. Foil box	40	10	0	30	75
Apple (green)	10	0	10	0	100
Apple (red)	10	0	10	0	100
Band-aid box	40	2	0	38	95
Cereals box	40	0	0	40	100
Ceramic bowl	10	10	0	0	100
Chocolate bar	40	30	0	10	100
Chocolate box	40	0	0	40	100
Coffeemate	50	50	0	0	100
Cookies box	40	2	0	38	95
Flashlight	40	40	0	0	100
Glass1	50	50	0	0	100
Glass2	50	50	0	0	100
Hairbrush1	40	40	0	0	100
Hairbrush2	40	40	0	0	100
Mug1	40	40	0	0	100
Mug2	40	40	0	0	100
Mustard	50	50	0	0	100
Orange	10	0	10	0	100
Plastic bowl	10	10	0	0	100
Pringles can	50	47	0	3	94
Ritz box	40	10	0	30	75
Soap	40	2	0	38	95
Soup can	50	49	0	1	98
Tape	10	10	0	0	100
Tennis ball	10	0	10	0	100
Toothpaste	40	30	0	10	75
Travel mug1	50	50	0	0	100
Travel mug2	40	38	2	0	95
Wafers box	40	0	0	40	100

[5]Five of the test set objects—glass (green), mug (orange), tennis ball, coffeemate, plastic bowl—also were used in development of the algorithm (along with a box-like object not in the test set). The rest of the 25 objects in the test set were novel to the algorithm.

[6]For the purpose of grasp detection, long narrow box-like objects were represented equally well by cylindrical and box-like primitives.

TABLE VI

COMPARISON OF ALGORITHM GRASP POSE AND USER-TELEOPERATED GRASP POSE (SEE TEXT FOR DETAILS)

Object	User 1					User 2					User 3				
	g	δR°	δP°	δY°	δTcm	g	δR°	δP°	δY°	δTcm	g	δR°	δP°	δY°	δTcm
Al. Foil box	T	11.45	-13.75	9.64	3.84	T	0.52	-6.88	-9.74	1.55	T	12.61	-7.45	13.18	5.19
Apple (green)	S	-11.28	9.74	-3.56	3.20	T	-1.72	4.01	-1.81	4.26	S	6.83	7.85	17.85	2.37
Apple (red)	S	-13.01	2.86	-9.83	2.66	S	9.12	1.83	2.55	5.04	T	3.08	10.31	24.21	6.23
Band-aid box	T	3.44	-5.16	6.21	1.92	T	-0.86	-1.88	-2.66	3.56	T	4.58	-7.96	8.03	4.58
Cereals box	S	-1.71	9.74	15.5	4.47	S	10.27	1.32	-3.59	5.52	S	2.48	10.89	5.24	9.06
Ceramic bowl	P	9.17	-2.29	40.83	10.21	P	1.72	0.92	-11.99	3.06	P	-1.72	3.70	4.67	5.22
Chocolate bar	T	4.01	-9.17	-6.63	1.17	S	-8.59	-2.86	0.34	3.71	T	-8.59	-3.15	4.09	8.53
Chocolate box	T	13.75	-5.16	-3.92	5.46	T	-3.44	-12.03	2.38	3.53	T	2.29	-4.41	12.09	4.67
Coffeemate	T	8.02	-8.50	1.27	5.21	T	9.17	-3.50	-1.14	4.20	S	-1.82	8.94	25.5	5.09
Cookies box	T	0.17	-5.73	-6.80	3.83	T	-6.30	-8.59	-0.61	5.61	T	-15.47	-8.02	-0.89	7.06
Flashlight	T	6.30	-4.58	3.63	4.84	T	10.89	-12.03	-3.97	4.66	T	0.52	-3.44	5.39	3.44
Glass1	S	3.97	9.74	-22.67	2.48	S	4.42	7.91	-7.54	3.76	S	3.79	12.61	33.06	4.90
Glass2	S	0.53	8.59	-12.81	5.73	S	5.11	6.88	23.01	4.74	S	2.43	6.44	21.51	3.71
Hairbrush1	T	-6.47	-2.29	-6.04	3.54	T	5.16	-8.59	-12.28	3.90	T	-10.31	-11.46	3.43	3.36
Hairbrush2	T	6.30	-13.75	-15.10	4.69	T	-11.46	-2.29	13.36	6.39	T	2.92	1.60	-8.51	4.52
Mug1	S	5.68	10.88	21.1	4.00	T	0.00	-3.67	44.61	4.11	S	3.39	2.86	13.4	3.32
Mug2	S	0.47	6.88	12.43	3.26	S	3.97	7.45	7.48	1.62	S	-0.96	6.88	30.26	4.19
Mustard	S	0.87	18.33	37.57	4.91	S	8.50	6.88	-3.07	3.58	S	7.40	6.53	23.70	2.59
Orange	S	3.39	4.58	2.78	5.28	T	26.67	-4.58	7.83	2.56	S	-3.71	12.60	22.32	6.04
Plastic bowl	P	22.92	-1.03	13.25	10.80	P	12.03	0.86	9.85	2.32	P	-2.80	-9.74	20.66	7.36
Pringles can	S	10.30	1.71	5.14	12.71	T	16.04	-11.46	1.75	3.22	T	3.21	-1.72	4.13	3.39
Ritz box	T	-4.01	1.95	-0.09	4.12	T	-10.31	-6.53	-6.56	6.17	T	-5.16	-6.30	3.35	4.62
Soap	T	13.18	1.15	7.36	1.34	T	-8.02	-9.74	-4.95	3.62	T	-15.47	-7.45	-9.13	3.54
Soup can	S	25.7	-9.74	20.20	7.38	S	4.54	4.76	-11.32	3.35	S	-2.68	-0.40	36.90	4.67
Tape	P	0.57	-2.86	-12.24	10.71	P	14.47	-7.45	-9.72	7.15	P	-1.49	-7.33	-6.39	4.27
Tennis ball	S	-1.80	9.16	10.64	2.66	S	-13.62	-4.58	-8.46	3.55	T	20.36	9.17	12.01	3.35
Toothpaste	T	6.30	-4.58	3.15	4.33	T	4.01	-5.73	2.78	3.04	T	10.89	-4.30	1.58	3.79
Travel mug1	S	1.10	5.16	23.3	5.93	S	3.97	8.02	5.13	3.77	S	-1.19	10.66	31.34	3.92
Travel mug2	S	-3.08	28.6	-21.2	3.06	T	4.54	4.24	25.24	4.97	S	3.97	9.74	20.16	4.82
Wafers box	T	8.59	5.39	-1.72	7.68	T	-6.88	5.16	2.61	5.20	T	9.91	-2.29	15.33	4.41
Mean		4.16	1.53	3.71	5.04		2.79	-1.73	1.65	4.05		0.97	1.17	12.94	4.74
$\pm$ S.E.		1.54	1.71	2.78	0.52		1.65	1.17	2.24	0.24		1.39	1.40	2.27	0.29

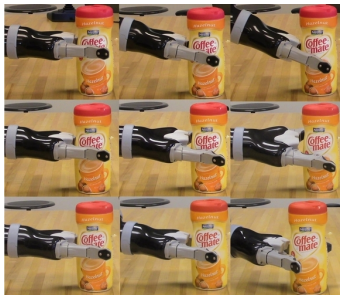

Fig. 8. Visualization of δR (top row), δP (middle row) and δY (bottom row) for 0°, 10° and 20° (left to right).

Pinch (P) Grasp. Across all users and objects (90 trials), the values for δR (2.64° $\pm$ 0.88), δP (0.32° $\pm$ 0.84), δY (6.10° $\pm$ 1.49) and δT (4.61cm $\pm$ 0.21) were quite small. To help qualify *small*, Figure 8 provides a visualization of δR, δP and δY at 0°, 10° and 20°.

These results highlight the algorithm's capability to generates grasps that are similar in pose (both orientation and position) to what a human user generates when teleoperating the assistive robot. This similarity may help the human to predict the robot's target during shared-control executions in collaborative tasks.

Grasp Robustness: Lastly, we evaluated our algorithm in experiments with the robot autonomy grasping the test set household objects, again using the MICO robot. Each object from the test set was placed alone on a table within the reach of the robot in one of the pose configurations. The algorithm computed grasps $\mathcal{G}$ for the presented object, and each was executed using the manipulation planning framework [28] that allows the robot to plan in the presence of constraints on end-effector pose using the Task Space Regions (TSRs) representation. Success was defined by grasping the object, lifting it off the table and holding it in air stably for 60 seconds. There were 50 trials in total for the experiment and the robot achieved a success rate of 82%. Figure 9 shows the resulting grasps on a sample of test objects.

VI. CONCLUSIONS AND FUTURE WORK

We have presented an algorithm for detecting robotic grasps from point cloud depth data. Our method has several advantages over existing approaches for grasp detection as it does not require known models of the object or a predefined grasp dataset, and can be used when sensor data is available for only part of the object. Furthermore, the algorithm is independent of an object recognition or training phase. The system can be used in real-world and novel environments, without requiring expert intervention. Our target application is to provide assistance in shared-control robot domains. The results have shown that the approach is capable of detecting human-like grasps for a wide range of unseen household objects, and the detected grasps also were used successfully for autonomous grasping by the MICO robotic arm. Future work will involve the extension of the shape primitive library, the decomposition of objects into multiple primitives and an evaluation within a shared-control manipulation framework that includes a large-scale user study.

Fig. 9. A sample of the grasps generated by the algorithm and executed by the MICO robot.

ACKNOWLEDGMENT

Research reported in this publication was supported by the National Institute of Biomedical Imaging and Bioengineering (NIBIB) and National Institute of Child and Human Development (NICHD) under award number R01EB019335. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

REFERENCES

[1] C. Stanger, C. Anglin, W. S. Harwin, D. P. Romilly, *et al.*, "Devices for assisting manipulation: a summary of user task priorities," *IEEE Transactions on Rehabilitation Engineering*, vol. 2, no. 4, pp. 256–265, 1994.

[2] M. Ciocarlie, K. Hsiao, A. Leeper, and D. Gossow, "Mobile manipulation through an assistive home robot," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[3] T. L. Chen, M. Ciocarlie, S. Cousins, P. M. Grice, K. Hawkins, K. Hsiao, C. C. Kemp, C.-H. King, D. A. Lazewatsky, H. Nguyen, *et al.*, "Robots for humanity: A case study in assistive mobile manipulation," 2013.

[4] S. Jain, A. Farshchiansadegh, A. Broad, F. Abdollahi, F. Mussa-Ivaldi, and B. Argall, "Assistive robotic manipulation through shared autonomy and a body-machine interface," in *Proceeding of IEEE International Conference on Rehabilitation Robotics (ICORR)*, 2015.

[5] K. M. Tsui, D.-J. Kim, A. Behal, D. Kontak, and H. A. Yanco, "I want that: Human-in-the-loop control of a wheelchair-mounted robotic arm," *Applied Bionics and Biomechanics*, vol. 8, no. 1, pp. 127–147, 2011.

[6] A. Jain and C. C. Kemp, "El-E: an assistive mobile manipulator that autonomously fetches objects from flat surfaces," *Autonomous Robots*, vol. 28, no. 1, pp. 45–64, 2010.

[7] E. A. Sisbot, R. Ros, and R. Alami, "Situation assessment for human-robot interactive object manipulation," in *Proceedings of IEEE RO-MAN*, 2011.

[8] A. Bicchi, "On the closure properties of robotic grasping," *The International Journal of Robotics Research*, vol. 14, no. 4, pp. 319–334, 1995.

[9] W. S. Howard and V. Kumar, "On the stability of grasped objects," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 6, pp. 904–917, 1996.

[10] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis-a survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, 2014.

[11] P. Brook, M. Ciocarlie, and K. Hsiao, "Collaborative grasp planning with multiple object representations," in *Proceeding of IEEE International Conference onRobotics and Automation (ICRA)*, 2011.

[12] C. Goldfeder, M. Ciocarlie, J. Peretzman, H. Dang, and P. K. Allen, "Data-driven grasping with partial sensor data," in *Proceeding of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2009.

[13] S. El-Khoury and A. Sahbani, "Handling objects by their handles," in *Proceeding of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2008.

[14] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2008.

[15] B. Kehoe, A. Matsukawa, S. Candido, J. Kuffner, and K. Goldberg, "Cloud-based robot grasping with the google object recognition engine," in *Proceeding of IEEE International Conference onRobotics and Automation (ICRA)*, 2013.

[16] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *The International Journal of Robotics Research*, vol. 34, no. 4-5, pp. 705–724, 2015.

[17] A. T. Miller, S. Knoop, H. Christensen, P. K. Allen, *et al.*, "Automatic grasp planning using shape primitives," in *Proceeding of IEEE International Conference onRobotics and Automation (ICRA)*, 2003.

[18] N. Yamanobe and K. Nagata, "Grasp planning for everyday objects based on primitive shape representation for parallel jaw grippers," in *Proceeding of IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2010.

[19] K. Huebner and D. Kragic, "Selection of robot pre-grasps using box-based shape approximation," in *Proceeding of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS*, 2008.

[20] R. B. Rusu, A. Holzbach, R. Diankov, G. Bradski, and M. Beetz, "Perception for mobile manipulation and grasping using active stereo," in *Proceeding of IEEE-RAS International Conference on Humanoid Robots*, 2009.

[21] A. D. Dragan, K. C. Lee, and S. S. Srinivasa, "Legibility and predictability of robot motion," in *Proceeding of ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2013.

[22] M. Zhao, R. Shome, I. Yochelson, K. Bekris, and E. Kowler, "An experimental study for identifying features of legible manipulator paths," in *Proceeding of International Symposium on Experimental Robotics (ISER)*, 2014.

[23] R. Balasubramanian, L. Xu, P. D. Brook, J. R. Smith, and Y. Matsuok, "Human-guided grasp measures improve grasp robustness on physical robot," in *Proceeding of IEEE International Conference on Robotics and Automation (ICRA)*, 2010.

[24] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz, "Towards 3d point cloud based object maps for household environments," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 927–941, 2008.

[25] M. Pauly, M. Gross, and L. P. Kobbelt, "Efficient simplification of point-sampled surfaces," in *Proceedings of the Conference on Visualization*, 2002.

[26] N. Dyn, K. Hormann, S.-J. Kim, and D. Levin, "Optimizing 3d triangulations using discrete curvature analysis," *Mathematical methods for curves and surfaces*, vol. 28, no. 5, pp. 135–146, 2001.

[27] Y. S. Choi, T. Deyle, T. Chen, J. D. Glass, and C. C. Kemp, "A list of household objects for robotic retrieval prioritized by people with als," in *Proceeding of IEEE International Conference on Rehabilitation Robotics (ICORR)*, 2009.

[28] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *International Journal of Robotics Research (IJRR)*, vol. 30, no. 12, pp. 1435 – 1460, 2011.