Auton Robot
DOI 10.1007/s10514-014-9391-2

Semantic grasping: planning task-specific stable robotic grasps

Hao Dang · Peter K. Allen

Received: 1 March 2013 / Accepted: 11 May 2014
© Springer Science+Business Media New York 2014

Abstract We present an example-based planning framework to generate *semantic grasps*, stable grasps that are functionally suitable for specific object manipulation tasks. We propose to use partial object geometry, tactile contacts, and hand kinematic data as proxies to encode task-related constraints, which we call *semantic constraints*. We introduce a *semantic affordance map*, which relates local geometry to a set of predefined semantic grasps that are appropriate to different tasks. Using this map, the pose of a robot hand with respect to the object can be estimated so that the hand is adjusted to achieve the ideal approach direction required by a particular task. A grasp planner is then used to search along this approach direction and generate a set of final grasps which have appropriate stability, tactile contacts, and hand kinematics. We show experiments planning semantic grasps on everyday objects and applying these grasps with a physical robot.

Keywords Grasp planning · Task-specific grasping · Semantic grasping

1 Introduction

Grasp planning is a fundamental problem in the field of robotics which has been attracting an increasing number of researchers. Previously proposed methods are reasonably successful in generating stable grasps for execution. However, there is still a gap between "stable robotic grasping" and "successful robotic grasping for object manipulation". If we consider planning a grasp for a specific manipulation task, the stability of the grasp is no longer sufficient to describe all of the constraints on the grasp. Different object manipulation tasks could impose extra constraints on the grasp to be used. Such constraints include relative hand orientation with respect to the object, specific object parts the hand should make contact with, or specific regions of the object the hand should avoid. We call these constraints required by a specific task *semantic constraints*. As is shown in Fig. 1 and Sect. 3, a good robotic grasp should satisfy the semantic constraints associated with an intended manipulation task.

In our work, we take an example-based approach to build a grasp planner that considers semantic constraints of specific tasks as a planning criterion and searches for stable grasps satisfying these semantic constraints. This approach is inspired by psychological research which showed that human grasping is to a very large extent guided by previous grasping experience Castiello (2005). To mimic this process, we propose that semantic constraints can be embedded into a database which includes partial object geometry, hand kinematics, and tactile contacts. We design a semantic affordance map which contains a set of depth images from different views of an object and predefined example grasps that satisfy semantic constraints of different tasks. These depth images help infer the approach direction of a robot hand with respect to an object, guiding the hand to an ideal approach direction. Predefined example grasps provide hand kinematic and tactile information to the planner as references to the ideal hand posture and tactile contact formation. Utilizing this information, our planner searches for stable grasps with an ideal approach direction, hand kinematics, and tactile contact formation.

In Sect. 2, we discuss related work. In Sect. 3, we propose a definition of semantic grasps and semantic affordance

H. Dang (✉) · P. K. Allen
Computer Science Department, Columbia University, New York, NY 10027, USA
e-mail: dang@cs.columbia.edu

P. K. Allen
e-mail: allen@cs.columbia.edu

Published online: 27 May 2014

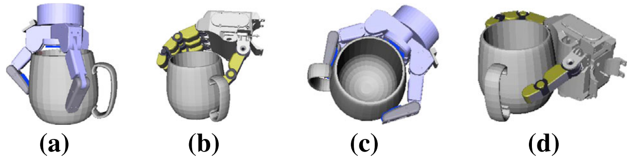


Fig. 1 Stable grasps with different semantic constraints satisfied. All grasps are suitable for *Pick-and-Place* tasks. However, for grasps in **a**, **b**, the hand blocks the opening area of the mug. Thus, they are not suitable for *Pour-Water* tasks

maps. Detailed discussions and explanations of the planning framework are presented in Sects. 4 and 5. Experiments are done on several everyday objects and they are summarized in Sect. 6, followed by discussions and conclusions in Sect. 7 and 8. Part of this work has been previously presented in Dang and Allen (2012). Compared to the published paper, this article contains more detailed explanation of our algorithm and experiments.

2 Related work

There has been previous work on the problem of planning stable robotic grasps. Ciocarlie and Allen (2009) proposed the eigen-grasp idea. This method effectively reduces the dimension of the search space for grasp planning and results in a faster search process for form-closure grasps. Based on this approach, a data-driven grasping pipeline is proposed by Goldfeder and Allen (2011) to transfer stable grasps between similar objects. Geidenstam approximated 3D shapes with bounding boxes on decomposed objects and trained a neural network to learn good grasps Geidenstam et al. (2009). Kehoe et al. (2013) utilized cloud-based computing techniques to facilitate 3D robot grasping and implemented a novel robot grasping pipeline. Berenson and Srinivasa (2008) and Berenson et al. (2007) proposed methods to generate collision-free stable grasps for dexterous hands in cluttered environments. Saxena et al. (2007) and Popovic et al. (2010) used synthesized image data to train a classifier to predict grasping points based on features extracted from 2D images. Boularias et al. (2011) proposed a probabilistic approach based on Markov Random Field for learning to grasp objects. Dogar and Srinivasa (2011) introduced a framework for planning grasping strategies using a set of actions derived analytically from the mechanics of pushing. Ben Amor et al. (2008) proposed a novel data-driven method to synthesize natural looking grasps. All the work above focuses on the stability of a grasp without considering the suitability of the grasp for a specific robotic task.

Compared to the previous work above, our work on grasp planning takes into consideration not only the stability of a grasp but also its suitability to a specific task. Along this direction, there has also been previous work in planning grasps considering some specific task requirements. Researchers, such as Li and Sastry (1987), Prats et al. (2007), and Haschke et al. (2005), analyzed task-specific grasping using task wrench space. These approaches are mainly based on the analysis of contacts and potential wrench space of a grasp, since optimal grasp wrench space could improve the performance of object manipulation tasks that require dynamic stability as indicated by Manis and Santos (2011a,b). However, some geometry related constraints cannot be guaranteed by the analysis of grasp wrench space.

In our work, we take a more explicit approach to represent these constraints. In previous work, Rosales et al. (2010) presented a method to solve the configuration problem of a robot hand to grasp a given object with a specific contact region. Ying et al. (2007) took a data-driven approach to grasp synthesis using pre-captured human grasps and task-based pruning. Song et al. (2010) designed a Bayesian network to model task constraints in goal-oriented grasping. Gioioso et al. (2012) introduced an object-based approach to map human hand synergies to robot hands with different hand kinematics focusing on specific tasks.

In addition, some researchers have been trying to analyze graspable parts for semantic grasping. Detry et al. (2010) developed a method to analyze grasp affordance on objects based on object-edge reconstructions. Hillenbrand and Roa (2012) designed an algorithm to transfer functional grasps through contact warping and local replanning. Ben Amor et al. (2012) developed an imitation learning approach for learning and generalizing grasping skills based on human demonstrations. Aleotti and Caselli (2011, 2012) proposed a part-based planning algorithm to generate stable grasps. Sahbani and El-Khoury (2009) proposed a method to plan grasps on handles of objects by training a classifier on synthesized data. Varadarajan and Vincze (2011a,b) used proposed methods to recognize functional parts of objects for robotic grasping.

It is widely accepted that many everyday objects are designed with a graspable part. However, with certain knowledge of where the graspable part is, it is still difficult to determine how to grasp the graspable part appropriately. In fact, looking solely at the stability of the grasp, the potential task wrench space, or the location of the graspable part of an object leaves out some important semantic constraints for object grasping in the context of a specific manipulation task, which include (1) how a robot hand is oriented and shaped with respect to the object and (2) the locations of the contacts established between the hand and the object. Different from the work above, we will take these aspects into consideration.

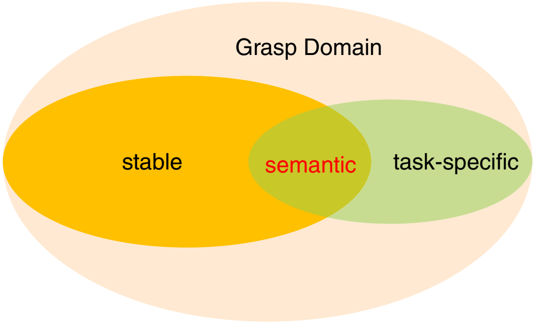


Fig. 2 A topological categorization of the domain of robotic grasps [a robotic grasp is specified by a combination of (i) position and orientation of a robot hand with respect to an object and (ii) joint angles of a robot hand. Robot arm kinematics is not considered]. There is a subset of grasps which are stable. There is also a subset of grasps which are task-specific. Task-specific grasps may or may not be stable. The intersection of the stable grasp domain and the task-specific task domain is a subset of grasps that are both stable and task-specific, which we call semantic grasps. In this work, we focus on planning semantic grasps

3 Semantic grasping

3.1 Semantic constraints and robotic grasping

Most robotic grasps are used for an intended manipulation task. To perform a specific manipulation task, some constraints are required to be satisfied by the robotic grasp. For a mug, *Pick-and-Place* and *Pour-Water* are two possible manipulation tasks. For *Pick-and-Place*, stability is one constraint. This constraint requires a grasp to be able to resist possible external disturbances during the manipulation process. For *Pour-Water*, stability is still necessary, but in addition to this, an extra constraint might require the robot hand to avoid blocking the opening area of the mug or to grasp the handle of the mug.

In order to plan appropriate grasps for different tasks, it is essential to satisfy semantic constraints. Figure 1 shows some examples of robotic grasps on a mug that are evaluated as stable grasps according to the epsilon quality metric in Ferrari and Canny (1992). All the grasps in Fig. 1 are stable in terms of force/form closure and they are all suitable for a *Pick-and-Place* task. However, if we consider using these grasps for a *Pour-Water* task, only grasps shown in Fig. 1c, d are suitable because in the first two grasps the palm blocks the opening area of the mug conflicting with the second semantic constraint required by a *Pour-Water* task. This example demonstrates that semantic constraints for grasps should be considered in grasp planning procedures.

Figure 2 shows a topological categorization of the domain of robotic grasps. Stable grasps only consider the stability of a grasp in resisting external disturbances. Task-specific grasps may not require the stability. In this work, we focus on the intersection of these two sub-domains of grasps, which are both stable and task-specific. We refer to this subset of grasps as *semantic grasps*.

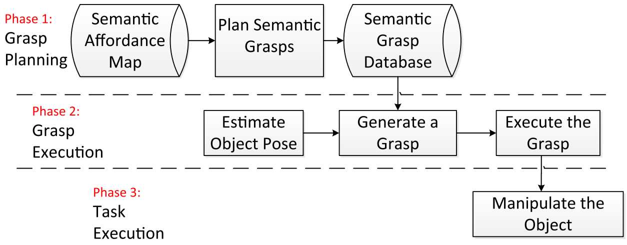


Fig. 3 A typical grasping pipeline considering semantic constraints. In the grasp planning phase, semantic grasps are planned and stored into a semantic grasp database. In the grasp execution phase, a planned semantic grasp is retrieved and executed based on requested manipulation tasks. Once a semantic grasp is executed, a manipulation chain as proposed in Dang and Allen (2010) could be executed to accomplish a manipulation task

In general, a grasping pipeline consists of three phases: grasp planning, grasp execution, and task execution. Figure 3 illustrates a typical grasping pipeline where our method fits in: compared to a traditional robotic grasping pipeline, our method considers semantic constraints in the grasp planning phase while keeping the rest of the pipeline intact. In the grasp planning phase, our grasp planning method is used to build a semantic grasp database. In the grasp execution phase, semantic grasps are retrieved and applied to the objects according to requested manipulation tasks. Once a semantic grasp is executed, a robot can proceed to the following manipulation task. We follow such a grasping pipeline where grasp planning and execution are separated and focus on developing a grasp planning method that considers semantic constraints.

3.2 Embedding semantic constraints

Semantic constraints are high-level concepts that are difficult to describe and difficult to generalize. Instead of representing semantic constraints explicitly, we attempt to specify semantic constraints using a predefined example grasp and use the example grasp to infer corresponding semantic constraints.

Many everyday objects are designed such that their geometries are appropriate for the corresponding manipulation tasks that they are associated with. For example, a mug has a handle and a body which are designed to be grasped. For a *Pour-Water* task, it is always a good strategy to grasp the body of the mug or to grasp the handle from the direction in which it stretches away from the body, because these two grasps satisfy the two semantic constraints a *Pour-Water* task requires: (1) grasp stability and (2) avoid blocking the opening area of the mug.

Semantic constraints imply requirements on the following aspects:

- Graspable part of the object to be approached
- Hand-object relative orientation
- Hand posture
- Contact locations on the hand

The graspable part of an object to be approached by a robot hand can be encoded using 3D depth images of the object from the approach direction of the hand. The depth images describe the partial geometry in view. It also indicates the orientation of this part of the object with respect to the hand. Hand posture can be derived directly from the joint values of the hand and contact information can be extracted from a set of tactile sensor arrays on the hand. Thus, we propose embedding semantic constraints into the related sensory data. Given an example grasp which has already satisfied specific semantic constraints, we can compare these quantities to those of grasp candidates on the same object or novel objects of the same class. If they are similar, we consider the corresponding semantic constraints as satisfied. Otherwise, we consider the semantic constraints as unsatisfied.

3.3 Semantic grasp

We define the *semantics* of a robotic grasp as the intended manipulation task whose semantic constraints are satisfied by the grasp. It is a symbolic label or a phrase that uniquely identifies a task of the object (e.g. "pour-water").

A *semantic grasp* is a robotic grasp that satisfies the semantic constraints imposed by a manipulation task. We write a semantic grasp formally as

$$\mathcal{SG} =< \mathcal{S}, \mathcal{K}, \mathcal{T} >, \ where$$

$\mathcal{S}$ is the semantic label of the grasp, e.g. "pour-water"; $\mathcal{K}$ is the hand kinematic data of the grasp, e.g. a set of joint angles and the orientation and location of the wrist with respect to the object; $\mathcal{T}$ is the tactile contacts of the grasp, e.g. arrays of tactile sensor readings.

3.4 Semantic affordance map

In Sect. 3.2, we discussed that the semantic constraints can be indirectly embedded in the object depth images, hand kinematic data, and tactile contacts. We now introduce a semantic affordance map $\mathcal{M}_\mathcal{C}$, which encapsulates all the related information and associates semantic grasps with an object class $\mathcal{C}$.

A *semantic affordance map* $\mathcal{M}_\mathcal{C}$ is a set of triples:

$$\mathcal{M}_\mathcal{C} = \{< \mathcal{P}, \mathcal{F}(\mathcal{D}), \mathcal{B} >\}, \ \ (\mathcal{B} = \{\mathcal{SG}\}), \ where$$

$\mathcal{P}$ is the hand approach direction to the object; $\mathcal{D}$ is a depth image of the object from $\mathcal{P}$; $\mathcal{F}(\cdot)$ is a function that extracts features from $\mathcal{D}$, which will be explained in Sect. 4.2; $\mathcal{B}$ is a set of semantic grasps associated with $\mathcal{P}$, which can be defined with different robot hands. Two relations are established in this map:

1. $\mathcal{F}(\mathcal{D}) \rightarrow \mathcal{P}$, given an image feature descriptor $\mathcal{F}(\mathcal{D})$ of a depth image of an object from a particular viewing angle (approach direction), this mapping tells us the current approach direction of the hand to the object. If the object is symmetric, this mapping can be one to many.
2. $\mathcal{F}(\mathcal{D}) \rightarrow \mathcal{B}$, given an image feature descriptor $\mathcal{F}(\mathcal{D})$ of a depth image of the object from a particular viewpoint, this mapping tells us the possible semantic grasps on the corresponding geometry.

A semantic affordance map is considered as a manual for semantic usage of an object. In a semantic affordance map, it is probable that many triples have an empty set of semantic grasps. This is because there are many approach directions that are not good for any manipulation tasks. So, only a few $\mathcal{B}$'s in a semantic affordance map contain semantic grasps. We will discuss a way to build a semantic affordance map in Sect. 4.

3.5 Planning method overview

Before we get into the details, we now provide an overview of our method. Our planning method is inspired by Castiello (2005) who showed that both cognitive cues and knowledge from previous experience play major roles in visually guided grasping. We use an example-based approach to mimic this experience-based procedure. By analogy, $\mathcal{M}_\mathcal{C}$ acts like an experience base of grasping objects of class $\mathcal{C}$. $\mathcal{B}$ records all the successful grasps that are experienced before. $\mathcal{P}$ and $\mathcal{D}$ are used to mimic human knowledge of object geometry.

To plan a grasp with semantics $\mathcal{S}$ on a target object of class $\mathcal{C}$, we assume a semantic affordance map on this object class, $\mathcal{M}_\mathcal{C}$, has been built using a representative object of this class. The representative object and the target object are similar in shape. With this semantic affordance map, a semantic grasp of semantics $\mathcal{S}$ is retrieved as a reference. Then, an initial approach direction to the target object is arbitrarily chosen and a depth image of the target object from this approach direction is taken. With the depth image, the current hand approach direction to the target object is estimated by looking up in the semantic affordance map. Utilizing this estimated approach direction, along with the tactile and kinematic information stored in the predefined semantic grasp, our method adjusts the hand to the ideal approach direction and searches along the ideal approach direction for stable grasps that have similar tactile contact formation and hand kinematics. With all these similarities being achieved, we consider the semantic constraints specified by the predefined

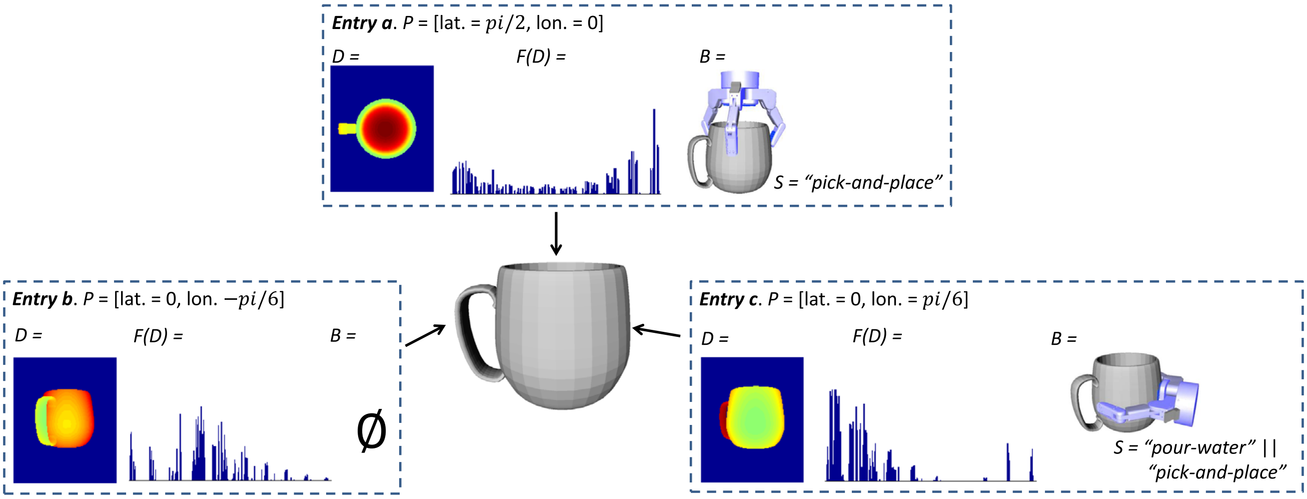

Entry a. P = [lat. = pi/2, lon. = 0]
D =
F(D) =
B =
S = "pick-and-place"
Entry b. P = [lat. = 0, lon. −pi/6]
D =
F(D) =
B =
∅
Entry c. P = [lat. = 0, lon. = pi/6]
D =
F(D) =
B =
S = "pour-water" || "pick-and-place"

Fig. 4 An example of semantic affordance map built on a mug model (shown in the *middle*). In this example, we show three entries of this semantic affordance map, labeled as **Entry a**, **Entry b**, and **Entry c** at the *upper left corner* of each *box*. Each entry is associated with an approach direction to the mug model indicated by an *arrow* next to the *box*. In each entry, we show (1) the approach direction $\mathcal{P}$ in the form of a latitude-longitude combination, (2) the depth image $\mathcal{D}$ of the object from the approach direction, (3) the shape context feature $\mathcal{F}(\mathcal{D})$based on the depth image, and (4) the bag of semantic grasps $\mathcal{B}$ along the approach direction. The *color* of a depth image goes from *light green* (close) to *red* (far). For **Entry a** and **Entry c**, only one semantic grasp is recorded in the bag of semantic grasps $\mathcal{B}$. For **Entry b**, there is no appropriate semantic grasp associated. The reason is that the robot hand is approaching the mug in an inappropriate direction towards the handle of the mug, which can be seen from the depth image in this entry. So the bag of semantic grasps of this entry is empty (Color figure online)

semantic grasp are also satisfied by the newly planned grasp on the target object.

In the following sections, we discuss in detail how to build a semantic affordance map for an object class and the steps our grasp planner takes to plan semantic grasps for a specific manipulation task.

4 Building a semantic affordance map

A semantic affordance map is dedicated to one object class. It is built in simulation and is designed to be an experience base that stores semantic grasps suitable for possible manipulation tasks an object of this class could be involved in. It also provides information to our grasp planner about how to satisfy these semantic constraints by relating semantic grasps with partial object geometry and approach direction. To build a semantic affordance map, we first choose a representative object of an object class and collect depth images from different approach directions to this object. This will provide all the $\mathcal{P}$'s and $\mathcal{D}$'s of a semantic affordance map. Then, these depth images are encoded and all the $\mathcal{F}(\mathcal{D})$'s are obtained. As the last step, exemplar semantic grasps are manually defined and stored in the $\mathcal{B}$'s where they are associated with the ideal approach directions. In the rest of the paper, we use the term *source object* to refer to the representative object of an object class in order to distinguish it from a target object.

Figure 4 illustrates a semantic affordance map of mugs using a representative mug model shown in the middle. We show detailed components of an entry as defined in Sect. 3.4. Although there are many entries in this semantic affordance map associated with different approach directions, we only show three of them here: **Entry a**, **Entry b**, and **Entry c**. **Entry a** is associated with an approach direction from the top of the mug. It stores the depth image of the mug from the top, the corresponding shape context feature, and a semantic grasp with a semantic label "pick-and-place". This grasp can be used for "pick-and-place" tasks. **Entry c** is another example storing related information for a semantic grasp labeled with semantics "pour-water". This grasp is suitable for *Pour-Water* tasks. **Entry b** is an entry with whose approach direction there is no valid semantic grasp associated since a robot hand will have to interact with the handle and the body of the mug from an inappropriate approach direction. In this specific semantic affordance map, we do not specify a semantic grasp on the handle of the mug. The reason is that the robot hand we used in our work is relatively too big to grasp the handle properly.

4.1 Sampling strategy

To sample around an object model in simulation and obtain the object's partial geometry information, we first create a unit sphere centered at the geometrical center of the object model. Along the latitude lines and the longitude lines, every

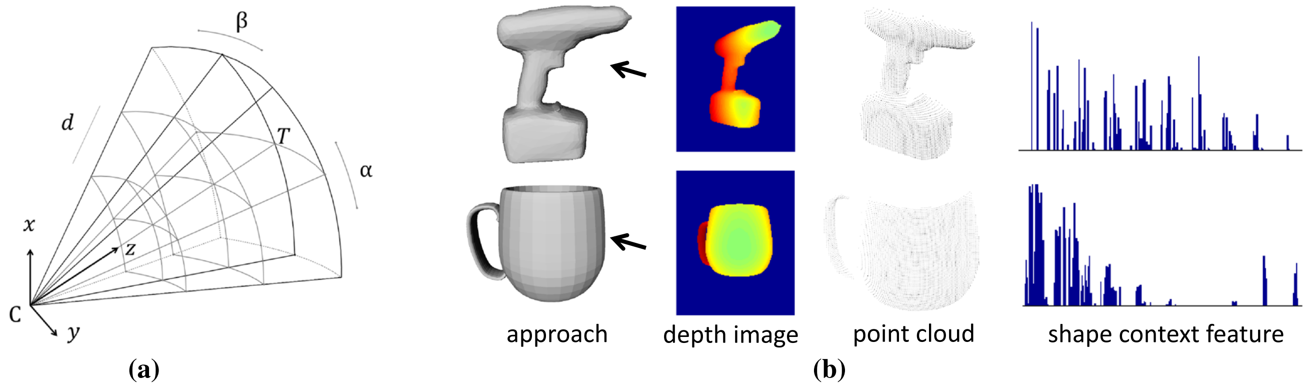


Fig. 5 Shape context computation and examples. **a** Illustrates the coordinate systems we used to compute shape context features. $C - xyz$ is the camera coordinate system with its origin at point C. In an eye-on-hand system, C is also the origin of the palm. The three dimensions of the spherical coordinate system for our shape context feature are as follows: d is the depth from the origin of the camera; α is the latitude in the camera coordinate system; β is the longitude in the camera coordinate system. **b** Shows two examples of our shape context features based on the depth images of the object obtained from the approach direction specified by the *arrow*. From *left to right* show the approach direction with a *black arrow*, the depth image from the approach direction, the point cloud reconstructed from the depth image, and our shape context feature of the depth image. **a** The spherical coordinate system for computing our shape context feature, **b** two examples of our shape context feature

2°, we collect a depth image of the object model using OpenGL. The virtual camera in OpenGL is placed at the crossing of the longitude and latitude with its horizontal axis aligned with the latitude line and its depth axis aligned with the radius of the unit sphere. We also move the virtual camera along the radius such that the bounding box of the object model is contained within the field of view. By doing this, we make sure that the entire object model is in the view.

4.2 Encoding a depth image

Given a set of sampled depth images using the strategy above, we encode them such that they can be used as keys to index all the samples effectively. Then, using the relation $\mathcal{F}(\mathcal{D}) \rightarrow \mathcal{P}$ as in Sect. 3.4, we can estimate the approach direction of a robot hand given the depth image from its angle. To encode a depth image, we use a similar idea from the shape context method by Belongie et al. (2002).

4.2.1 Shape context in 3D

Shape context is originally designed to measure the relationship of contour points on a shape with respect to a reference point on the contour. It captures the geometrical distribution of other points relative to the reference in a pre-designed histogram space, e.g. log-polar space in Belongie et al. (2002). So, it can be used as a global discriminative feature. In our problem, we want to capture the geometry feature of the partial object which is approached by the hand. We also want to distinguish the pose of the hand with respect to the partial object. To this end, we choose the origin of the palm, which is also where a virtual camera is placed to take a depth image of the object model, as the reference point to compute our shape context feature.

To compute our shape context feature from a depth image, we first transform each point of the object in the reconstructed point cloud from the Cartesian space into a 3-dimensional spherical coordinate space. One dimension is the radial distance (d) from the camera origin to a point on the object. This data is stored directly in the depth image. The other two dimensions are the latitude (α) and the longitude (β) of each point in the camera coordinate system. In Fig. 5a, we illustrate these dimensions graphically. The camera is placed at point C looking towards the object's center of gravity T with its x, y, z axes oriented as drawn in the figure. Then, for a given point P sensed from a depth image, its latitude α is calculated as the angle between the z axis and ray CP's projection on plane xCz. Its longitude β is measured as the angle between the z axis and ray CP's projection on plane yCz.

4.2.2 Normalization

In our sampling strategy, the distance at which a depth image is taken is determined by the actual size of the object in the field of view. So, for objects of different sizes, the distances at which the camera is placed would be different. We do not want this difference to influence our encoding. So we normalize the depth of each point such that their values are rescaled to between zero and one. For the other two dimensions: the longitude and latitude, we also normalize them so that their values are rescaled to between zero and one.

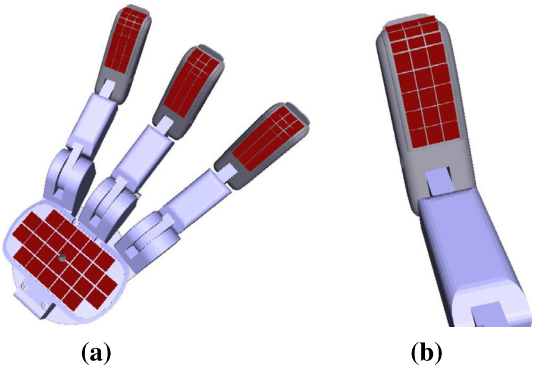
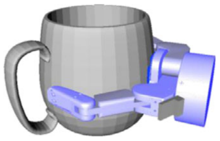


Fig. 6 Tactile sensor configuration of a Barrett hand. 96 tactile sensors, illustrated in *red*, are distributed on the surface of the palm and three fingertips. **a** Palm, **b** finger

After we have normalized the spherical coordinates for each point, all the values in each dimension range from zero to one. Then, we divide the normalized 3-dimensional spherical space into different sub-regions similar to Fig. 5a. We calculate the percentages of points in each sub-region and build a histogram out of it. Figure 5b shows two examples of our shape context feature. Using this feature, we can estimate approach direction based on a depth image. We will explain our method to estimate approach direction in Sect. 5.2.

4.3 Recording semantic grasps

From the previous sampling step, we have already computed all the $\mathcal{P}$'s and $\mathcal{F}(\mathcal{D})$'s of a semantic affordance map. To complete a semantic affordance map, we need to input sets of semantic grasps and associate them with ideal approach directions. To do this, we manually select a good approach direction for each possible manipulation task. Along this approach direction, a semantic grasp is specified manually. Then, the following three components are recorded and associated with this semantic grasp. The first component is the semantic label of the grasp. This can be a short phrase such as "pick-and-place" or "to-drill". The second component is the hand kinematic data, which indicate the shape of the hand when the grasp is applied. To store the hand kinematics, we store the angle value for each joint. The third component is the tactile contacts. To record them, we store the location and the orientation of the active contacts when a grasp is being applied on an object. As an example, Fig. 6 illustrates the distribution of tactile sensors on a Barrett hand. Figure 7 shows one grasp and the three components that are stored with the grasp in the semantic affordance map. Throughout this process, we have defined a semantic grasp and associated it with one approach direction to the object. If there is more than one approach direction for a specific manipulation task, we apply the same method to define additional semantic grasps along other approach directions.



$\mathcal{SG} = <\ \mathcal{S},\mathcal{K},\mathcal{T}\ >$
$\mathcal{S}$: *pour-water*
$\mathcal{K}$: *[0 0.75125 0.250417 0.7125 0.28375 0.79375 0.264583]*
$\mathcal{T}$: *[0 0 0 0 0 0 0 9.02527 0 0 0 0 0 0 0.100722 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0.00238977 0 0 7.36216 0.808482 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3.73941 0 0 4.13163
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1.36381 3.30897 0
1.61785 1.73428 0 0 0 0 0 0]*

Fig. 7 A semantic grasp example and the related grasp information stored. In this example, the grasp semantic label ($\mathcal{S}$) is a phrase "pour-water" which specifies the robotic task this grasp is suitable for. The hand kinematic data ($\mathcal{K}$) is represented by the array of seven numbers which correspond to the seven joint angles of the robot hand. The tactile sensor readings ($\mathcal{T}$) record the simulated force values sensed on each of the sensors which are distributed on the fingertips and the palm. Based on $\mathcal{K}$ and $\mathcal{T}$, the location and orientation of each tactile contact can be calculated

5 Planning semantic grasps

Given a manipulation task and a semantic affordance map, planning a semantic grasp on a target object can be thought of as a search for stable grasps that satisfy the semantic constraints indicated by an exemplar semantic grasp predefined on a source object. Figure 8 illustrates the three steps of our planning procedure. In the following, we will explain each of the three steps in detail.

5.1 Step 1: retrieve a semantic grasp

The first step is to retrieve a predefined semantic grasp from the semantic affordance map. This is done by searching within the semantic affordance map and looking for semantic grasps with an appropriate semantic label according to the requested manipulation task. In the following steps of the pipeline, we will use this semantic grasp as a reference for planning. In Fig. 8, we are trying to plan a semantic grasp on a drill (shown as the target object on the top of Fig. 8). The intended task is "to-drill". Thus, the semantic grasp which has a semantic label "to-drill" is retrieved from the semantic affordance map (shown in "Step 1" in Fig. 8).

It is worth noting that it is possible that there are more than is semantic grasp associated with a required task. In this situation, more information should be given and considered in the decision making process, such as kinematic feasibility of the robot arm.

5.2 Step 2: achieve the ideal approach direction

Once the exemplar semantic grasp is retrieved, the first semantic constraint we need to satisfy on the target object is the geometric constraint which requires a specific part of the target object to be grasped. This constraint can be inferred by the approach direction of the hand. So, in order to get to the most appropriate approach direction required by the semantic constraints, we first estimate the current approach

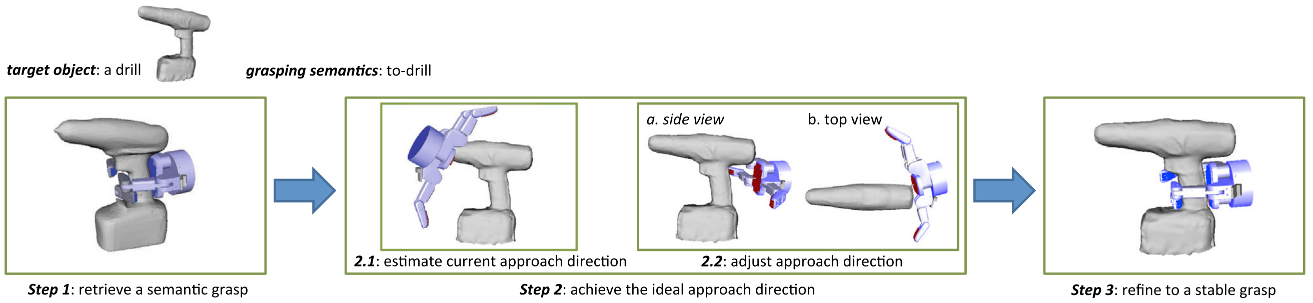

target object: a drill
grasping semantics: to-drill
a. side view
b. top view
2.1: estimate current approach direction
2.2: adjust approach direction
Step 1: retrieve a semantic grasp
Step 2: achieve the ideal approach direction
Step 3: refine to a stable grasp

Fig. 8 Process of planning a semantic grasp on a target object (i.e., a drill) with a given grasping semantics "to-drill" and a semantic affordance map built on a source object (i.e., another drill shown in *Step 1*, which is similar to the target drill). *Step 1* is to retrieve a semantic grasp that is stored in the semantic affordance map. This semantic grasp is used as a reference in the next two steps. *Step 2* is to achieve the ideal approach direction on the target object according to the exemplar semantic grasp. Once the ideal approach direction is achieved, a local grasp planning process starts in *Step 3* to obtain stable grasps on the target object which share similar tactile feedback and hand posture as that of the exemplar semantic grasp

direction of the hand with respect to the object, which is done by using the relation $\mathcal{F}(\mathcal{D}) \rightarrow \mathcal{P}$ as we discussed in Sect. 3.4. To do this, a depth image of the target object is taken from the hand's current approach direction. We encode the depth image as in Sect. 4.2 to generate our shape context feature. Then, we look up in the semantic affordance map and search for k nearest neighbors based on this feature vector.

To match against the entries in the semantic affordance map, we used χ^2 distance to calculate the difference between two feature vectors. Since k could be larger than one, we need to use some rules to decide a neighbor that is most widely agreed among these k nearest neighbors. To do this, we calculate a cumulative distance for each neighbor from the remaining neighbors. This cumulative distance indicates the extent to which other neighbors agree with it. If the cumulative distance is very small, it means this neighbor is very close to all the other neighbors, thus it is agreed by all the other neighbors. If the cumulative distance is large, it indicates that there are some neighbors far away from this one and they do not agree with this neighbor. We illustrate this scheme in Algorithm 1. In this algorithm, $D(\cdot)$ denotes a distance function that calculates the actual angle between the approach directions represented by the two neighbors. The most agreed neighbor is used as an indication to the current approach direction of the robot hand to the object.

Once the current approach direction is estimated, adjustment can be done by calculating the transform between the current approach direction and the ideal approach direction that satisfies the semantic constraints.

Algorithm 1: Find the most agreed neighbor

```
Input: $k$ nearest neighbors $N = \{n_1, \ldots, n_k\}$
Output: the most agreed neighbor $n_m$
Initialize array $v$ with $k$ entries for cumulative distances
foreach $n_i \in N$ do
    foreach $n_j \in N - \{n_j\}$ do
        //$D(n_x, n_y)$ measures the angle between the two approach
        directions associated with $n_x, n_y$
        $v[i] += D(n_i, n_j)$;
        $v[j] += D(n_j, n_i)$;
    end
end
$n_m = \arg\min_i(v[i])$
return $n_m$
```

5.3 Step 3: refine to a stable grasp

Based on the previous two steps, a promising hand approach direction has been achieved for the specific manipulation task. This is only a good start to satisfy all the semantic constraints embedded in the predefined semantic grasp, because solely relying on the approach direction is not sufficient. For example, the stability of the grasp, similar kinematics, and tactile contacts of the hand cannot be guaranteed simply by approaching the object from an ideal direction. We consider them in a grasp refinement step. In this step, we use the Eigengrasp planner by Ciocarlie and Allen (2009) to generate a sequence of stable grasps along the approach direction. We then sort them according to their similarities with the predefined semantic grasp. We will explain these methods below.

5.3.1 The eigengrasp planner

The Eigengrasp planner is a stochastic grasp planner that searches for stable grasps in a low-dimensional hand posture space spanned by eigenvectors called Eigengrasps. As an example, for a Barrett hand which was used in our experiments, it has seven joints and 4° of freedom. Two Eigengrasps $E =< e_1, e_2 >$ were used to describe the hand posture. One controls the spread angle and the other controls the finger flexion as illustrated in Fig. 9. The wrist pose is sampled locally around the initial grasping pose using a complete set of six parameters: $P \ =< roll, pitch, yaw, x, y, z\ >$. These six parameters generates a hand offset transformation from the initial hand pose. Thus, the search space for the

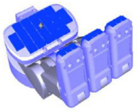
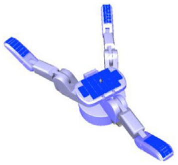
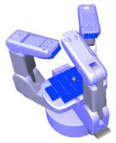
e1			e2		
Description	min	max	Description	min	max
Spread			Finger flexion		

Fig. 9 Two Eigengrasps used to specify the hand posture of a Barrett hand, e_1 and e_2. e_1 controls the spread angle between two fingers. e_2 controls the finger flexion of all the three fingers. Interested readers could refer to Ciocarlie and Allen (2009) for more details about Eigengrasps

Eigengrasp planner in our case is an eight-dimensional space $S = \{E, P\}$. The stability of a grasp is evaluated using the epsilon quality, ϵ, which measures the minimum magnitude of external disturbance to break the grasp Ferrari and Canny (1992).

In order to generate grasps with similar hand kinematics as the exemplar semantic grasp, during this refinement step, we place a constraint on the hand kinematics, i.e. the DOF's of the hand. We use the corresponding Eigengrasps of the DOF values recorded in the semantic grasp as a kinematic reference and probabilistically limit the search region to be around this kinematic reference. More detailed explanation about grasp neighbor generation and the stochastic grasp planner can be found in Ciocarlie and Allen (2009). Based on this setup, the output grasps of the planner should share similar kinematics of the exemplar semantic grasp, maximumly preserving a similar hand posture during grasping.

5.3.2 Grasp ranking

The Eigengrasp planner generates a list of stable grasps which share similar hand kinematics with the exemplar semantic grasp. In this step, we would also like to have a more detailed evaluation of their similarity to the exemplar semantic grasp. Our similarity measurement considers both the tactile contacts and the hand kinematics. With similar tactile contacts and hand kinematics, we expect the hand to touch similar parts of the object which improves the possibility that the planned grasp holds the object in the way which is defined in the exemplar semantic grasp. To measure the similarity between a newly found grasp candidate and the exemplar semantic grasp, we designed a distance function as follows.

$$dist(\mathcal{G}_1, \mathcal{G}_2) = \frac{1}{2} \cdot \sum_{m=1}^{N_1} \min_n \left(||c_m^1 - c_n^2||\right) + \frac{1}{2} \cdot \sum_{m=1}^{N_2} \min_n \left(||c_m^2 - c_n^1||\right) + \alpha ||js_1 - js_2|| \qquad (1)$$

where c_m^i is the m^{th} contact of the grasp i, N_i is the number of contacts of grasp i, and js_i is the joint values for the selected DOFs of the grasp i. α is a scaling factor for the euclidean distance between the joint angles of the selected DOFs. The first two parts of the right side of the equation measure the euclidean distance between the two sets of contacts in terms of their Cartesian positions. The third part measures the difference between the joint angles for the selected DOFs. The smaller the distance is, the more similar the two grasps are. More detailed explanation about the similarity evaluation can be found in our work in Dang and Allen (2014).

6 Experiments

In this section, we show the experiments we have conducted. We first explain how we built our semantic affordance maps on several everyday objects. We present evaluation results of the performance of our method to estimate the approach direction based on the shape context feature described in Sect. 4.2. Then, we show our experiments on planning semantic grasps inside the *GraspIt!* simulator by Miller and Allen (2004) and grasping objects with the planned semantic grasps using a physical robot. The last two sets of experiments correspond to the first two levels of the grasping pipeline shown in Fig. 3. The object classes are manually determined in our current experimental setup. In the future, object classification algorithms can be exploited to automate this process.

6.1 Semantic affordance maps

The first step of our experiments was to build semantic affordance maps for different object classes. We chose four source objects (shown in Fig. 10) from four object classes which we frequently encounter in our everyday life: mugs, phones, handles, and drills. In table-top object manipulation tasks, objects are usually placed in their canonical upright orientations. So, we assumed that all the object models are defined in their canonical upright orientations with respect to a common plane. In a physical experiment, this common plane is

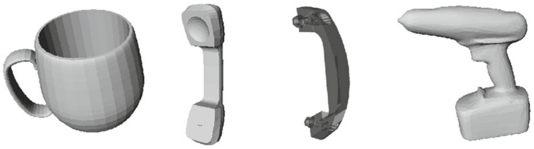


Fig. 10 Source objects selected as representatives of four object classes: mugs, phones, handles, and drills. A semantic affordance map is built based on each of these objects

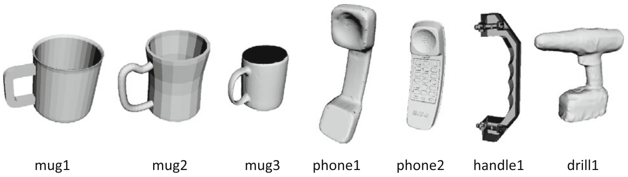


Fig. 11 Target objects selected in our experiments. Each of these target objects belongs to one of the object classes represented by the source objects shown in Fig. 10. These target objects are different from the source objects in shape and size

the surface of a table. For each of these source objects, we built a semantic affordance map using the method discussed in Sect. 4. In the sampling step, we collected depth images of each object model from 8100 different approach directions. To encode a depth image, we empirically discretized the spherical coordinate space by 10 steps along dimension d, 6 steps along dimension α, and 6 steps along dimension β (see Fig. 5a). Thus, the length of our shape context feature vector is $10 \times 6 \times 6 = 360$. To define exemplar semantic grasps, the *Graspit!* simulator was used to record stable grasps suitable for related tasks, which were specified and determined by a human user. Tactile sensors were simulated based on the soft finger model proposed by Ciocarlie et al. (2007). Detailed explanation about our tactile simulation process can be found in our previous work Dang et al. (2011). As an example, Fig. 4 shows three entries of a semantic affordance map on mugs.

6.2 Time for building a semantic affordance map

Building a semantic affordance map involves two steps. The first step is to generate partial object geometries. The second step is to associate semantic grasps with ideal approach directions. In our work, the machine we used to compute depth images was equipped with an Intel Core 2 Quad Q8300 processor, an Nvidia Quadro FX1500 GPU, and 4 GB RAM. To generate 8100 depth images for an object model, it took about 45 min without any code optimization. It is worth noting that this step takes place offline and the generated depth images can be stored and reused. To define a semantic grasp, it took <3 min for a user to specify hand kinematics, test grasp stability, and associate the grasp with an approach direction.

6.3 Estimating an approach direction

In our method, the "Step 2" as described in Sect. 5 and Fig. 8 is to estimate the current approach direction and use the estimated pose information to adjust the hand to achieve the ideal approach direction from which a grasp refinement procedure takes place in "Step 3" (see Sect. 5; Fig. 8). In "Step 3", the search for stable grasps is probabilistically constrained within the neighborhood of the adjusted approach direction achieved in "Step 2", so it is beneficial to see how accurate the approach direction estimation is. This could help us decide the size of the search space around the adjusted approach direction. To evaluate the performance of our approach direction estimation method, we used the semantic affordance maps built in Sect. 6.1. We chose $k = 5$ in "Step 2" as described in Sect. 5. We then selected seven target objects as test objects (shown in Fig. 11). Each of these target objects belongs to one object class represented by one source object shown in Fig. 10. Compared to the source objects which were used to build the semantic affordance maps of each object class, the target objects have different sizes and shapes. The objects *mug3*, *phone2*, and *drill1* are modeled using a NextEngine 3D scanner. In the experiment, we first took a depth image at each of the sampling approach directions as we did on the source object. We encoded each depth image using the method we discussed in Sect. 4.2. Then, we applied our method in Sect. 5.2 to estimate the approach direction. The error was measured by the angle between the estimated approach direction and the true approach direction.

Figure 12 shows the experimental results on the accuracy of the estimation of the approach direction on the test objects: *mug1, mug3, phone1, handle1*, and *drill1*. Out of these five objects, more than 90 % of the tests have errors <20°. Four of these five objects have more than 76 % of the tests with errors less than 10°. For target objects which have larger variance in shape and size from the source objects, we observed that our method has difficulties predicting the accurate approach direction. Figure 13 shows the performance of our experiments on two target objects: *mug2* and *phone2*. Compared to the source object *mug*, object model *mug2* is concave and is much thinner. Compared to the source object *phone*, object model *phone2* is shorter, fatter, and does not have a thin handle between the speaker and the microphone. These factors make it difficult to have an accurate prediction based on the statistics of the point distributions. We think that considering surface normal at each sampling point would help alleviate this issue.

6.4 Planning semantic grasps

Our grasp planning experiments are summarized in Table 1. It presents the object classes we used, the intended task a planned grasp is used for, and the corresponding

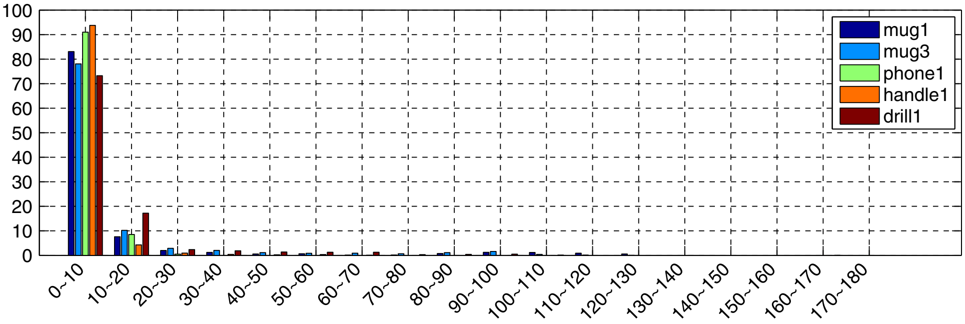


Fig. 12 Accuracy of approach direction estimation using our shape context feature on several everyday objects. *Horizontal axis* shows the brackets of error ranges (in degree). *Vertical axis* shows the percentage (%) of test trials whose errors are within each error range indicated on the *horizontal axis*

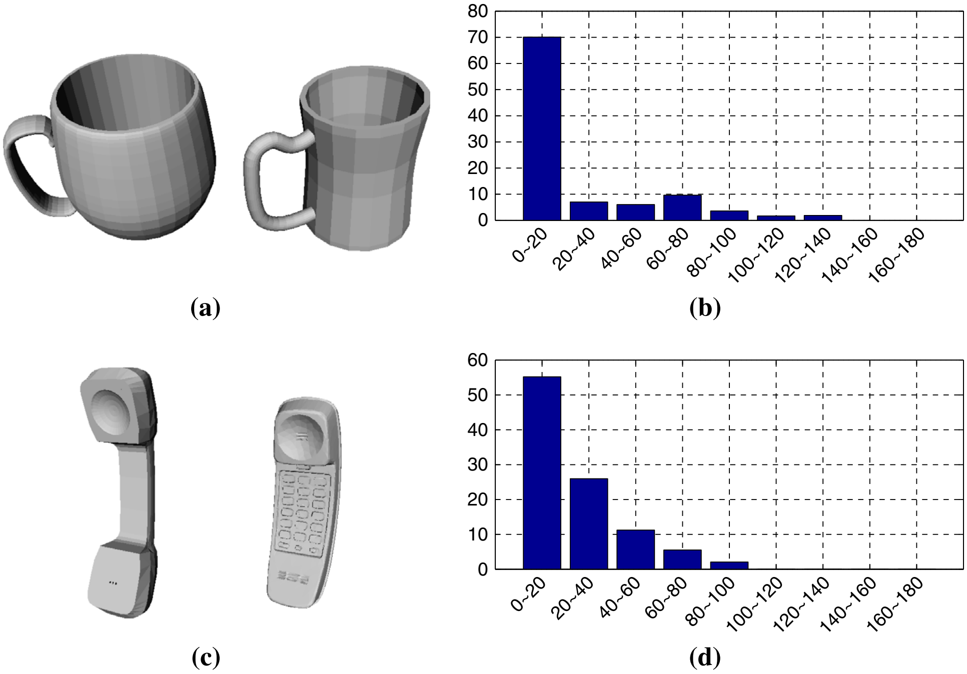


Fig. 13 Performance of estimating approach directions on objects with larger variance in shape and size. **a**, **c** Show the source models (*left*) and the target models (*right*). **b**, **d** Show the performance of estimating approach directions to the target models using the semantic affordance maps built on the source models. *Horizontal axis* shows the brackets of error ranges (in degree). *Vertical axis* shows the percentage (%) of tests whose errors are within each error range indicated on the *horizontal axis*

Table 1 Objects, manipulation tasks, and semantic constraints

Object class	Manipulation task	Semantic constraints
Mug	Pour water	Not blocking the open area
Phone	Answer a call	Grasping the handle
Handle	Pull/push the handle backward/forward	Power grasping the mid-point
Handle	Push the handle sideway	Hook grasping the mid-point
Drill	Hold and drill	Grasping the handle

semantic constraints of each task the grasp should satisfy. We used the seven objects shown in Fig. 11 as the target objects. We applied our planning algorithm to plan semantic grasps for each target object. For object class *mug*, we tested our planning method on three different target mug models. For object class *handle*, we tested our planning method for two different manipulation tasks.

In Fig. 14, we show experimental results of planning semantic grasps on these target objects. In each row, the second column shows a predefined semantic grasp that was stored in the semantic affordance maps. The third column shows the comparison of the geometry between the source(left) and the target(right) objects. The source objects are those ones that were used for building semantic affordance maps. They are different from the target objects, but similar in shape. The last two columns show the top two ranked grasps generated by our planning method according to their tactile and hand posture similarities. The experimental results indicate that, by using our planning algorithm, semantic grasps can be synthesized from similar objects with predefined exemplar semantic grasps.

6.5 Semantic grasping with a physical robot

Following the planning experiments, we connected our planning method to a grasp execution system and tested our entire grasping pipeline from modeling a physical object using an off-the-shelf 3D scanner to planning a semantic grasp on

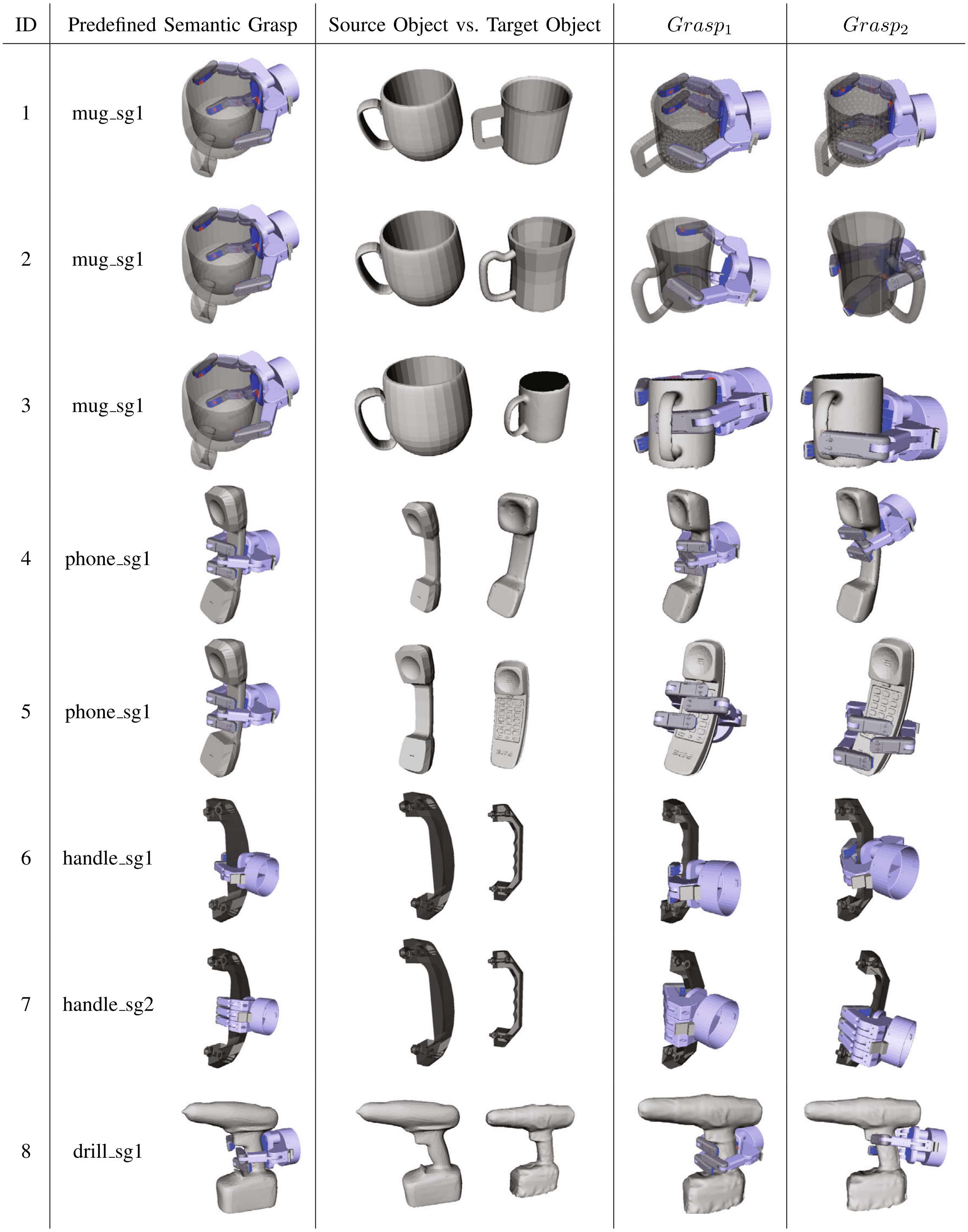


Fig. 14 Semantic grasps planned on typical everyday objects. From *left to right*: experiment ID, the predefined semantic grasps stored in the semantic affordance map, a pair of source object and target object for each experiment, and top two grasps generated. Last *two columns* for the top two grasps were obtained within 180 s and are both stable in terms of their epsilon quality. Some objects are displayed with transparency to show the grasp

the model and to executing a semantic grasp for a requested manipulation task on a physical object.

We chose a mug, a phone, and a drill as target objects, shown in experiments 3, 5, and 8 in Fig. 14 respectively. The geometrical models of the physical objects obtained using a NextEngine 3D scanner are shown in the right part of the third column of each experiment. Using our proposed method, we planned semantic grasps and stored them in a semantic grasp

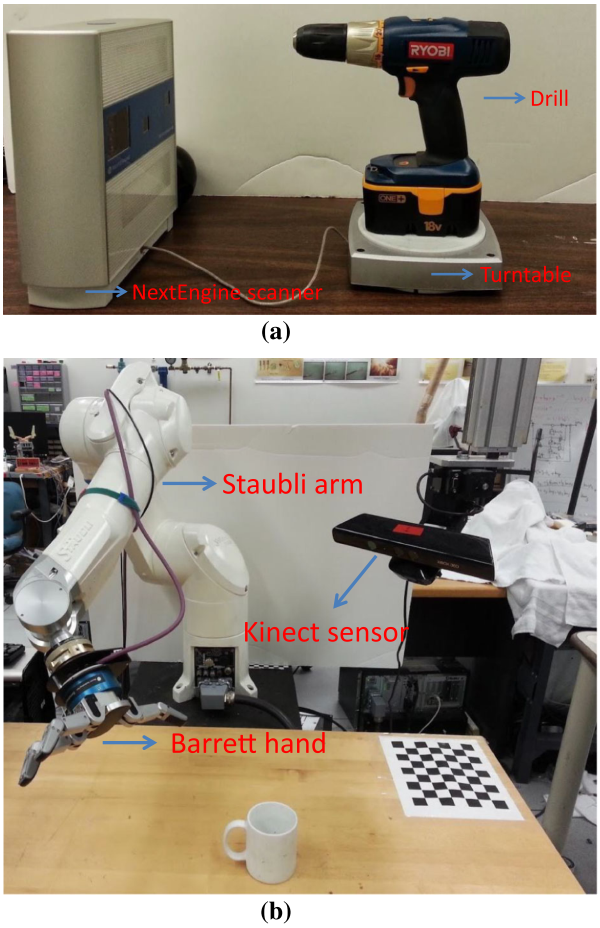


Fig. 15 Experimental setup. **a** A NextEngine 3D scanner scanning a drill, **b** the robot platform used in our experiments. A Barrett hand is attached to the end of a Staubli arm. A Kinect sensor is used to obtain point clouds of the work space

database. In the grasp execution stage, a target object was placed in front of the robot. A Kinect sensor acquired a 3D point cloud of the scene. The recognition method proposed by Papazov and Burschka (2010) was used in our perception system, which uses partial geometry of an object to recover its full 6D pose. Once the pose of an object was recovered, a planned semantic grasp was retrieved from the semantic grasp database according to the object name and the semantic label. Finally, the OpenRave planner by Diankov and Kuffner (2008) generated a collision-free trajectory to the final grasping pose and the hand moved to the target pose and executed the grasp.

Figure 15a shows a snapshot of a NextEngine 3D scanner scanning a drill. Figure 15b shows the setup of our robot platform. The robot arm we used was a Staubli arm, which is a six-DOF robot arm. A Barret hand is attached to the end of the Staubli arm.

Figure 16 shows snapshots of the process of the grasping pipeline using semantic grasps in the experiments. The first two columns show the predefined semantic grasps on the source objects and the generated semantic grasps on the target objects. The third column shows the physical objects placed in the robot workspace. The fourth column shows the point clouds of the workspace reconstructed from a Kinect sensor. The fifth column shows the final grasp of a Barrett hand. To examine pose estimation, we placed the full 3D models of the physical objects (shown as pink points) at the estimated pose from the perception algorithm and overlaid them on the point clouds of the workspace.

7 Discussion

As our first step towards planning task-specific grasps, our work focuses on using the hand kinematics, tactile feedback, and object geometry to encode semantic constraints, there are also other attributes we would like to consider in our future work, such as the potential force and torque space. We believe this can be integrated into the refinement step of our grasp planning algorithm.

The determination of representative object models and object classes is done by human beings. Thus, human intelligence is involved in this process. There has been research work focusing on shape classification, e.g. Ling and Jacobs (2007). By integrating with these algorithms, it is possible to automate the process of object selection and classification and finally minimize the dependence on human intelligence.

Although we used a Barrett hand in our work, the presented framework can be applied to other robot hands as well. Semantic grasps in $\mathcal{B}$ can be defined with different robot hands. To add semantic grasps defined with a new robot hand into an existing semantic affordance map, we need to first specify semantic grasps with the new robot hand model and then store the newly generated semantic grasps into the semantic affordance map. In our current work, we manually define semantic grasps inside our *GraspIt!* simulator. As our future work, we will explore other approaches to automate the grasp recording process, making it less dependent on human users.

A model of the object may not be necessary. For a system where grasp planning and executing are separated, it is ideal to get the model of the object beforehand. If the object model is not obtained in advance, some existent modeling algorithms can be used to reconstruct the object from the real scene using depth sensors, such as a Kinect sensor. Another approach can be to connect our algorithm with a physical robot system, obtaining depth images directly from physical depth sensors and making hand movements with the robot. In this case, Step 2 of Fig. 8 takes place in the physical world instead of the virtual simulation environment. Then, we are merging the virtual world and the physical world. The planning process which used to be in a virtual world

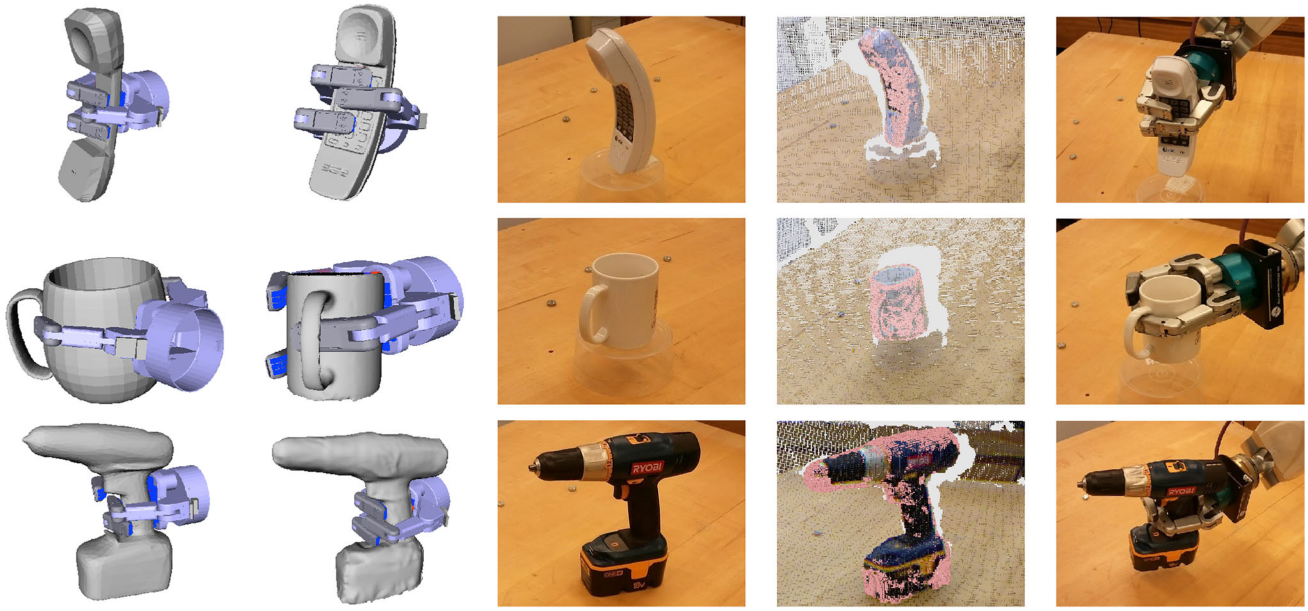


Fig. 16 Grasping physical objects with semantic grasps. From *left to right*: predefined semantic grasps on source objects, semantic grasps on target objects generated using the proposed method, a snapshot of the physical object in the workspace, reconstructed point cloud of the object in the workspace, and the final grasp of a Barrett hand attached to a Staubli arm. *Pink points* in the point clouds are object models placed with the estimated pose (Color figure online)

now becomes an exploration process in the physical world, defining a control process to achieve a semantic grasp specified in a predefined exemplar grasp, making this planning algorithm more like a control algorithm.

Currently, the refinement step in our method consists of (1) planning a sequence of stable grasps using a stochastic grasp planner and (2) sorting planned grasps based on their similarities to predefined semantic grasps. However, stochastic planning may not be the ideal solution. It can take more time than necessary to find an appropriate grasp. One potential alternative approach is to integrate task-related information with local geometry information and make hand adjustments as developed by Hsiao et al. (2011).

Symmetric objects may raise challenges for our matching method which estimates the approach direction of the hand: multiple views of a symmetric object could produce the same depth image, thus the approach direction estimation method will face difficulty in deciding an accurate approach direction from a depth image. In these situations, our method may have difficulties in “Step 2” to generate the ideal hand adjustment and adjust the hand to the ideal approach direction. We believe that by utilizing more sophisticated localization methods these problems could be alleviated.

Although we assumed that objects have canonical upright orientations, our work can be extended to deal with objects without canonical upright orientations. One approach is to sample the object in a more comprehensive way such that we can cover many potential orientations. Once these samples are generated, we can store them into the semantic affordance map and use them for approach direction estimation.

The robustness of our method is closely related to the performance of the feature which we used to encode a depth image. This controls the requirement of similarity between objects so that this method can succeed in transferring semantic grasps between them. We have shown in our experiments that our method can deal with objects with similar shapes. With large shape variance, our method may have difficulties in providing accurate predictions of approach directions. However, this issue can be alleviated by using a larger search space in the grasp refinement step. Thus, our method remains relatively robust overall. As our next step, to further improve the robustness and capability of our method to deal with objects with large shape variance, we will be exploring more advanced features to estimate approach directions towards an object.

Our current shape context feature focuses on global object shapes. This to some extent limits the capability of our method to deal with objects with large shape variance. However, with careful modification, we can improve the generality of our method. For objects within the same class, they may have large overall shape variance, but they usually share similar functional parts. These functional parts should be grasped in similar ways. For example, two mugs may have dramatically different global shapes, but their handles can be very similar and we can grasp the handles in a similar way. Thus, if we define semantic grasps on these similar functional parts and build semantic affordance maps based on these functional parts instead of the complete object models, our method will be able to transfer semantic grasps across functional parts. In this sense, we will be able to improve our method to deal with objects with large overall shape variance. To realize this approach, a segmentation method is necessary to recognize and separate functional parts in the range scan of the object, e.g. Varadarajan and Vincze (2011b).

In our work, we do not consider the kinematic feasibility of a grasp. For example, a planned semantic grasp may not be kinematically feasible in a real environment due to collisions or other workspace constraints. This could be solved by using collision checking to filter out infeasible grasps after a number of good semantic grasps are produced or utilizing algorithms such as in Chang et al. (2008) to achieve the required pre-grasp pose of the object.

8 Conclusion

In this paper, we develop an example-based grasp planning method to plan stable robotic grasps which satisfy semantic constraints required by a specific manipulation task. We propose using partial object geometry, hand kinematic data, and tactile contacts to embed semantic constraints. To this end, we introduce a semantic affordance map which relates partial geometry features to semantic grasps and records grasp related information. Using this map, our method considers the semantic constraints imposed by a specific task and plans semantic grasps accordingly. We show experiments of planning and executing semantic grasps on everyday objects.

For the next step, we will be considering possible ways to generalize the semantic affordance map so that it would be easier to transfer grasping knowledge between objects and tasks while preserving their semantic affordance. Although we have shown our approach with a Barrett hand, our method can be applied to different robot hands. The overall framework and workflow will remain the same as described in this paper. We will also expand the classes of objects and shape features to fit into our framework.

Acknowledgments This work is funded by NSF Grant IIS-0904514.

References

Aleotti, J., & Caselli, S. (2011). Part-based robot grasp planning from human demonstration. In *ICRA* (pp. 4554–4560).

Aleotti, J., & Caselli, S. (2012). A 3d shape segmentation approach for robot grasping by parts. *Robotics and Autonomous Systems*, *60*(3), 358–366. doi:10.1016/j.robot.2011.07.022.

Belongie, S., Malik, J., & Puzicha, J. (2002). Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*, 509–522.

Ben Amor, H., Heumer, G., Jung, B., & Vitzthum, A. (2008). Grasp synthesis from low-dimensional probabilistic grasp models. *Computer Animation and Virtual Worlds*, *19*(3–4), 445–454. doi:10.1002/cav.252.

Ben Amor, H., Kroemer, O., Hillenbrand, U., Neumann, G., & Peters, J. (2012). Generalization of human grasping for multi-fingered robot hands. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 2043–2050).

Berenson, D., & Srinivasa, S. (2008). Grasp synthesis in cluttered environments for dexterous hands. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids08)*.

Berenson, D., Diankov, R., Nishiwaki, K., Kagami, S., & Kuffner, J. (2007). Grasp planning in complex scenes. In *7th IEEE-RAS International Conference on Humanoid Robots* (pp. 42–48). doi:10.1109/ICHR.2007.4813847.

Boularias, A., Kroemer, O., & Peters, J. (2011). Learning robot grasping from 3-d images with markov random fields. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1548–1553). doi:10.1109/IROS.2011.6094888.

Castiello, U. (2005). The neuroscience of grasping. *Nature Reviews Neuroscience*, *6*(9), 726–736.

Chang, L. Y., Zeglin, G., & Pollard, N. (2008). Preparatory object rotation as a human-inspired grasping strategy. In *IEEE-RAS International Conference on Humanoid Robots* (pp. 527–534).

Ciocarlie, M., Lackner, C., & Allen, P. (2007). Soft finger model with adaptive contact geometry for grasping and manipulation tasks. In *World Haptics Conference* (pp. 219–224).

Ciocarlie, M. T., & Allen, P. K. (2009). Hand Posture Subspaces for Dexterous Robotic Grasping. *The International Journal of Robotics Research*, *28*(7), 851–867.

Dang, H., & Allen, P. (2010). Robot learning of everyday object manipulations via human demonstration. In *IROS* (pp. 1284–1289). doi:10.1109/IROS.2010.5651244.

Dang, H., & Allen, P. (2012). Semantic grasping: Planning robotic grasps functionally suitable for an object manipulation task. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 1311–1317).

Dang, H., & Allen, P. (2014). Stable grasping under pose uncertainty using tactile feedback. *Autonomous Robots*, *36*(4), 309–330.

Dang, H., Weisz, J., & Allen, P. (2011). Blind grasping: Stable robotic grasping using tactile feedback and hand kinematics. In *IEEE International Conference on Robotics and Automation (ICRA)* (pp. 5917–5922). doi:10.1109/ICRA.2011.5979679.

Detry, R., Kraft, D., Buch, A., Kruger, N., & Piater, J. (2010). Refining grasp affordance models by experience. In *IEEE International Conference on Robotics and Automation* (pp. 2287–2293). doi:10.1109/ROBOT.2010.5509126.

Diankov, R., & Kuffner, J. (2008). Openrave: A planning architecture for autonomous robotics. Tech. Rep. CMU-RI-TR-08-34, Robotics Institute, Pittsburgh, PA.

Dogar, M., & Srinivasa, S. (2011). A framework for push-grasping in clutter. In P. Abbeel (Ed.), *Hugh Durrant-Whyte NR. Science and Systems VII*. Cambridge: MIT Press.

Ferrari, C., & Canny, J. (1992). Planning optimal grasps. In *ICRA* (vol. 3, pp. 2290–2295).

Geidenstam, S., Huebner, K., Banksell, D., & Kragic, D. (2009). Learning of 2D grasping strategies from box-based 3D object approximations. In *Proceedings of Robotics: Science and Systems*.

Gioioso, G., Salvietti, G., Malvezzi, M., & Prattichizzo, D. (2012). An object-based approach to map human hand synergies onto robotic hands with dissimilar kinematics. In *Proceedings of Robotics: Science and Systems, Sydney, Australia*.

Goldfeder, C., & Allen, P. K. (2011). Data-driven grasping. *Auton Robots*, *31*, 1–20. doi:10.1007/s10514-011-9228-1.

Haschke, R., Steil, J. J., Steuwer, I., & Ritter, H. (2005). Task-oriented quality measures for dextrous grasping. In *Proc. Conference on Computational Intelligence in Robotics and Automation*.

Hillenbrand, U., & Roa, M. A. (2012). Transferring functional grasps through contact warping and local replanning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Hsiao, K., Kaelbling, L., & Lozano-Prez, T. (2011). Robust grasping under object pose uncertainty. *Autonomous Robots*, *31*, 253–268. doi:10.1007/s10514-011-9243-2.

Kehoe, B., Matsukawa, A., Candido, S., Kuffner, J., & Goldberg, K. (2013). Cloud-Based Robot Grasping with the Google Object Recognition Engine. http://goldberg.berkeley.edu/pubs/ICRA-Cloud-Grasping-May-2013.pdf. Accessed 22 May 2014.

Li, Z., & Sastry, S. (1987). Task oriented optimal grasping by multifingered robot hands. In *ICRA* (pp. 389–394).

Ling, H., & Jacobs, D. (2007). Shape classification using the inner-distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(2), 286–299. doi:10.1109/TPAMI.2007.41.

Manis, R., & Santos, V. (2011a) Characteristics of a three-fingered grasp during a pouring task requiring dynamic stability. In *Proc. Neural Control of Movement Ann Mtg*.

Manis, R., & Santos, V. (2011b). Multi-digit coordination during a pouring task that requires dynamic stability. In *Proc Ann Mtg Amer Soc Biomech*.

Miller, A. T., & Allen, P. K. (2004). Graspit a versatile simulator for robotic grasping. *IEEE Robotics and Automation Magazine*, *11*(4), 110–122.

Papazov, C., & Burschka, D. (2010). An efficient ransac for 3d object recognition in noisy and occluded scenes. In *ACCV* (pp. 135–148).

Popovic, M., Kraft, D., Bodenhagen, L., Baseski, E., Pugeault, N., Kragic, D., et al. (2010). A strategy for grasping unknown objects based on co-planarity and colour information. *Robotics and Autonomous Systems*, *58*(5), 551–565.

Prats, M., Sanz. P., & del Pobil, A. (2007). Task-oriented grasping using hand preshapes and task frames. In *IEEE International Conference on Robotics and Automation* (pp. 1794–1799).

Rosales, C., Ros, L., Porta, J. M., & Suarez, R. (2010). Synthesizing Grasp Configurations with Specified Contact Regions. *The International Journal of Robotics Research*,. doi:10.1177/0278364910370218.

Sahbani, A., & El-Khoury, S. (2009). A hybrid approach for grasping 3d objects. In *IROS* (pp. 1272–1277).

Saxena, A., Driemeyer, J., Kearns, J., & Ng, A. Y. (2007). Robotic grasping of novel objects. *Advances in Neural Information Processing Systems*, *19*, 1209–1216.

Song, D., Huebner, K., Kyrki, V., & Kragic, D. (2010). Learning task constraints for robot grasping using graphical models. In *IROS* (pp. 1579–1585). doi:10.1109/IROS.2010.5649406.

Varadarajan, K., & Vincze, M. (2011a) Knowledge representation and inference for grasp affordances. In Crowley, J., Draper, B., Thonnat, M. (Eds.) Computer Vision Systems, Lecture Notes in Computer Science (vol. 6962, pp. 173–182). Berlin: Springer. doi:10.1007/978-3-642-23968-7_18.

Varadarajan, K., & Vincze, M. (2011b). Object part segmentation and classification in range images for grasping. In *15th International Conference on Advanced Robotics (ICAR)* (pp. 21–27). doi:10.1109/ICAR.2011.6088647.

Ying, L., Fu, J., & Pollard, N. (2007). Data-driven grasp synthesis using shape matching and task-based pruning. *IEEE Transactions on Visualization and Computer Graphics*, *13*(4), 732–747. doi:10.1109/TVCG.2007.1033.