Auton Robot (2014) 36:51–65
DOI 10.1007/s10514-013-9366-8

Learning of grasp selection based on shape-templates

Alexander Herzog · Peter Pastor · Mrinal Kalakrishnan · Ludovic Righetti · Jeannette Bohg · Tamim Asfour · Stefan Schaal

Received: 2 March 2013 / Accepted: 17 August 2013 / Published online: 11 September 2013

Abstract The ability to grasp unknown objects still remains an unsolved problem in the robotics community. One of the challenges is to choose an appropriate grasp configuration, i.e., the 6D pose of the hand relative to the object and its finger configuration. In this paper, we introduce an algorithm that is based on the assumption that similarly shaped objects can be grasped in a similar way. It is able to synthesize good grasp poses for unknown objects by finding the best matching object shape templates associated with previously demonstrated grasps. The grasp selection algorithm is able to improve over time by using the information of previous grasp attempts to adapt the ranking of the templates to new situations. We tested our approach on two different platforms, the Willow Garage PR2 and the Barrett WAM robot, which have very different hand kinematics. Furthermore, we compared our algorithm with other grasp planners and demonstrated its superior performance. The results presented in this paper show that the algorithm is able to find good grasp configurations for a large set of unknown objects from a relatively small set of demonstrations, and does improve its performance over time.

Electronic supplementary material The online version of this article (doi:10.1007/s10514-013-9366-8) contains supplementary material, which is available to authorized users.

A. Herzog (✉) · L. Righetti · J. Bohg · S. Schaal
Autonomous Motion Department, Max-Planck Institute for Intelligent Systems, 72076 Tübingen, Germany
e-mail: aherzog@tuebingen.mpg.de

P. Pastor · M. Kalakrishnan · L. Righetti · S. Schaal
Computational Learning and Motor Control Lab, University of Southern California, Los Angeles, CA 90089, USA

T. Asfour
High Performance Humanoid Technology Lab (H2T), Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany

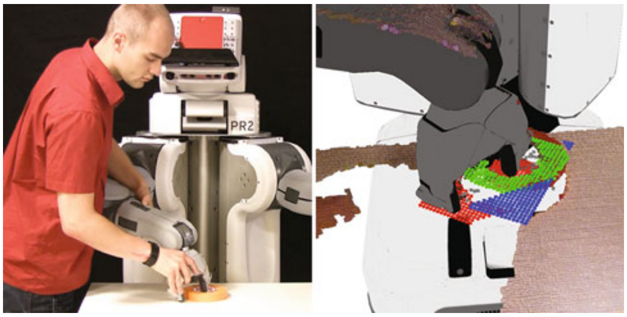


Fig. 1 User demonstrates a feasible grasp to the PR2 robot (*left*). Extracted template heightmap and gripper-pose (*right*)

Keywords Model-free grasping · Grasp synthesis · Template learning

1 Introduction

Autonomous robotic grasping is one of the pre-requisites for personal robots to become useful when assisting humans in daily life. Seemingly easy for humans, it still remains a very challenging task for robots. An essential aspect of robotic grasping is to automatically choose an appropriate grasp configuration given an object as perceived by the sensors of the robot. Developing an algorithm that provides such grasp hypotheses reliably is hard given the large variety in size and geometry of objects (e.g. household objects, see Table 1).

Grasp planning in unstructured environments has been studied extensively in the past. Planners that require exact object models describing their size and geometry usually involve sampling methods to generate feasible grasp configurations and choose the one that maximizes a grasp quality metric (e.g. the metric proposed by Ferrari and Canny

1992). Usually the quality of grasps is evaluated using simulators such as GraspIt! (Miller and Allen 2004), OpenRave (Diankov and Kuffner 2008) or OpenGRASP (Leon et al. 2010; Ulbrich et al. 2011) that can execute grasps given perfect knowledge of the object and its environment. Evaluating all possible combinations of object model and gripper configuration is computationally intractable. Algorithms have been proposed that sample this space e.g. by approximating object shape with geometric primitives (Rubio et al. 2010; Miller et al. 2003; Goldfeder et al. 2007; Przybylski and Asfour 2010). Other geometric approaches have been presented that instead of solving the grasp selection problem for each object in isolation, aim at transferring successfully executed grasps to novel object models (Ratliff et al. 2007; Curtis et al. 2008; Goldfeder and Allen 2011).

All these approaches are commonly used to generate a large database of objects with pre-computed grasps offline. At run-time, target object and database models are matched according to some similarity measure. The grasps associated to the best matching model can then be applied to the target object. It is important to note that objects in cluttered environments will usually only be partially perceived due to visibility constraints induced by a particular viewing direction or by occluding objects. Matching this partial sensor data against geometric models in the grasp database is difficult and still an open research problem. For the case of known objects in the scene, the problem of object recognition and pose estimation has been adressed e.g. in Huebner et al. (2009) and Papazov et al. (2012). The shape matching problem for the case of familiar objects is even harder, i.e. for objects that are not the same but similar to the objects in the database. To this end, an interesting approach has been proposed by Goldfeder and Allen (2011). The authors generate a codebook of shape features extracted from synthetic depth maps that are generated from a number of viewpoints around a large set of object models. Given data from a real sensor, shape features are extracted and matched against the synthetically generated one in the database. The most similar one is then retrieved along with the associated grasps .

We propose to completely avoid relying on geometrical object models and work directly with sensor data. Thereby we avoid the problem of matching synthetically generated with real-world sensor data. To do so, we developed a local shape descriptor that can be automatically extracted from real sensor data along with the associated grasp pose. We show that a rather small set of such shape templates is sufficient to grasp a large variety of unknown objects.

All the aforementioned approaches for grasp synthesis (Rubio et al. 2010; Miller et al. 2003; Goldfeder et al. 2007; Przybylski and Asfour 2010; Ratliff et al. 2007; Curtis et al. 2008; Goldfeder and Allen 2011) allow to generate grasps offline for a large number of objects. However, these grasps are usually computed using idealistic assumptions about the contact interaction between the object and the hand. Balasubramanian et al. (2012) showed that grasps that are highly ranked according to the often used ϵ-metric by Ferrari and Canny (1992) do not perform well when tested in the real world. Grasps that were demonstrated by humans were significantly more successful. Following this study, we show that a small number of example grasps generated from user demonstrations and linked to local object shape can be generalised to grasp a large variety of objects.

There are some model-free approaches that propose geometrical features which indicate good grasps as for example in Klingbeil et al. (2011), Richtsfeld and Zillich (2008), Popović et al. (2011), Hsiao et al. (2010) and Stückler et al. (2011). Hsiao et al. (2010) developed an algorithm that searches among feasible top and side grasps to maximize a set of criteria such as the amount of perceived object mass between the finger tips of a parallel jaw gripper. These approaches generate a ranked list of grasp hypotheses suitable for execution on a robot. Their advantage is that no supervised learning is required, but on the other hand ranking of grasp hypotheses is fixed and does not adapt over time.

Methods have been proposed that learn the success rate of grasps given a descriptor extracted from sensor data. These require large amounts of labeled training data which can be acquired either by evaluating grasps on a real robotic system (Montesano and Lopes 2009; Detry et al. 2010; Erkan et al. 2010) or from synthetic sensor data with a manually chosen label (Boularias et al. 2011; Bohg and Kragic 2009; Stark et al. 2008; Saxena et al. 2008). Learning in these approaches is often restricted to 3D grasp points or 6D grasp poses rather than to a full gripper pose and finger configuration. Thus, grasps are executed for a fixed finger configuration which restricts the types of objects that the robot is able to grasp. The algorithm of Saxena et al. (2008) learns complete hand configurations rather than only 3D grasp points. Their approach learns grasp hypotheses from many synthetic images of objects annotated with appropriate grasp locations and recorded from several viewing angles. However, this method requires to work around the discrepancy of the synthetically generated images and images taken with real robots.In our algorithm sensor data is used directly to learn grasps.

Recently, interesting approaches have been presented that match object parts extracted from sensor data against new objects. Kroemer et al. (2012) define a part by weighted sum of Gaussians around points of the object's point cloud. The weights are modelled as an isotropic Gaussian centered at a manually-chosen subpart frame. Based on this, a kernel function is defined that evaluates the similarity between two subparts. This can then be used to find a so-called affordance bearing subpart on a new object. The selection of this part is improved over time by trial-and-error. Rather than grasp poses, this method generates full trajectories for a specific task. An evaluation is shown on only a few objects grasped at a handle. Detry et al. (2013) match hand sized point cloud parts, extracted from user demonstration, against new objects. They show impressive generalisation results on a set of household objects. However, their approach requires detailed object models during training. Hand poses can only be taught from a fixed set of finger configurations.

In a previous contribution (Herzog et al. 2012) we showed preliminary results on a novel grasp selection algorithm with following favorable characteristics:

- We propose a local grasp shape descriptor, which encodes hand-sized regions on the object that are suitable for grasping.
- Appropriate grasp poses together with finger configurations can be taught through kinesthetic teaching. Instances of the shape descriptors are then stored to describe the grasped object part (Fig. 1).
- Graspable object parts on unknown objects can be recognized by matching shape templates from demonstrations. Associated grasps can then be applied to the new situation.
- In addition our algorithm is able to autonomously improve the ranking of generated grasp candidates over time based on feedback from previous grasp executions.
- The approach is not restricted to a particular hand kinematic and can be applied in principle to any end-effector.

In this paper we will provide a more elaborate analysis comprising the following:

- The proposed method is described in more detail and a discussion on possible extensions of the algorithm is provided.
- We introduce a new distance measure and evaluate it together with the one proposed in our previous contribution.
- Experiments are extended with a comparison between our grasp planner and two state of the art approaches.

We exploit the assumption that the part of an object that is in contact with a hand is most important for the success of a grasp. Although, there might be more factors that influence the choice of a good grasp, (e.g. its surface, weight or inertia properties) we will show that local shape parts provide a major feature for successful grasp selection. For example, a pen can be grasped from the table with a strategy similar to that used to grasp a screwdriver of the same size. Compared to holistic shape representations, a part representation is capable of better generalization, because parts of the object that are not in contact with the hand can be ignored for matching. A hand-sized part captures larger regions than local descriptors (e.g. edge, corner or SIFT) and thus is more expressive and less likely to lead to poor matching performance.

Recently, templates have been successfully used to encode local regions of terrains enabling a quadruped robot to choose good footholds (Kalakrishnan et al. 2009). In Kalakrishnan et al. (2009), templates have been used to encode terrain heightmaps. In contrast, in our work, we use templates to encode object heightmaps that are observed from several approach directions. We store known object shapes represented by templates together with feasible grasp configurations in a library. To obtain good grasp hypotheses for novel objects, our algorithm samples and compares shape patches to those patches in the library and retrieves the associated grasp configuration for the best match. An initial set of object shapes can be acquired by demonstrating feasible grasp configurations for a particular set of objects to the robot and store them as a template associated to a grasp pose. A grasp configuration is given by a six DOF end effector pose as well as the joint configuration of the robots gripper.

In Sect. 2 we explain our algorithm. The results are presented in Sect. 3. We discuss our approach in Sect. 4 and finish with a conclusion in Sect. 5.

2 Template-based grasp selection

In the following, we explain the single components of our planner as they are illustrated in Fig. 2. In Sect. 2.1, we describe the proposed object part descriptor, in the following referred to as the grasp heightmap. We compute a set of candidate heightmaps on an object as explained in Sect. 2.2. Section 2.3 describes how heightmaps are extracted in detail. Specific instances of the descriptor are stored as templates into a library. In Sect. 2.4 we show how positive examples are learned from user demonstration. Two distance metrics between candidate heightmaps and library templates are proposed in Sect. 2.5. The ability of our algorithm to learn from failed grasp attempts is described in Sect. 2.6.

2.1 Grasp heightmaps

Our shape descriptor, the grasp heightmap, is directly extracted from point clouds as captured by a 3D sensor, e.g. a Microsoft Kinect. We assume that the object lies on a flat support surface to simplify segmentation of the object from the background. To extract our shape descriptor, we find a tangential plane on the object point cloud and measure the distances between a rectangular region on this plane and points from the point cloud. This will be described in more detail in the next section. The tangential plane is parameterized by its normal which we call we describe the proposed object part *height-axis*. We rasterize the rectangular region and store the measured distances, so called *height values*, into its tiles (see Fig. 3).Therefore, a heightmap also stores

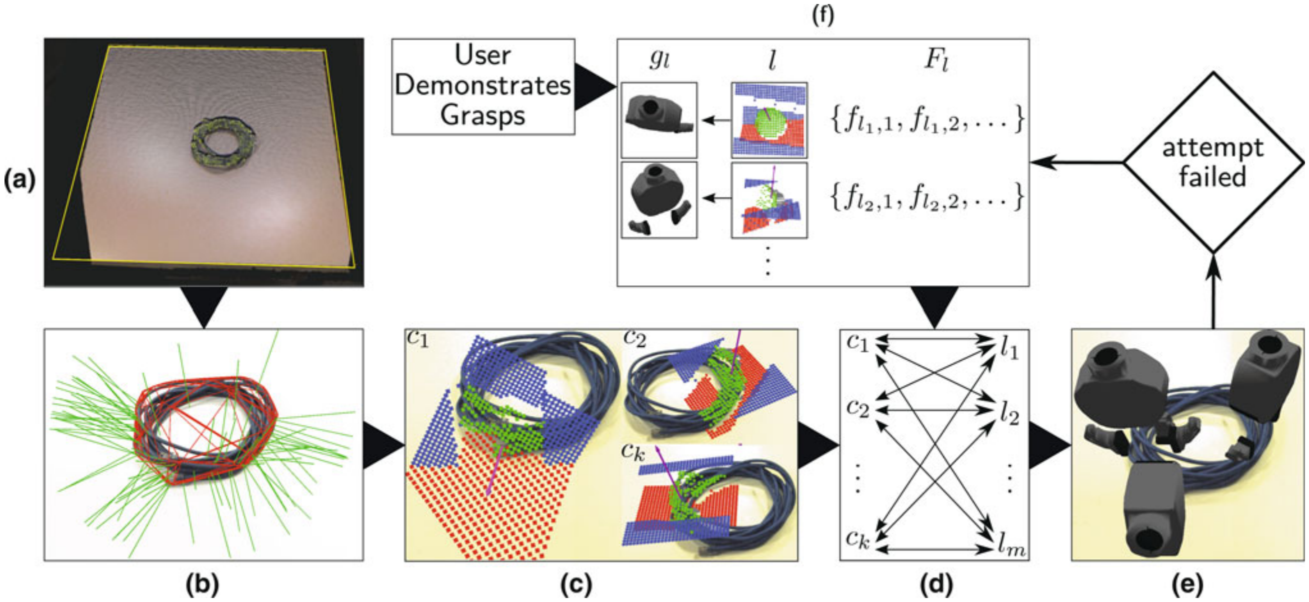


Fig. 2 Overview of the presented algorithm. **a** A 3D point cloud showing the object with background as it is perceived by the robot. The object (*yellow point cloud*) is segmented out from the table (*yellow rectangle*) for further processing. **b** The convex hull of the object point cloud (*red polygon mesh*) is used as approximation of the surface to compute a set of normals (*green lines*). These normals are used as height-axes in order to extract heightmaps. **c** Three heightmap candidates extracted from the object relative to various (*purple*) height-axes. **d** Candidates are matched against template heightmaps in the grasp library to look-up good grasp configurations. **e** The matching cost provides a ranking for the resulting hand configurations. If a grasp attempt fails, feedback is returned to the grasp library and incorporated in successive matching steps to adapt ranking. **f** New grasps are added by user demonstrations to the grasp library. Each template **l** is stored together with the hand configuration $\mathbf{g_l}$ and a set F_l of negative templates from failed grasp attempts (Color figure online)

height values measured from regions surrounding the object. We distinguish four different region types:

- Regions on the object *surface* describe the shape of the grasped object part and are to be enclosed by the hand.
- For a grasp configuration it is desirable to avoid premature contacts with the object. Thus, fingers need to sweep into *void* regions.
- Collisions between hand and *background* should be avoided.
- Regions that cannot be determined as one of the previous types due to *occlusion* are also encoded.

Hence each tile t in a grasp heightmap T contains a height value labeled with a region type and is defined as

$$t \in T = \mathbb{R} \times \mathbb{S}, \; \mathbb{S} = \{\mathfrak{s}, \mathfrak{v}, \mathfrak{o}, \mathfrak{b}\},$$

where $\mathfrak{s}, \mathfrak{v}, \mathfrak{o}$ and $\mathfrak{b}$ stand for *surface*, *void*, *occlusion* and *background*. A grasp heightmap with a granularity of $n \times n$ tiles is then defined as a vector $\mathbf{c} \in T^{n^2}$ with height value components $c_i \in \mathbb{R}, \; i = 1 \ldots n^2$ and region types $\hat{c}_i \in \mathbb{S}$. The compact representation in form of a two dimensional grid has the advantage that matching can be performed quickly. The width of heightmaps is set according to the hand in use. We associate a hand pose to a heightmap in order to clamp exceeding height values to the bounding box of the hand. Heightmaps are extracted from perceived point clouds as illustrated in Fig. 3 and described in more detail in the following.

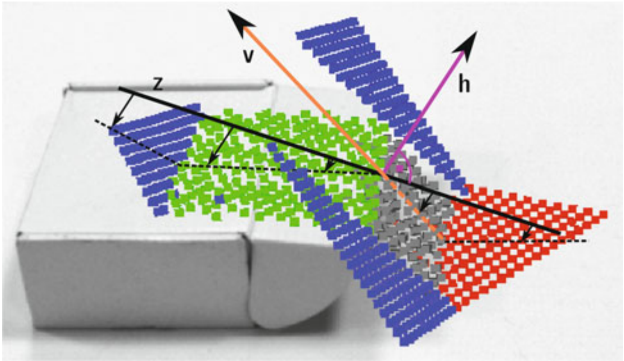


Fig. 3 One example heightmap extracted from the point cloud of a *card box*. Height values (*small black arrows*) are measured relative to a tangent plane (indicated by the *black line* Z). The plane is perpendicular to the height-axis **h**. Tiles of type *surface* (*green*) are extracted from the object surface. Tiles of type *background* (*red*) describe the height between Z and the table. *Void* regions (*blue*) are clamped to the limits of the bounding box of the hand. Tiles of type *occlusion* store the height of the upper bound of an occluded region. They depend on the viewing direction **v** and the detected object *surface* (Color figure online)

2.2 Heightmap search space

Before height values can be extracted, it is necessary to define the heightmaps coordinate frame relative to the object, i.e. a height-axis at a point on the object and a rotation around it. To make the algorithm computationally efficient, we limit the search space of these coordinate frames. Therefore, we compute height-axes that are perpendicular to the object surface. A rough approximation of the surface is obtained by computing the convex hull of the object point cloud. For each polygon of the resulting mesh we use the center to compute a height-axis as shown in Fig. 2b. The rotation around a height-axis is discretized into r steps resulting in r heightmaps per polygon. Hence, the number of computed heightmaps is the number of polygons of the convex hull times r. We ignore polygons with normals pointing away from the viewpoint. As they are extracted from a convex hull, they are on the backside of the object relative to the viewpoint. Approaching the object from such a direction usually results in a kinematically infeasible grasp.

2.3 Extracting heightmaps

Given a coordinate frame for a heightmap relative to the object point cloud, the next step is to extract the height values and region types as shown in Fig. 3. For this, the point cloud is transformed into the heightmap's coordinate frame. We assume that a segmentation of the object point cloud is given and points are labeled as either object or background. In the following, the computation is done in three steps.

First, surface regions are labeled. We iterate through all object points $\mathbf{p}$ and project them down onto the plane Z to find the according tile. The distance between $\mathbf{p}$ and Z is the height-value and it is stored together with region type $\mathfrak{s}$. An example is point $\mathbf{p_1}$ in Fig. 4. If several points get projected onto the same tile, the highest value is stored.

In the second step, we iterate through the remaining tiles and classify them as either *void* or *occlusion*. We say that a point $\mathbf{p}$ occludes a straight line S, if $\mathbf{p} \in \mathbf{conv}(\{\mathbf{v}\} \cup S)$, where $\mathbf{conv}(\cdot)$ is the convex hull of a set of points and $\mathbf{v}$ is the robot's view point. In practice we apply a threshold on the distance between $\mathbf{p}$ and $\mathbf{conv}(\{\mathbf{v}\} \cup S)$. We consider a tile as void if the straight line S, defined by the tile center and the height-axis, is not occluded by any point from the object. For instance consider S_1 in the figure. It is not occluded by any of the points. In that case we set the tile to *void* and bound the height value by the gripper bounding box. If a point, however, occludes S, we compute the line of sight, defined by the position of the camera $\mathbf{v}$ and the occluding point $\mathbf{p}$. In our example in Fig. 4 we have $\mathbf{p_1} \in \mathbf{conv}(\{\mathbf{v}\} \cup S_2)$ and the according line of sight is the dashed line crossing $\mathbf{v}$. The distance between Z and the intersection of the line of sight with S is stored in the tile. Again, for each tile the highest value is stored.

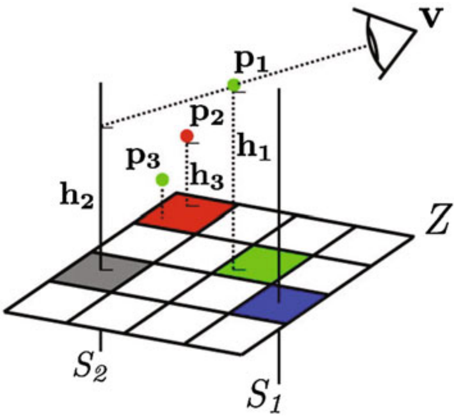


Fig. 4 Illustration of heightmap extraction procedure as described in Sect. 2.3

In the last step *background* tiles are set (e.g. $\mathbf{p_2}$). Similar to the first step, we compute distances between Z and points labeled as background. We overwrite tiles with the according height and type *background*, if the new height value is bigger than the current one (e.g. $\mathbf{p_2}$ in the figure is higher than $\mathbf{p_3}$). In our case, we can assume the table to be a surface and speed up computation. For each tile we just need to find the intersection of the *table* and S to compute the height value. However, in general the background can have any shape.

For our experiments, we use a naive implementation where for each tile, we iterate through all points. However, using ray-tracing on a voxel grid could speed-up the procedure.

2.4 Learning good grasp configurations from demonstration

An initial set of grasp configurations can be learned from demonstration through kinesthetic teaching. To add a good grasp configuration to the library, the user is required to move the robot's hand to a favorable grasp configuration as shown in Fig. 1. The proposed method then automatically extracts the heightmap on the convex hull, which is closest to the robots palm. It is stored as positive template together with the demonstrated finger configuration and hand pose relative to this template into the library. Thus, each library entry consists of an extracted template heightmap $\mathbf{l} \in T^{n^2}$, an associated hand pose $\mathbf{g_l} \in \mathbb{R}^6$ and a finger joint configuration. Decoupling the hand pose from the heightmap frame allows us to reduce the search space for heightmap frames as described in Sect. 2.2 without restricting possible hand poses. We want to emphasize that for extending the robot's grasp repertoire, these grasp configurations can be taught by a user who is not required to have any expert knowledge.

2.5 Distance metrics

A crucial part of our template matching approach is computing the distance between heightmaps. As discussed in more detail in Sect. 4, it is non-trivial to design a metric that weighs in the grasp-relevant differences between heightmaps. We designed and evaluated two different metrics. The first distance measure was proposed in our previous work (Herzog et al. 2012) and is described in Sect. 2.5.1. The second distance measure is described in Sect. 2.5.2. For both methods we refer to candidate heightmaps as $\mathbf{c} \in T^{n^2}$ and to library templates as $\mathbf{l} \in T^{n^2}$ (associated to a hand pose $\mathbf{g_l}$). The bounding box of the gripper relative to $\mathbf{g_l}$ is expressed in the coordinate frames of $\mathbf{c}$ and $\mathbf{l}$ in order to bound height values. Tiles with exceeding values are set to type *void*. A comparison of both distance measures is provided in Sect. 3.

2.5.1 Distance measure A

We define the distance measure between a candidate heightmap $\mathbf{c}$ and a library template $\mathbf{l}$ according to (i) overlap of region types and (ii) geometric information of the heightmaps. For the latter we use a weighted ℓ_1 distance over the height values:

$$s_\sigma = \sum_i^{n^2} W_{\hat{c}_i,\hat{l}_i} |c_i - l_i|, \quad W_{\hat{c}_i,\hat{l}_i} = \begin{cases} w_1 & \text{if } \hat{c}_i = \hat{l}_i = \mathfrak{s} \\ w_2 & \text{else} \end{cases} \tag{1}$$

where $W_{\hat{c}_i,\hat{l}_i}$ puts a higher weight on surface regions and a lower weight on the remaining region combinations, i.e. $w_1 > w_2$. We take additional information from region segmentation into account by counting overlapping surface regions in the two heightmaps. The number of tiles in a heightmap $\mathbf{c}$ labeled as region type $j \in \mathbb{S}$ is expressed as

$$\#_j(\mathbf{c}) = \left| \{ i \mid \hat{c}_i = j, \ i = 1 \ldots n^2 \} \right| . \tag{2}$$

For overlapping tiles of a type $j \in \mathbb{S}$ in two heightmaps $\mathbf{c}, \mathbf{l}$ we write $\#_j(\mathbf{c}, \mathbf{l}) = \left| \{ i \mid \hat{c}_i = \hat{l}_i = j, \ i = 1 \ldots n^2 \} \right|$. This allows us to define an overlap measure

$$o_\sigma = \frac{\max\{\#_\mathfrak{s}(\mathbf{c}), \#_\mathfrak{s}(\mathbf{l})\}}{\#_\mathfrak{s}(\mathbf{c}, \mathbf{l})} . \tag{3}$$

The objective is to maximize overlap of surface regions. Since heightmaps may represent differently sized objects, we normalize for the number of surface tiles in the numerator of Eq. (3). The distance between $\mathbf{c}$ and $\mathbf{l}$ then is defined as

$$\sigma(\mathbf{c}, \mathbf{l}) = o_\sigma s_\sigma . \tag{4}$$

The lower $\sigma(\mathbf{c}, \mathbf{l})$ is, the more similar are $\mathbf{c}$ and $\mathbf{l}$ to each other according to geometrical shape and region overlap. Note that for Distance Measure A we only distinguish between *surface* regions and the remaining three types. We will now present a measure that takes all four types into account.

2.5.2 Distance measure B

In Eq. (3) overlap is inversely proportional in the number of overlapping surface tiles $\#_\mathfrak{s}(\mathbf{c}, \mathbf{l})$. Thus for heightmaps with small surface regions (e.g. small objects) the region distance o_σ varies strongly with overlapping surface tiles. This makes it less likely to match small object parts to each other, when heightmaps are slightly misaligned. To reduce sensibility to noise we tried a region distance that is additive in the number of overlapping tiles,

$$o_\kappa = \frac{1}{n^2} \sum_{j \in \mathbb{S}} k_j (\max\{\#_j(\mathbf{c}), \#_j(\mathbf{l})\} - \#_j(\mathbf{c}, \mathbf{l})),$$

with region dependent weights $k_j \in \mathbb{R}$. Here we take overlap of all region types into account while o_σ only accounts for surface regions. For the shape distance presented in this paragraph we use a ℓ_1 distance

$$s_\kappa = \frac{1}{n^2} \sum_i^{n^2} |c_i - l_i| .$$

We combine the two measures to a distance

$$\kappa(\mathbf{c}, \mathbf{l}) = k_0 s_\kappa + o_\kappa , \tag{5}$$

where k_0 is a weighting parameter that is necessary to account for different magnitudes in the two summands. The advantage of this distance measure to the one presented in Eq. (4) is that there are no multiplicative terms between shape and region types. This way parameters are easier to chose. In Sect. 3 we evaluate the performance of the distance measures presented in this section.

2.6 Matching cost

The objective of the matching cost function, described in this section, is to express the dissimilarity between a candidate heightmap and a positive template. Additionally, it has to incorporate experience from failed grasp attempts in form of negative templates. Heightmaps from failed attempts are considered for feedback by extending the grasp library such that a set $F_\mathbf{l} \subset T^{n^2}$ of failures is added to each positive library template $\mathbf{l}$. If a candidate heightmap was matched to $\mathbf{l}$ and the related hand pose $\mathbf{g_l}$ led to a failed grasp attempt, the candidate heightmap is added as negative template to $F_\mathbf{l}$. Feedback is not applied globally to all library templates as it is related to a particular hand pose $\mathbf{g_l}$. Example matches are shown in Fig. 5.

In the following we would like to describe a function that evaluates the matching cost between a candidate heightmap

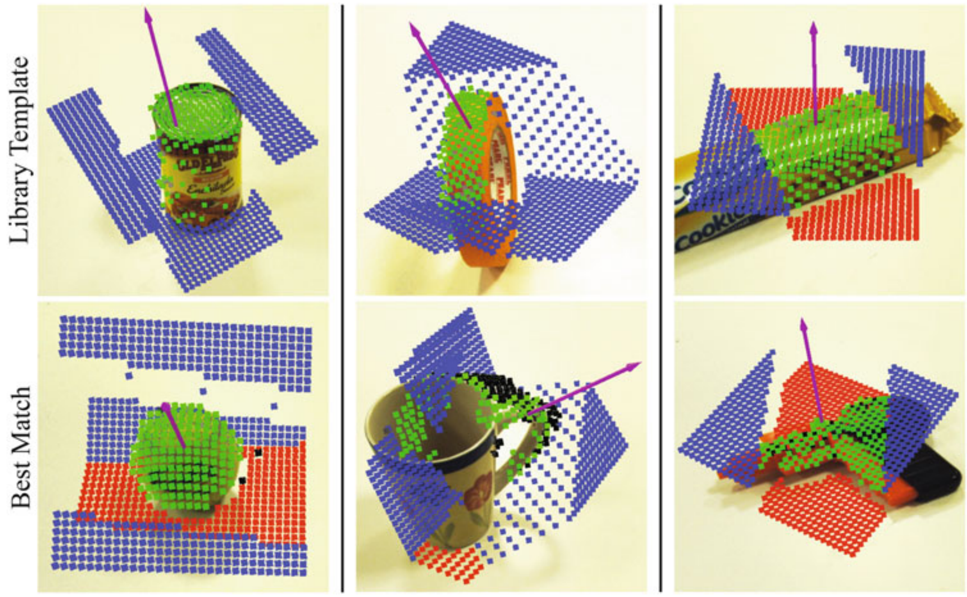


Fig. 5 The *bottom row* shows candidate templates from unknown objects; the *top row* shows the corresponding best match to the templates contained in the library. The *purple arrows* show the height-axes of the templates (Color figure online)

and a positive template under consideration of negative examples. It is composed of three distances

$$\sigma(\mathbf{c}, \mathbf{l}),$$
$$\beta(\mathbf{c}, F_{\mathbf{l}}) = \min\{\sigma(\mathbf{c}, \mathbf{f_i}) \mid \mathbf{f_i} \in F_{\mathbf{l}}\},$$
$$\gamma(\mathbf{l}, F_{\mathbf{l}}) = \min\{\sigma(\mathbf{l}, \mathbf{f_i}) \mid \mathbf{f_i} \in F_{\mathbf{l}}\},$$

where σ from Eq. (4) can be replaced by κ from Eq. (5). If $F_{\mathbf{l}} = \emptyset$, then $\beta = \gamma = \infty$ (initial state of the grasp library after demonstration). In addition to the distance between candidate and positive template, expressed by σ, the distance to the closest negative template β is taken into account. The objective is to avoid choosing candidates that are close to templates which led to failures in previous grasp attempts. γ is a robustness measure for $\mathbf{l}$. The higher it is, the more robust is $\mathbf{l}$ to shape variation. These three competing measures are combined in the following function

$$\mathrm{m}(\mathbf{c}, \mathbf{l}, F_{\mathbf{l}}) = \frac{\sigma}{\left[1 - \exp\left(-\mathrm{k}_1\beta^2\right)\right]\left[1 - \exp(-\mathrm{k}_2\gamma^2)\right]}, \qquad (6)$$

which yields a matching cost for a candidate heightmap $\mathbf{c} \in T^{n^2}$ with a positive library template $\mathbf{l} \in T^{n^2}$ by also incorporating feedback from negative templates $F_{\mathbf{l}}$. The lower $\mathrm{m}(\mathbf{c}, \mathbf{l}, F_{\mathbf{l}})$, the better $\mathbf{c}$ and $\mathbf{l}$ match.

$\mathrm{f}_{\mathrm{k}_i}(x) = 1 - \exp(-\mathrm{k}_i x^2)$ is a sigmoid shaped function of $x \in \mathbb{R}$ with shape parameters k_i, $i = 1, 2$ (see Fig. 6). It was chosen because of two favorable characteristics. The first one can be described by the following example. Let σ and β have small values for some $\mathbf{c}, \mathbf{l}, \mathbf{f}_i$, which means that the candidate is similar to a user demonstration but also to a negative template that has previously led to a failed grasp. In this case, the matching cost should have a high value. By choosing $\mathrm{f}_{\mathrm{k}_i}(x)$ to be of sigmoidal shape, the desired behavior of $\lim_{\beta \to 0} \frac{\sigma}{\mathrm{f}_{\mathrm{k}_i}(\beta)} = \infty$ is achieved. Furthermore, $\mathrm{f}_{\mathrm{k}_i}(x)$ does not grow significantly for $x \gg 0$. This is favorable, because increases in already high values for β or γ should have less influence on the matching cost than changes in σ. The functions $\mathrm{f}_3(\beta)$ and $\mathrm{f}_{10}(\gamma)$ are shown in Fig. 6. In our experiments we realized that $\mathrm{k}_1, \mathrm{k}_2$ can be set to 1 when using the distance measure in Eq. (5).

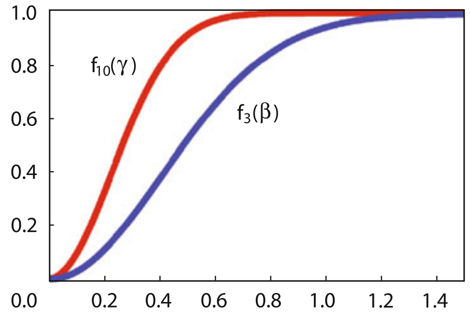


Fig. 6 Functions $f_3(\beta)$ and $f_{10}(\gamma)$ used for the matching function m

This matching is applied for each pair of candidate and library template under consideration of negative templates. The resulting matching costs $\mathrm{m}(\mathbf{c}, \mathbf{l}, F_{\mathbf{l}})$ provide a ranking among candidate heightmaps. Hand poses that come with matched templates are transformed into the coordinate frames of candidate heightmaps. Starting with the highest ranked grasp the planer excludes kinematicaly infeasible grasps until it finds a valid one for execution. Let $|C|$, $|L|$ be the number of candidate heightmaps and library templates, respectively. Then, the matching process requires $|C| \times |L|$ evaluations of Eqs. (4) or (5) (depending on the choice of distance measure). Additionally, the distances need to be evaluated $|C| \times \sum_{\mathbf{l} \in L} |F_{\mathbf{l}}|$ times to compute $\beta(\mathbf{c}, F_{\mathbf{l}})$. Adding a new negative template requires updating $\gamma(\mathbf{I}, F_{\mathbf{I}})$, which is one evaluation of Eqs. (4) or (5). To add a new user demonstration, no updates of the matching costs need to be performed. It should be possible to further reduce the library for better scalability (e.g. with clustering techniques), but our experiments showed that already a relatively small library of templates performs well on a challenging test-set of objects.

3 Results

The grasp selection algorithm was evaluated on two different robots with different hand designs. On each of them, two kinds of experimental setups were used to evaluate (i) the quality of retrieved grasp hypotheses and (ii) the ability to improve such hypotheses from trial and error. We compared our approaches to two baseline methods. The proposed algorithm was evaluated using the two distance measures presented in Sect. 2.5. To initialize the grasp library, a user demonstrated grasps on a training set of objects. During the experiments objects from a test set were placed in front of the robot on a table one at a time. A ranked list of grasp hypotheses was created by the planner. The best that was reachable was chosen and applied to the test object. We evaluated our algorithm on different data sets consisting of objects with a large variety.

3.1 Experimental setup

The two robots used for our experiments (see Fig. 7) have very different hand designs. They are described in the remainder of this paragraph together with details about the implementation of our algorithm.

3.1.1 Willow garage PR2

The Willow Garage PR2 is a robotic platform, used for research on service robotics (Bohren et al. 2011). For 3D point cloud extraction, we used a head-mounted Microsoft Kinect. (see Fig. 7). The proposed algorithm was integrated in the *pr2 tabletop manipulation pipeline*.[1] The robot has a parallel jaw gripper consisting of two fingers and a single DOF for controlling the aperture of the fingers (see Fig. 8).

3.1.2 Barrett WAM robot

Furthermore, the proposed approach was evaluated on a Barrett WAM robot[2] with a WAM arm and the Barrett Hand BH280 (see Fig. 7). A head-mounted Asus Xtion sensor has been used to obtain point clouds. The Barrett Hand is different from the PR2 gripper. It has three fingers, each of them having one actuated DOF. An additional DOF controls the spread between the left and right fingers resulting in four actuated DOFs (see Fig. 9). The grasp planner was embedded into the manipulation architecture, presented by Righetti et al. (2013). The gripper pose was adapted by hybrid force and position control using the 6-axis force torque sensor in the wrist of the hand. A desired force of −1 N along the approaching direction was set to guarantee contact between the palm and the object. A desired torque of 0 Nm for pitch and yaw was set to adapt for slightly miss-oriented grasp poses which lead to premature contact with the table. These settings were set globally for all the experiments on the WAM robot.

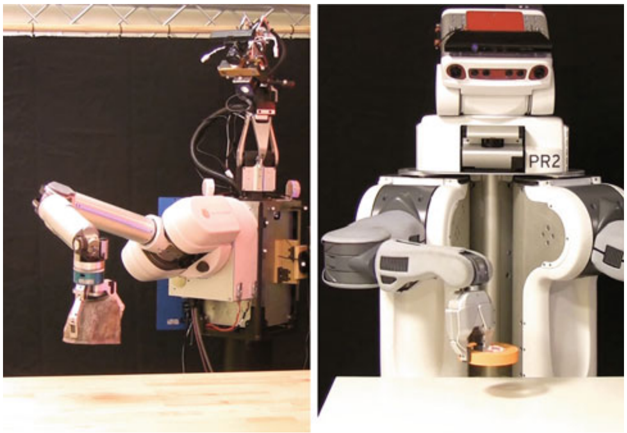


Fig. 7 The Barrett WAM robot with a WAM arm and a Barrett Hand together with a head-mounted Asus Xtion (*left*) and the Willow Garage PR2 with a pair of parallel jaw gripper and a head-mounted Microsoft Kinect (*right*)

3.1.3 Implementation

The grasp planner was integrated in the planning and control framework of both robots using ROS[3](Robot Operating System). Parts of the *tabletop object detector* (e.g. described in Hsiao et al. 2010) were used to segment the object from the table and execute grasps. We use a simple segmentation algorithm that fits a plane into the point cloud and subtracts points that are close to it. Then a mean shift algorithm is used to cluster the remaining points based on their projection onto the fitted plane. An interface to PCL[4] (Rusu and Cousins 2011) (Point Cloud Library) in ROS was used to compute convex hulls of point clouds as described in Sect. 2.2. An implementation of the algorithm is publicly available.[5]

[1] http://www.ros.org/wiki/pr2_tabletop_manipulation_apps/.

[2] http://thearmrobot.com/aboutRobot.html.

[3] http://www.ros.org.

[4] http://www.pointclouds.org.

[5] https://github.com/usc-clmc/usc-clmc-ros-pkg.

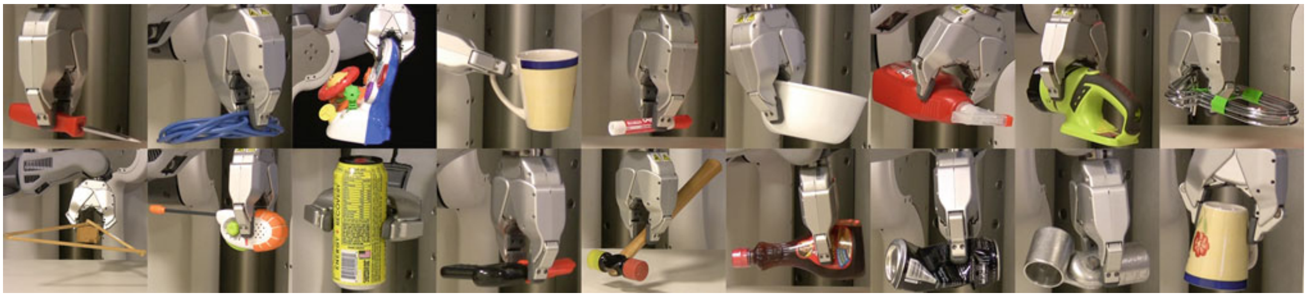


Fig. 8 Subset of the achieved grasps on the test set on the PR2

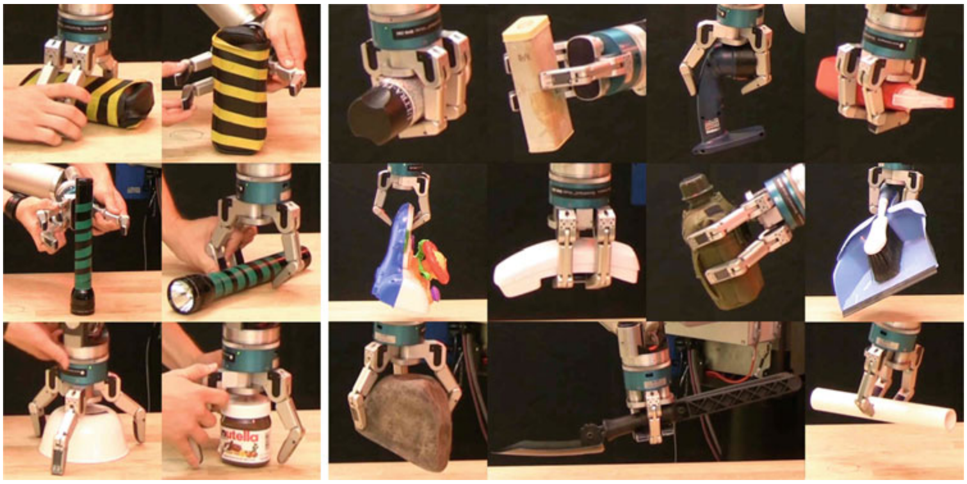


Fig. 9 All demonstrated grasps on the WAM robot (*left*) and a subset of the achieved grasps (*right*). Depending on the size and shape of the training objects, the user demonstrated finger spread and finger positions accordingly. When e.g. grasping the shovel, the robot recognized the cylindrical middle part being similar to the flashlight and applied the demonstrated gripper configuration (*middle row*, *second from left*) successfully. On the other hand grasping the canteen required the fingers to be widened as the body of the object is bigger than that of the shovel. In this case the algorithm successfully applied the grasp taught on the leather bag (*upper row*, *second from left*). Situation dependent the algorithm chose different finger configurations suited for grasping the unknown objects

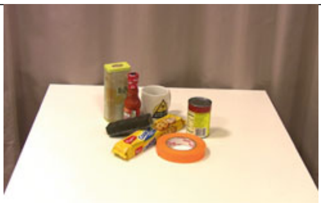
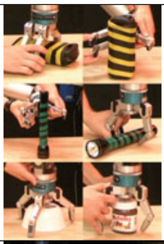
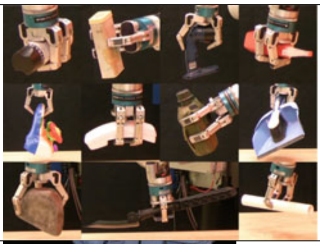
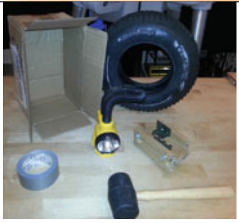
We evaluated the algorithm with the two different distance measures as presented in Sect. 2.5. When using Distance Measure A from Eq. (4) we set $w_1 = 50,\ w_2 = 12$. This way we give more importance to tiles that encode object surface. Parameters in the matching function in Eq. (6) were set to $\mathrm{k}_1 = 3,\ \mathrm{k}_2 = 10$. For Distance Measure B in Eq. (5) we set $k_{\mathfrak{v}} = k_{\mathfrak{b}} = 1.0,\ k_{\mathfrak{s}} = 2.0$ to give more importance on overlap of surface regions, $k_{\mathfrak{o}} = 0$ to express missing information about occluded regions and $k_0 = 500$ to make the two summands of Eq. (5) have roughly the same magnitude. The values for the matching function in Eq. (6) were simply $\mathrm{k}_1 = \mathrm{k}_2 = 1$. The granularity of heightmaps was set to $n = 30$ and we computed $r = 16$ heightmaps for each height-axis. Although hand designs of the two robots are quite different, the only parameter that had to be adapted was the size of templates, because it was fitted to the bounding box of the gripper. For the PR2, templates were of size 15 cm × 15 cm and for the Barrett Hand they were of size 25 cm × 25 cm. For all the other parameters, we used the same set of values for both robots.

3.2 Generalization from demonstrated grasps

For the first experiment a user demonstrated 15 grasps on a training set of seven objects to the PR2 as described in Sect. 2.4. For this gripper design, maximum aperture size was chosen for all demonstrations. The trained algorithm was tested on a set of 38 differently shaped objects (see Table 1a). They were placed in different orientations to vary the viewpoint on the object. After grasping and lifting the object, the robot waited a few seconds to see if the object slipped due to a bad grasp. A particular grasp was considered a success if the robot was able to lift the object off the table, indicating a stable grasp. Using the presented grasp planner, the robot achieved a success rate of 87 %, i.e. 83 out of 95 grasp hypotheses lead to a stable grasp. A subset of the achieved grasps are shown in Fig. 8 as well as in the video supplement.[6] The results indicate that the template representation

[6] A short summary of the grasping experiments is also shown at http://www.youtube.com/watch?v=C7_xVxu8_RU.

Table 1 Summary of conducted experiments

	Training set	Test set	Method	Success(%)	Fail	Infeasible
a)			Distance A, no feedback	83(87%)	12	
b)			Distance A, no feedback	44 (70%)	17	2
c)			Distance B, no feedback	43 (72%)	11	6
d)			Hsiao et. al.	37 (62%)	11	12
e)			Distance A, with feedback	41 (82%)	9	
f)			Distance B, with feedback	20 (83%)	4	0
g)			PCA planner	17 (71%)	7	0

(a)–(d) show experiments on the PR2 and (e)–(g) on the WAM Robot. We use our approach together with either of the two distance measures described in Sects. 2.5.1 and 2.5.2. Baseline approaches different from the one presented in this paper are evaluated as well: (d) Hsiao et al. (2010) and (g) the "PCA Planner" as described in the text. The rightmost column encounters attempts when the planner could not find any feasible grasp. This was caused mostly by big objects for which the intersection of feasible approach directions and the robots workspace did not include good grasps

is able to generalize from only 15 demonstrations to a large variety of objects. It also indicates that the algorithm is robust against different viewpoints on the object since the pose of the objects was varied. A few attempts failed due to slightly miss-oriented grasp poses or hypotheses that collided with the table in a way that premature contact with the object could not be avoided.

A different training-set set was chosen to demonstrate six gripper configurations on four objects to the Barrett Hand as shown in Table 1e. The robot executed five grasps on each of 10 test-objects resulting in 50 grasp executions. Different from the previous experiment, feedback from a user was exploited after each trial. Whenever a grasp succeeded, the object pose was varied. After unsuccessful attempts we placed the object in the same pose as before for the next out of the five trials. The algorithm computed grasp hypotheses that resulted in 41 out of 50 successful trials as shown in Table 2. In case of an unsuccessful grasp attempt, the robot achieved to grasp the object in the same orientation after at most two additional trials using feedback from the previously failed attempts. The results showed that depending on the object to grasp, different finger configurations where required which was computed correctly most of the time by our planner.

3.3 Comparison of different approaches

In the experiments in Table 1b–d, we evaluated the performance of our algorithm using the two different distance measurements as described in Sect. 2.5 and compared the performance to the grasp planner presented by Hsiao et al. (2010).

Table 2 Grasp attempts on a WAM robot with Barrett Hand

Object	Success rate
Spray Can	4/5
Pipe	4/5
Flashlight	4/5
Shovel	3/5
Phone Handle	5/5
Toy Wheel	3/5
Box	4/5
Red Spray Bottle	5/5
Canteen	5/5
Duster with Pan	4/5
Overall	41/50

We added big objects to the data set comprising some boxes, file folders and a pelican case. This makes grasping more difficult, because the gripper can fit only parts of the objects which the grasp planner is required to find. Also these parts are often occluded. As can be read from Table 1, the results of the three methods all show a good performance on this difficult data set with a slight tendency towards a better performance by our approach. We noticed that the approach by Hsiao et al. works reliably on smaller objects, but often has problems to find feasible grasps on bigger objects, since it only considers a limited number of approach directions and the workspace of the robot is limited by the big objects. Splitting the results into bigger and smaller objects shows that their approach performs similar to ours on hand sized objects, whereas we see a performance difference when only big objects are considered. Also their planner is specific to the parallel jaw gripper of the PR2, where our planner is not specific to a hand design and can also be used with more complex hands. Although both distance measures (see Table 1b, c) perform equally well, we prefer distance measure B, because it has fewer parameters to be set and those are easily chosen.

For the WAM Robot, we implemented a simple grasp planner as a baseline that is similar to the one described by Stückler et al. (2011). We perform a principle component analysis on the object point cloud and pick the axis with the largest principle component. We place the desired gripper pose such that it encloses this axis at its center. We generate 32 grasps by rotating the grasp pose around the axis. Poses for which fingers are not colliding with the table are ranked higher. The results are shown in Table 1g. We executed four grasps on each of the test objects in different poses. A similar experiment was done with our approach (see Table 1f). Different to the "PCA Planner", we exploited feedback after each trial and in case of an unsuccessful attempt we made the robot grasp the object in the same pose for the next execution. Our method always achieved to compute a successful grasp after at most two failed attempts, summing up to 4 failed and 20 successful grasps. The baseline "PCA planner" performed surprisingly well on small objects that fit entirely in the robots hand. However, big objects (the tire and the carton box) required the robot to grasp specific parts of the objects rather than the center of the point cloud, such that our planner outperformed the baseline planner. It chose object parts that were similar to user demonstrations and resulted in successful grasps.

Overall these experiments showed that our planner performs at least as well as the two model-free baseline planners. However, we saw that for grasping bigger objects, that are graspable only on few sub parts of the object, our planner is able to make a better decision on where to grasp successfully. Also both baseline planners were specific to the robot hands, but our approach is suitable for both without re-parameterization. The good performance of both distance measures A, B indicates that our algorithm has some robustness towards the choice of similarity metric.

3.4 Improvement over time

In another experiment, we want to test if the grasp planner is able to improve over time. We ran a sequence of grasp executions on one object. After each grasp, the algorithm was informed about failure or success and the object was put in a different pose. On the PR2 a whiteboard marker and a spray can were used for this evaluation in two separate runs.

The robot tried to grasp the whiteboard marker 34 times in a row. Although grasping failed quite often in the beginning, the algorithm could increase the overall success rate over 30 percentage points as shown in Fig. 10. The first trials failed mainly because of grasp poses that led to premature contact with the object. However, for the last trials the PR2 tended to approach the object perpendicular to the table surface such that premature contact was avoided and the grasps succeeded.

The robot showed a similar improvement on the spray can. Here an additional difficulty was the lower part of the object which is bigger than the gripper and thus is not graspable. For this experiment we used distance measure B from Eq. (5) and the training set shown in Table 1b. First the spray can was put in 8 poses standing and then in eight poses lying. We repeated poses from failed grasps up to three times. After the first run we had a success rate of 55.6 % and improved to 88.9 % in the second run. The results show that the presented algorithm is able to improve ranking over time using a simple form of feedback.

The same experiment was conducted with the Barrett WAM where the robot tried to grasp a rock shaped object 40 times in a row (see third image from left in the bottom row of Fig. 9). As shown in Fig. 10, success rate was increased from 30 to 60 %. For the first trials the robot chose the grasp configuration that was taught on the bowl as shown in the

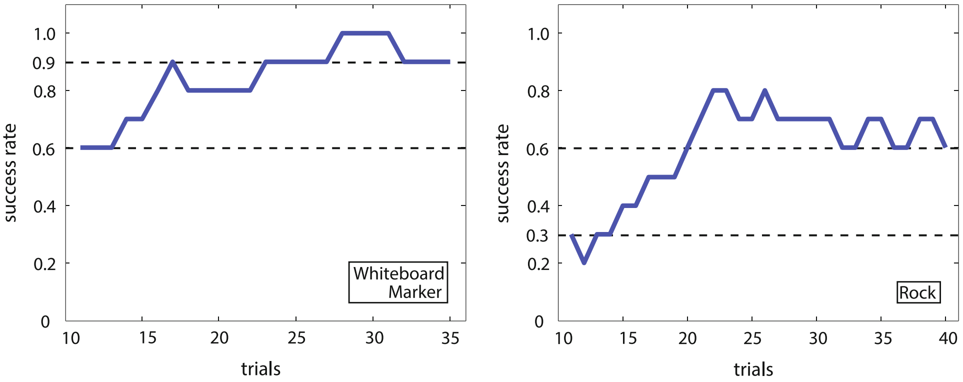


Fig. 10 Success rate with increasing number of grasp trials using feedback from failed attempts. After each grasp the ratio of successful grasps in the previous ten executions is computed. Results on a PR2 and a whiteboard marker (*left*) and a WAM robot and a rock (*right*)

bottom left image in Fig. 9. This made the rock slip away quite often. However, the ranking of grasp hypotheses was changed over time in a way that the configuration demonstrated on the leather bag was chosen more often. This led to more stable grasps and increased the success rate.

The time for computing a ranked list of grasps was between 5 and 30 s, when the algorithm was executed together with other tasks on the on-board computer on the PR2. However, our implementation of the planner leaves space for computational speedup.

4 Discussion

In the following we discuss the advantages and drawbacks of our approach and present future research directions to improve the algorithm.

4.1 Distance metric

A crucial aspect of our template-based approach is the distance metric in the space of heightmaps T^{n^2}. Unfortunately, designing a metric by hand that leads to the desired behavior is non-trivial. Height values and region types cannot be treated independently. Consider for instance parts of two heightmaps labeled as *void*. For both, height values are bounded by the same bounding box and thus lead to perfect shape matching. If one would match shape without taking into account region types, this would lead to a preference of heightmaps with large *void* parts. Besides the shape of two templates it is required to measure overlap of regions. However, overlap is not necessary for all types. While it is desired to match *surface* regions, it is not necessary to have overlap of background regions. *Background* only needs to be avoided by the gripper but not necessarily matched. A similar situation occurs with occlusions. It is not desirable to achieve an overlap between regions of which we have no shape information. For these reasons, we analyzed two distance metrics that vary in the way heightmap shape and region are coupled. Finding a distance measure that takes the characteristics of heightmaps into account was an important component to make our approach work. Ideally a distance measure should be learned, e.g. with supervised learning. Heightmap tuples considered as similar could be collected by a user and exploited for learning. This has to be investigated in future work.

4.2 Using experience from success

In our algorithm, we acquire positive templates from user demonstrations and improve matching by including negative templates in the library. In the following, we would like to discuss why we exploit experience from failures but do not try to improve matching using successful grasp executions.

The core assumption underlying the proposed algorithm is that similar objects can be grasped in the same way. Of course, the success of a grasp depends also on other physical characteristics of the hand and object, but for the development of this algorithm and the following discussion, we focus on geometrical properties. Let $\mathbf{c}$ be a candidate heightmap and $\mathbf{l}$ a library template that describe shapes of object regions. Furthermore, let $\mathbf{g_l}$ be a hand configuration known to result in a successful grasp on $\mathbf{l}$. Then, we can formalize the aforementioned assumption as

$$(\mathbf{c}, \mathbf{l} \text{ are similar}) \wedge (\text{execution of } \mathbf{g_l} \text{ on } \mathbf{l} \text{ succeeds}) \Rightarrow$$
$$(\text{execution of } \mathbf{g_l} \text{ on } \mathbf{c} \text{ succeeds}).$$

When a grasp fails on candidate heightmap $\mathbf{c}$, i.e. ¬(execution of $\mathbf{g_l}$ on $\mathbf{c}$ succeeds) and we know that it succeeded on template $\mathbf{l}$, i.e. (execution of $\mathbf{g_l}$ on $\mathbf{l}$ succeeds), then from the rules of induction, we imply that $\mathbf{c}$ and $\mathbf{l}$ were not similar. Therefore, we can re-write the above expression as

$$\neg(\text{execution of } \mathbf{g_l} \text{ on } \mathbf{c} \text{ succeeds}) \Rightarrow \neg(\mathbf{c}, \mathbf{l} \text{ are similar}). \quad (7)$$

Equation 7 implies that a candidate heightmap $\mathbf{c}$ that led to a failed grasp attempt can be used to learn that $\mathbf{c}$ and $\mathbf{l}$ are not similar. However, we cannot exploit the logic proposition to infer information about similarity from successful trials. To clarify this, consider the following example: let $\mathbf{c}$

describe the top of a cup that is placed normally on a table. Let $\mathbf{l}$ capture the bottom of a cup that is placed upside down on the table. The heightmaps describe two different shapes, but the same overhead grasp can be successfully applied in both situations. Thus, the implication that the heightmaps are similar would be wrong and positive feedback would be misleading. Hence similarity matching cannot be improved from positive feedback and this is why in the proposed approach only feedback from failed attempts is used.

Instead of including positive heightmaps for improvement of matching, one could think of acquiring positive templates autonomously. After a positive execution, a template could be stored along with user demonstrations into the library. However, this might lead to an undesirable effect. A newly added template might be very similar to an existing entry with the difference that the existing entry might already have a set $F_{\mathbf{l}}$ of negative feedback attached to it. Since the newly added template does not have negative feedback attached, previously experienced bad candidates can again be matched with a low cost. To make this approach work, clustering techniques may be applied to avoid storing the same template multiple times. This would also provide better scalability.

5 Conclusion

In this paper we proposed a model-free grasp planning algorithm. We introduced a local shape descriptor that encodes good grasp regions of object parts for grasp selection. Our algorithm can extract these shape descriptors, called grasp-templates, directly from sensor data. A set of positive examples can be learned from demonstration. These grasp-templates are matched against perceived point-clouds of unknown objects to generate a list of grasp hypotheses. We showed that a relatively small set of demonstrations is sufficient to compute good grasps on a set of objects with a large variety of shapes. This indicates that the proposed shape templates are able to capture point-cloud features relevant for model-free grasping. Furthermore, we showed that both experimental platforms, the Willow Garage PR2 robot and the Barrett WAM robot, were able to improve ranking of grasps through trial-and-error. Both robots showed good performance with the same set of parameters. A comparison of the algorithm with two different planners showed that the presented approach performs equally well. However, we want to stress that our approach (1) is independent of the hand kinematics (2) can automatically improve its grasp ranking (provided feedback) and (3) can be easily retrained by the user. Finally, our approach performs better than two State-of-the-Art algorithms on objects that can only be grasped at specific parts (this is for example the case for big objects).

We believe that the proposed shape descriptor is a good representation for grasping and is worth being developed further for application on more complex objects and manipulation tasks. Although we achieved a good success rate in our experiments, our goal in future work is to apply learning techniques in order to find an even better similarity measure between heightmaps. Further, the grasp library is to be improved by clustering heightmaps for better scalability and matching performance. An additional thread in future work is to include tactile sensing in order to prevent slippage and further increase the stability of grasps. This work was focusing on finding stable grasps, but in order to use an object for manipulation a task specific selection of hand pose and finger configuration is required, which will be addressed in future work as well.

Acknowledgments The work described in this paper was partially conducted within the EU Cognitive Systems project GRASP (IST-FP7-IP-215821) funded by the European Commission and the German Humanoid Research project SFB588 funded by the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft). Alexander Herzog received support from the InterACT - International Center for Advanced Communication Technologies. This research was supported in part by National Science Foundation grants ECS-0326095, IIS-0535282, IIS-1017134, CNS-0619937, IIS-0917318, CBET-0922784, EECS-0926052, CNS-0960061, the DARPA program on Autonomous Robotic Manipulation, the Army Research Office, the Okawa Foundation, the ATR Computational Neuroscience Laboratories, and the Max-Planck-Society.

References

Balasubramanian, R., Xu, L., Brook, P. D., Smith, J. R., & Matsuoka, Y. (2012). Physical human interactive guidance: Identifying grasping principles from human-planned grasps. *IEEE Transactions on Robotics*, *28*(4), 899–910.

Bohg, J., & Kragic, D. (2009). Grasping familiar objects using shape context. In *International conference on advanced robotics* (pp. 1–6).

Bohren, J., Rusu, R.B., Gil Jones, E., Marder-Eppstein, E., Pantofaru, C., Wise, M., et al. (2011). Towards autonomous robotic butlers: Lessons learned with the PR2. In *IEEE international conference on robotics and automation* (pp. 5568–5575).

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* (pp. 1548–1553).

Curtis, N., Xiao, J., & Member, S. (2008). Efficient and effective grasping of novel objects through learning and adapting a knowledge base. In *IEEE international conference on robotics and automation* (pp. 2252–2257).

Detry, R., Başeski, E., Popovic, M., Touati, Y., Krueger, N., Kroemer, O., et al. (2010). Learning continuous grasp affordances by sensorimotor exploration. In *From motor learning to interaction learning in robots* (pp. 451–465). Heidelberg: Springer.

Detry, R., Ek, C. H., Madry, M., & Kragic, D. (2013). Learning a dictionary of prototypical grasp-predicting parts from grasping experience. In *IEEE international conference on robotics and automation*.

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

Erkan, A., Kroemer, O., Detry, R., Altun, Y., Piater, J., & Peters, J. (2010). Learning probabilistic discriminative models of grasp affordances under limited supervision. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 1586–1591).

Ferrari, C., & Canny, J. (1992). Planning optimal grasps. In *IEEE international conference on robotics and automation* (pp. 2290–2295).

Goldfeder, C., & Allen, P. (2011). Data-driven grasping. *Autonomous Robots*, *31*, 1–20.

Goldfeder, C., Allen, P. K., Lackner, C., & Pelossof, R. (2007). Grasp planning via decomposition trees. In *IEEE international conference on robotics and automation* (pp. 4679–4684).

Herzog, A., Pastor, P., Kalakrishnan, M., Righetti, L., Asfour, T., & Schaal, S. (2012). Template-based learning of grasp selection. In *IEEE international conference on robotics and automation* (pp. 2379–2384).

Hsiao, K., Chitta, S., Ciocarlie, M., & Jones, E. G. (2010). Contact-reactive grasping of objects with partial shape information. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 1228–1235).

Huebner, K., Welke, K., Przybylski, M., Vahrenkamp, N., Asfour, T., et al. (2009). Grasping known objects with humanoid robots: A box-based approach. In *International conference on advanced robotics* (pp. 1–6).

Kalakrishnan, M., Buchli, J., Pastor, P., & Schaal, S. (2009). Learning locomotion over rough terrain using terrain templates. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 167–172).

Klingbeil, E., Rao, D., Carpenter, B., Ganapathi, V., Ng, A. Y., & Khatib, O. (2011). Grasping with application to an autonomous checkout robot. In *IEEE international conference on robotics and automation* (pp. 2837–2844).

Kroemer, O., Ugur, E., Oztop, E., & Peters, J. (2012). A kernel-based approach to direct action perception. In *IEEE international conference on robotics and automation* (pp. 2605–2610).

Leon, B., Ulbrich, S., Diankov, R., Puche, G., Przybylski, M., Morales, A., et al. (2010). Opengrasp: A toolkit for robot grasping simulation. In *2nd international conference on simulation, modeling, and programming for autonomous robots*.

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

Miller, A. T., Knoop, S., Christensen, H. I., & Allen, P. K. (2003). Automatic grasp planning using shape primitives. In *IEEE international conference on robotics and automation* (pp. 1824–1829).

Montesano, L., & Lopes, M. (2009). Learning grasping affordances from local visual descriptors. In *IEEE international conference on development and learning* (pp. 1–6).

Papazov, C., Haddadin, S., Parusel, S., Krieger, K., & Burschka, D. (2012). Rigid 3D geometry matching for grasping of known objects in cluttered scenes. *The International Journal of Robotics Research*, *31*(4), 538–553.

Popović, M., Kootstra, G., Jørgensen, J. A., Kragic, D., & Krüger, N. (2011). Grasping unknown objects using an early cognitive vision system for general scene understanding. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 987–994).

Przybylski, M., & Asfour, T. (2010). Unions of balls for shape approximation in robot grasping. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 1592–1599).

Ratliff, N., Bagnell, J., & Srinivasa, S. S. (2007). Imitation learning for locomotion and manipulation. In *IEEE-RAS international conference on humanoid robots* (pp. 392–397).

Richtsfeld, M., & Zillich, M. (2008). Grasping unknown objects based on 2.5D range data. In *Proceedings of IEEE international conference on automation science and engineering CASE* (pp. 691–696).

Righetti, L., Kalakrishnan, M., Pastor, P., Binney, J., Kelly, J., Voorhies, R., et al. (2013). An autonomous manipulation system based on force control and optimization. *Autonomous Robots Journal Special Issue: Autonomous Grasping and Manipulation*. doi:10.1007/s10514-013-9365-9.

Rubio, O., Huebner, K., & Kragic, D. (2010). Representations for object grasping and learning from experience. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 1566–1571).

Rusu, R. B., & Cousins, S. (2011). 3D is here: Point Cloud Library (PCL). In *IEEE international conference on robotics and automation* (pp. 1–4).

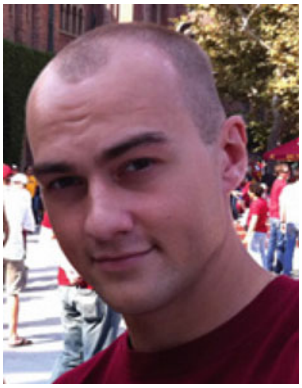
Saxena, A., Driemeyer, J., & Ng, A. Y. (2008). Robotic grasping of novel objects using vision. *The International Journal of Robotics Research*, *27*(2), 157–173.

Saxena, A., Wong, L., & Ng, A. Y. (2008). Learning grasp strategies with partial shape information. In *AAAI conference on artificial intelligence* (pp. 1491–1494).

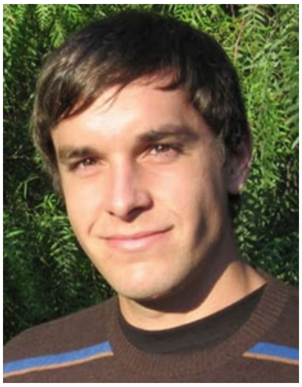
Stark, M., Lies, P., Zillich, M., Wyatt, J., & Schiele, B. (2008). Functional object class detection based on learned affordance cues. In *International conference on computer vision systems*. LNAI (Vol. 5008, pp. 435–444). Heidelberg: Springer.

Stückler, J., Steffens, R., Holz, D., & Behnke, S. (2011). Real-time 3D perception and efficient grasp planning for everyday manipulation tasks. In *European conference on mobile robots (ECMR)*. Örebro, Sweden.

Ulbrich, S., Kappler, D., Asfour, T., Vahrenkamp, N., Bierbaum, A., et al. (2011). The OpenGRASP benchmarking suite: An environment for the comparative analysis of grasping and dexterous manipulation. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 1761–1767).



Alexander Herzog studied Computer-Science at the Karlsruhe Institute of Technology, in Germany. He visited the Computational Learning and Motor Control Laboratory at the University of Southern California (USC) in 2011 to work on his diploma thesis. After receiving his Diploma in the same year, Alexander joined the Autonomous Motion Department at the Max-Planck Institute for Intelligent Systems (Tübingen, Germany) in 2012. His research interest comprises machine learning and control for manipulation and locomotion in humanoid robotics.



Peter Pastor is currently finishing his Ph.D. at the Computational Learning and Motor Control lab at the University of Southern California (USC). He received his diploma in Computer Science from the Karlsruhe Institute of Technology in 2008. He also received a M.Sc. in Computer Science from USC in 2010. His research interests include robotic grasping and manipulation as well as machine learning.

Mrinal Kalakrishnan received his M.S. in Computer Science in 2008 from the University of Southern California, where he is also currently a PhD candidate. His research focuses on the development of machine learning techniques to improve the performance of motion planners for autonomous robots in challenging domains such as rough terrain locomotion and autonomous grasping and manipulation. He is also interested in designing and learning controllers that use sensory and visual feedback loops to achieve compliant yet robust behavior.

Ludovic Righetti is a group leader at the Max-Planck Institute for Intelligent Systems (Tübingen, Germany) and has been a postdoctoral fellow at the Computational Learning and Motor Control Lab (University of Southern California) between March 2009 and August 2012. He studied at the Ecole Polytechnique Fédérale de Lausanne where he received a diploma in Computer Science in 2004 and a Doctorate in Science in 2008. His doctoral thesis was awarded the 2010 Georges Giralt PhD Award given by the European Robotics Research Network (EURON) for the best robotics thesis in Europe. His research focuses on the generation and control of movements for autonomous robots, with a special emphasis on legged locomotion and manipulation.

Jeannette Bohg is a research scientist at the Autonomous Motion Department, Max-Planck-Institute for Intelligent Systems in Tübingen, Germany. She holds a Diploma in Computer Science from the Technical University Dresden, Germany and a M.Sc. in Applied Information Technology with a focus on Art and Technology from Chalmers in Göteborg, Sweden. In 2011, she received her PhD from the Royal Institute of Technology (KTH) in Stockholm, Sweden. Her research interest lies at the intersection between robotic grasping and Computer Vision. Specifically, she is interested in the integration of multiple sensor modalities and information sources for enhancing the grasping and manipulation capabilities of robots.

Tamim Asfour is full professor at the Institute for Anthropomatics, Karlsruhe Institute of Technology (KIT). He is chair of Humanoid Robotics Systems and head of the High Performance Humanoid Technologies Lab (H2T). His major current research interest is high performance 24/7 humanoid robotics. He received his diploma degree in Electrical Engineering and his PhD in Computer Science from the University of Karlsruhe (TH) in 1994 and 2003 respectively. He has been active in the field of Humanoid Robotics for the last 14 years resulting in about 150 peer-reviewed publications with research focus on humanoid mechanoinformatics and design, grasping and dexterous manipulation, action learning from human observation and goal-directed imitation learning, active vision and active touch as well as software and hardware control architectures. He is developer of the ARMAR humanoid robot family.

Stefan Schaal is Professor of Computer Science, Neuroscience, and Biomedical Engineering at the University of Southern California, and a Founding Director of the Max-Planck-Insitute for Intelligent Systems in Tbingen, Germany. He is also an Invited Researcher at the ATR Computational Neuroscience Laboratory in Japan, where he held an appointment as Head of the Computational Learning Group during an international ERATO project, the Kawato Dynamic Brain Project (ERATO/JST). Before joining USC, Dr. Schaal was a postdoctoral fellow at the Department of Brain and Cognitive Sciences and the Artificial Intelligence Laboratory at MIT, an Invited Researcher at the ATR Human Information Processing Research Laboratories in Japan, and an Adjunct Assistant Professor at the Georgia Institute of Technology and at the Department of Kinesiology of the Pennsylvania State University. Stefan Schaal's research interests include topics of statistical and machine learning, neural networks, computational neuroscience, functional brain imaging, nonlinear dynamics, nonlinear control theory, and biomimetic robotics. He applies his research to problems of artificial and biological motor control and motor learning, focusing on both theoretical investigations and experiments with human subjects and anthropomorphic robot equipment.