Identifying relevant feature-action associations for grasping unmodelled objects

Thomsen, Mikkel Tang; Kraft, Dirk; Krüger, Norbert

Published in:
Paladyn. Journal of Behavioral Robotics

DOI:
10.1515/pjbr-2015-0006

Publication date:
2015

Document version
Submitted manuscript

Document license
Unspecified

Citation for pulished version (APA):
Thomsen, M. T., Kraft, D., & Krüger, N. (2015). Identifying relevant feature-action associations for grasping unmodelled objects. Paladyn. Journal of Behavioral Robotics, 6(1), 85-110. DOI: 10.1515/pjbr-2015-0006

Research Article Open Access

Mikkel Tang Thomsen*, Dirk Kraft, and Norbert Krüger

Identifying relevant feature-action associations for grasping unknown objects

Abstract: Action affordance learning based on visual sensory information is a crucial problem within the development of cognitive agents. In this paper, we present a method for learning action affordances based on basic visual features, which can vary in their granularity, order of combination and semantic content. The method is provided with a large and structured set of visual features, motivated by the visual hierarchy in primates and finds relevant feature action associations automatically. We apply our method in a simulated environment on three different object sets for the case of grasp affordance learning. For box objects, we achieve a 0.90 success probability, 0.80 for round objects and up to 0.75 for open objects, when presented with novel objects. In this work, we in particular demonstrate the effect of choosing appropriate feature representations. We could demonstrate a significant performance improvement by increasing the complexity of the perceptual representation. By that, we could present important insights in how the design of the feature space influences the actual learning problem.

Keywords: Human Vision, Affordance Learning, Cognitive Robotics

***Corresponding Author: Mikkel Tang Thomsen:** The Maersk Mc-Kinney Moller Institute, Faculty of Engineering, University of Southern Denmark, Niels Bohrs Allé 1, DK-5230 Odense M, Denmark, E-mail: mtt@mmmi.sdu.dk
Dirk Kraft: The Maersk Mc-Kinney Moller Institute, Faculty of Engineering, University of Southern Denmark, Niels Bohrs Allé 1, DK-5230 Odense M, Denmark, E-mail: kraft@mmmi.sdu.dk
Norbert Krüger: The Maersk Mc-Kinney Moller Institute, Faculty of Engineering, University of Southern Denmark, Niels Bohrs Allé 1, DK-5230 Odense M, Denmark, E-mail: norbert@mmmi.sdu.dk

1 Introduction

Identifying sensory features indicating action affordances is a crucial problem to be solved by cognitive agents since it allows for the identification of "action opportunities". A fundamental problem is the design of the perceptual feature space in which affordences emerge. This space can make the problem rather trivial (e.g., in case features that have a strong link to specific affordances are already provided). It can be also very difficult, when the link between affordances and actions can only be established by a high order combination of simple features (e.g., on the pixel level as in [1]).

It is in general acknowledged that for humans, vision is a strong cue for affordance generation. More than half of the primate's cortex is connected to visual tasks. As already pointed out in [2], the primate visual space is fundamentally of higher complexity compared to the action space. This in the first place concerns the dimensionality of visual information compared to a still rather low dimensionality of action parametrisation connected to the limited number of joints to be actuated.

The human visual system constitutes a deep hierarchy, covering a large number of complementary feature descriptors at different levels of granularity, different order and semantic abstraction (see Fig. 1 and [3] for a review of today's knowledge about the human visual system). More than $^2/_3$ of the visual cortex (the so called "occipital cortex") is associated to task-independent feature processing displayed as yellow areas in Fig. 1. In these areas, a rich set of visual feature descriptors covering different aspects of visual information such as colour, 2D and 3D shape as well as motion are extracted. At least at early stages of processing, this is done in largely separated processing streams [3].

As shown in Fig. 1, the level of abstraction of feature representation as well as the receptive field size increases (and by that the granularity of the features decreases) in this hierarchical process. Moreover, it is not only the features themselves but their combination that provide relevant affordance cues (see Fig. 2b). From search tasks it is known, that feature combinations up to third order

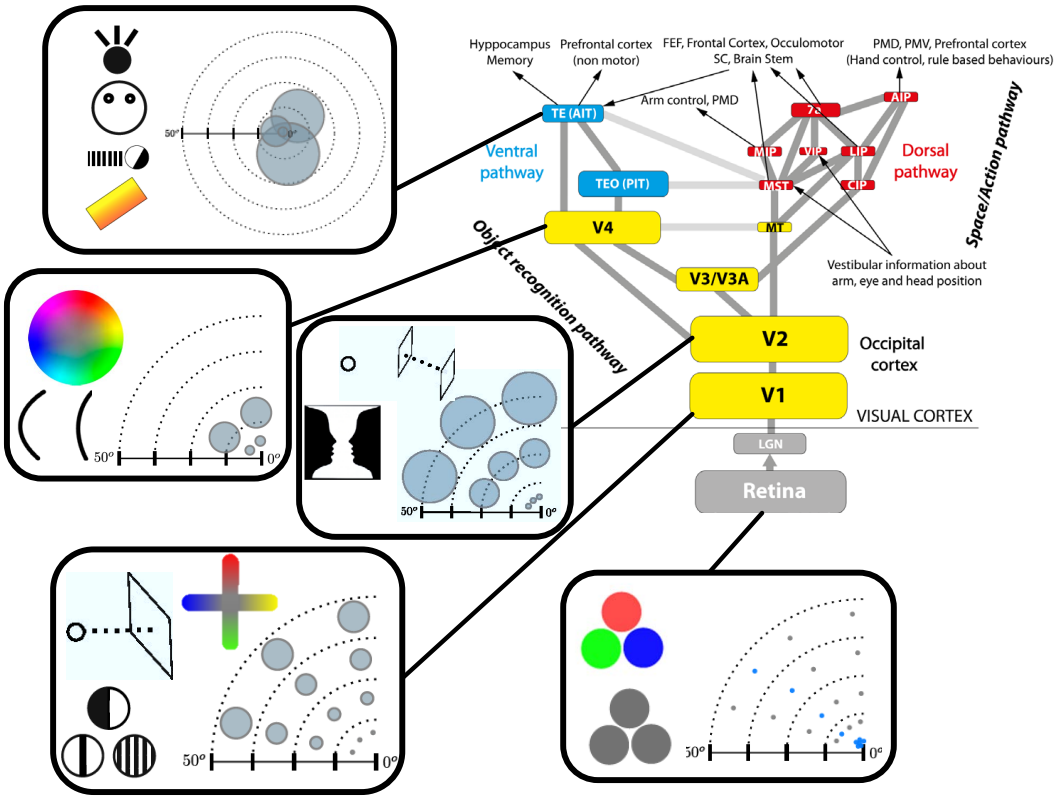


Fig. 1. The primate's visual cortex: The figure shows the deep hierarchical organization of the human visual system with the occipital cortex, the ventral and the dorsal pathway at the top right. For selected visual areas, the receptive field size of neurons are shown depending on where in the visual field their receptive field is positioned (right part of sub-figures) as well as some of the features that are assumed to be processed at the specific levels (left part of the subfigures). This figure uses material from [3] which we also refer to for further details.

are computed in parallel in the human visual system, which results in so called "pop-out effects" in visual search tasks [5]. Hence, finding structures relevant for affordance programming in this high dimensional space at appropriate levels of granularity, order and semantic abstraction poses one of the major problems for affordance learning.

In this paper, we investigate grasp affordances which are triggered by visual features of different order (see Fig. 3b), different granularity (see Fig. 3c) and semantic abstraction (see Fig. 3d). We are aware that the feature space we span is still of much lower complexity than what the human visual system provides in the occipital cortex. However, we investigate variation along three important dimensions of this feature space. Fig. 1 shows some of the mentioned aspects in the primate's brain. It shows how the granularity of receptive fields varies with the level of the hierarchy (in general neurons have smaller receptive fields at lower levels of the hierarchy). In general there also occurs an increase of semantic abstraction as well as order of processed features (e.g., absolute depth is coded at the level of V1 while relative depth — i.e., second order depth — is coded at the level of V2).

In this paper, we introduce a method for finding feature-action associations in a complex visual feature space. The method for affordance learning described in the paper is not specific for a certain type of affordances, it can be in principal applied to any parameterizable action affordance. In this paper we however choose grasping as an example problem because of three reasons. First, due to the general importance of grasping. Second, we can simplify the learning problem by neglecting certain feature dimensions provided by the human visual system. For example colour can be ignored as a relevant dimension for grasping. In this paper, we also neglect 2D shape information, which however might already be a more questionable design decision. A third reason for addressing grasping is that there exists already relevant related prior work: In [4] (see Fig. 2), grasp affordances have been manually designed as first and second order relations of visual entities (local sur-

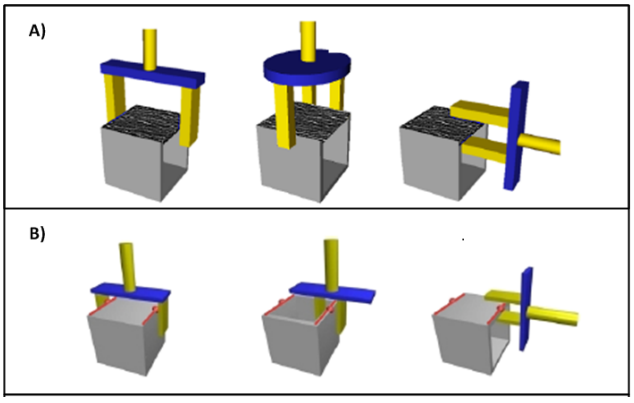


Fig. 2. Simple manually defined grasps: (A) Grasp affordances defined with respect to a single 3D surface feature (hence defined in respect to a first order feature relation), (B) Grasp affordances defined with respect to two 3D contours (hence defined in respect to second order feature relation). Source [4].

faces and 3D edges/contours). By that, we could already reach grasp performance of around 30% success. In [4], the grasp affordances however were defined "by hand" but in this paper, we aim at — besides improving performance — replacing such a manual design step by learning.

For this we want to explore the cross space of surface features and their combination, as shown in Figs. 3b–3d, and grasping actions. Fig. 4 shows how the variation of complexity of the input feature relates to the learning task. In Fig. 4a, left, we see a surface patch being related to a grasp. Learning grasp affordances with high success from this kind of weak feature is impossible, since actual successes would occur for the grasp at the right but not for the other two grasps shown in figure 4a. These cases are however indistinguishable when only one surface patch as a feature is used. When we extend the feature space to second order combinations of surface patches (see Fig. 4b), the grasp on the left would be also distinguishable as a non-successful one. However, it is impossible to learn that the middle grasp cannot be successful. However, when we also add the concept of a boundary and its direction to the surface patch (see figure 4c), the system is able to distinguish that only the right grasp can be successful. Similarly in this paper, we investigate the consequences for learning when we vary important dimensions of the feature space.

The algorithm we apply for that is a rather simple clustering method combined with a voting approach and part of the investigations is to explore the potential but also the limitations of such an approach. The complexities associated to our approach primarily stem from two sources:

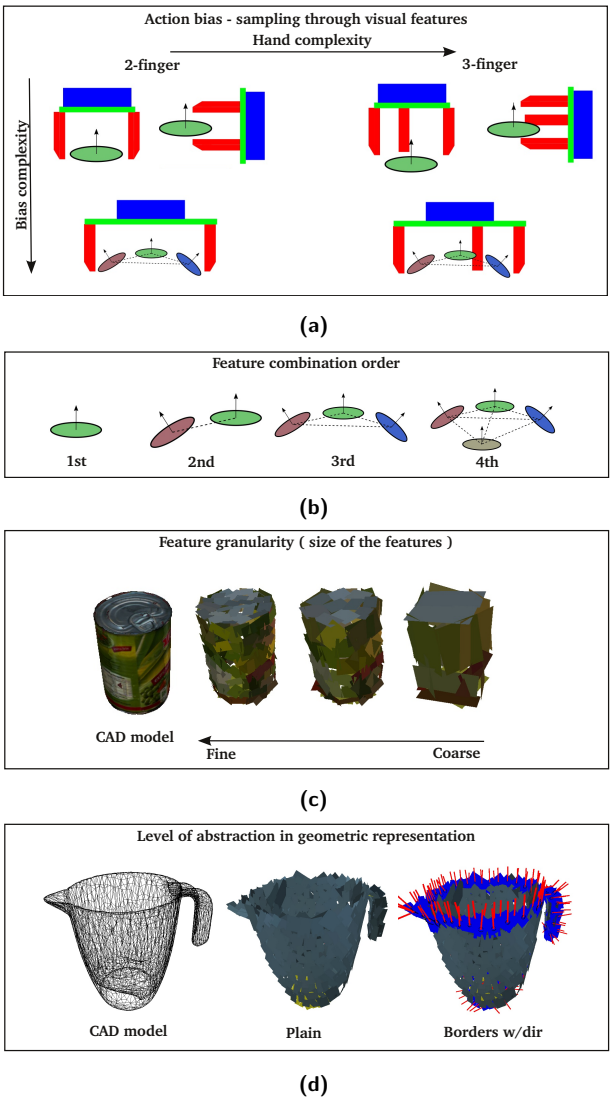


Fig. 3. Overview of different aspects of the perceptual and action space that are investigated throughout this paper. (a) shows an illustration of how we define different kinds of bias for grasping actions for a two or three finger hand. In (b), it is shown how we can increase complexity to the perceptual representation by means of combining multiple features into more elaborated structures. In (c), it is shown how we can increase/decrease the complexity of the perception side by changing the size of the features. In (d) it is shown how the level of abstraction of the feature representation can be raised by means of semantic (here adding a boundary label and a boundary direction to a surface patch).

Appropriate action bias: Non-successful actions are of limited usefulness for action affordance computation — although these can be used for sorting out non-interesting areas — and hence the system needs to be able to initially perform actions with a certain percentage of success likelihood. This can be achieved by introducing action bias (see Fig. 3a), e.g., by designing

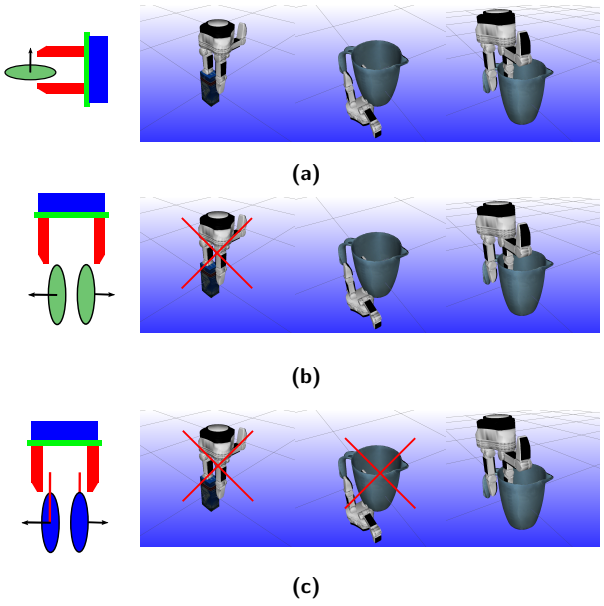


Fig. 4. Illustration of how different perceptual spaces can be used to limit the amount of grasp options. (a) shows a single feature grasp association which would not be able to distinguish between the three grasping situations on the left from which only the very left one leads to a success. (b) shows a second order-feature grasp association being rich enough to distinguish the left grasp situation as non successful. (c) shows a two-feature grasp association for which also the boundary direction (red line) is taken into account. This enriched features allows for distinguishing that only the very right situation leads to a success.

simple feature based heuristics that trigger actions with sufficient success likelihood (as in, e.g., [4]). In our case, we define rather weak biases that already lead to reasonable success likelihoods between 10–50% depending on the object class.

Feature space design: A further problem is to provide a feature space which covers features that are sufficiently correlated to successful actions. The feature space applied in this work does not provide feature coefficients that are independent. On the contrary, the feature space is highly structured: It provides geometric relations between surface patches which require appropriate parametrisations, careful choices of metrics as well as proper association of semantics.

Which features actually are relevant might depend significantly on the actual task and as we show most features will are highly uncorrelated to action successes and therefore insignificant. The richer the visual space we provide, the more complex the learning problem will be, since then feature actions need to be found in a larger space. This holds in particular when feature relations of high order are computed since this will very quickly lead to a dimensionality which cannot be explored exhaustively anymore (dimensionality explosion). As a way to reduce the learning problem, the semantic content of features can be increased (as indicated in Fig. 3d). This however usually requires the introduction of additional heuristics and by that would jeopardize the genericness of the approach. In our work, we show how the different design choices change the statistical distributions of particles in the feature space and by that the actual learning problem.

In this paper, we will describe how we approach the above mentioned complexities. We demonstrate how the affordance learning problem constitutes itself when important parameters such as the order of features, their granularity and their semantic complexity are varied.

In particular we show:

- that we can learn grasp affordances (as compared to manually defined affordances as in [4]).
- that the complexity of the feature space we span is of significant importance for the ability to learn affordances with a high rate of success.
- that we can improve the quality of affordance prediction by combining multiple features and adding semantic information.
- that the feature representations can also carry insufficient information to be considered as a good basis for grasp affordance learning.
- that we are able to identify grasp affordances for a set of different object types with a high likelihood of a success.

The paper is structured as follows: We relate our work to the state of the art in grasp affordance learning and other relevant work in section 2. The problem formulation our approach is based on is outlined and formalised in section 3. The approach to address the problem domain is presented in section 4. In section 5, the experimental settings are explained, whereas the experimental results are presented in section 6. Finally the paper is concluded in section 7.

2 State of the art

Visual triggered action affordance learning is important for the development of cognitive agents. Within the grasping community typically an object is grasped to be further manipulated. However affordance work like [6–8] take a more generic approach towards affordance

learning, with the aim of finding what visual features afford actions.

In [8], visual triggered affordance learning was investigated, with the purpose of finding what visual 2D feature cues of an object afford graspability. A supervised learning approach was employed, where a robot interacts with an object to discover graspability and link it to extracted feature cues. A different approach is adopted in [7], were affordance cue's are extracted from inspection of human interaction. By identifying which areas of an object are occluded by the human during a grasp/action, it is learned what local areas of an object afford grasping, e.g., a handle.

In our work, we take a similar generic approach towards affordance learning, but while in the authors of[7] learn object properties, e.g., graspability, we learn the coupling of visual features and actions, that enable a specific action. In that sense our work is more in line with the work in [6], where grasping points are learned from local visual descriptors, resulting in particular grasping points with associated probabilities.

Given the grasping application in our work, also approaches towards learning of grasping unknown objects are of interest. This topic has been extensively investigated due to its importance for robotic applications. For the problem of grasping unknown objects, two different strategies have generally been adopted, either feature based methods or shape based method. Examples of feature based approached are [4, 9–12], where a hand designed grasp hypothesis is proposed given a certain situation. These works stretch from grasp hypothesis based on a single or a combination of two simple features in [4] to grasp hypothesis based on a circle-fitting approach for cylindrical objects [12].

In contrast to feature based approaches, shape driven approaches like [1, 13–15], the agent has a shape model in its database with associated grasps. Then the shape is matched to new scene and in case a good match to a shape primitive is found, the grasps associated to this shape are performed. In [15], a set of prototypical object instances are captured with associated grasps from human demonstration and afterwards used for matching in novel situations. Other approaches like [14] and [13] approximates the object in terms of a oriented bounding box [14] or multiple bounding boxes [13] and then suggest grasps hypothesis based on the configuration of the bounding box. In a similar sense [16] decomposes an object into super quadratics to get an approximated object on which grasping can be performed. Another example of a model based approach is [17], where object shape, based on height maps extracted from 3D data and human demonstrated grasps, are learned and matched against new scene context.

For a broader overview of the grasping domain see [18], where data driven grasp synthesis of known, familiar and unknown objects are surveyed extensively, including some of the work mentioned here.

Our work is very much in line with the feature based approaches, as we introduce simple feature constellation with associated actions, to be used for action prediction. Our work can be seen as an extension to the work performed in [4], but with the advantage that we learn feature to action constellation by exploring different visual representations. In a recent work [19], deep learning techniques were used to learn a feature representation suitable for learning grasp affordances. The approach shows improved performance when compared to a previous work [20] utilising the same fundamental idea, but where the available feature representation was designed by hand. In contrast to [19], in our work we provide some kind of hierarchy to the learning algorithm which can than pick out promising candidates from this hierarchy. However, as discussed in the next paragraph, our approach can be seen as a step toward the learning of a deep hierarchy.

The focus on the underlying visual representation also links to work in non action domains, namely the work by the group of Ales Leonardis on learning hierarchical representations [21]. In this work, visual hierarchies are built up layer by layer. Each element of higher level entity is a combination of usually three elements of a lower level, where such combination represents a certain spatial arrangement of simpler features. The selection of such combinations is done unsupervised for lower levels of the hierarchy based on, e.g., the criterion of frequency of occurrence and in an supervised fashion at higher levels. Our work can be understood as a step towards such hierarchy building, since relevant particles derived in this paper (see equation 4) are also spatial constellations of simpler entities which could be used as input of a higher level of a deep hierarchical structure. Different from Leonardis' work, we however apply 3D entities instead of 2D entities and we also have task specific evaluation criteria already on rather early levels of processing.

3 Problem description and formalisation

The main topic we investigate throughout this paper is the cross-space between perceptual features and actions. We explore how different aspects of the visual representation can provide relevant information for predicting action affordances in a reliable way.

3.1 Formalisation

To be able to perform these investigations, we initially formalise the building blocks, that we will utilise throughout the paper. The general space we are working in is a cross-space of perception and (grasping) action. We represent the perception side using 3D surfling features. 3D surfling features describe small surface patches in terms of a pose. In addition, we introduce a granularity measure that depicts the size of the features. Based on the previous description, we formalise 3D surfling features as $\mathbf{\Pi}^{\sigma} = \{SE(3)\}$ (see Fig. 5b). σ depicts the granularity level for the feature. The granularity is measured in the number of sub-features that a 3D surfling feature rely on and hence is a measure of the surface area it covers.

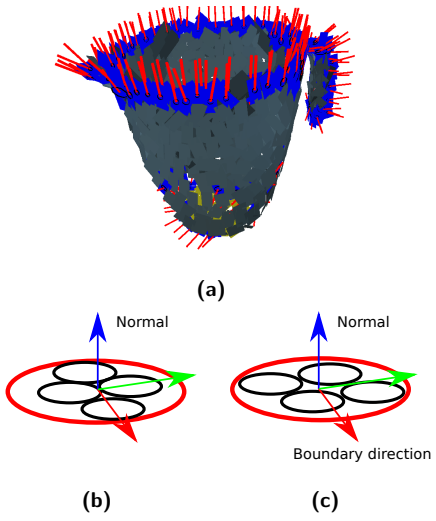


Fig. 5. Visualisation of the two basic building block. (b) a 3D surfling, Π^{σ}, where a principal component analysis is performed on the sub-features (black ones) to decide the orientation. (c) a boundary corrected 3D surfling, $\Pi^{\sigma,\beta}$, where the orientation is decided by the direction of a boundary. In (a), we see both boundary 3D surflings, blue with a red arrow, and standard 3D surflings.

With the description of the basis 3D surfling feature on the perception side, we introduce the concept of feature relations. Feature relations are essentially a combination of multiple features (3D surflings) described through their spatial and/or perceptual relationship, that allows for a set of higher level features.

One motivation for introducing the concept of feature relations is to compensate for the ambiguity in the 3D surfling feature pose, because the pose is derived from a principal component analysis of the underlying sub features (see Figs. 5b and 5c). The result is an unambiguous surface normal, but the other components in the pose are ill defined. Hence we need other means to define a stable orientation of a 3D surfling feature.

By introducing feature relations, we add information through the spatial relationships between features, which theoretically will compensate for the uncertainties in the original pose. Moreover, we gain local structure information when we combine multiple features and hence achieve a more expressive visual representation. By means of feature relations, we create a representation where we can derive robust structures for predicting action affordances despite the simplicity of the basic building blocks. A complementary approach to tackle the issue of pose ambiguity in the basic building block is to introduce a more elaborated or expressive feature by additional levels of semantic. A boundary feature is introduced, where the pose is decided by the direction towards a given boundary. The boundary surfling is described by $\mathbf{\Pi}^{\sigma,\beta} = \{SE(3)\}$, where β denotes it is a boundary surfling and by definition, the first axis of the pose-frame is directed towards the boundary, see Figs. 5a and 5c.

Based on these basic 3D surfling features, we introduce a notation used for feature relations in equation 1,

$$\mathbf{\Upsilon}_N^{\sigma} = f(\mathbf{\Pi}_0^{\sigma}, \mathbf{\Pi}_1^{\sigma}, ..., \mathbf{\Pi}_{N-1}^{\sigma}) \quad (1)$$

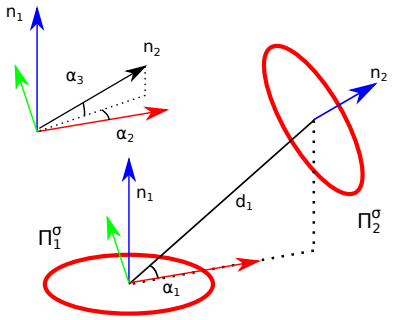


Fig. 6. Example of a feature relations of order two. It should be noted how the angles α_2 and α_3 describe the normal of the second feature Π_2^σ in terms of the coordinate system of the first feature, Π_1^σ.

where N denotes the number of combined features, also referred to as the order of the relation, and σ denotes the granularity of the features it relies on. The function f transfers a combination of features into a parametrisation depending on the order and abstraction. To exemplify the transfer, we will describe a feature relations of second order based on generic 3D surflings (an illustration of such feature relations is shown in Fig. 6) which is parametrised as described in equation 2. The angles α_1 to α_3 and distance d_1 are defined as depicted in Fig. 6, whereas the feature relation coordinate system is described in world coordinates.

$$\boldsymbol{\Upsilon}_2^\sigma = f(\boldsymbol{\Pi}_0^\sigma, \boldsymbol{\Pi}_1^\sigma) = \{SE(3)_W^P, \alpha_1, \alpha_2, \alpha_3, d_1\} \quad (2)$$

3.2 Action representation

Until now, we have not covered the action side of the perception × action space that we want to investigate. For this, we introduce grasping actions as an example. We define a minimalistic grasping action as follows:

$$\text{Action}_{\text{Grasp}} = \{SE(3)_W^A, E\} \quad (3)$$

which essentially describes a target action pose in world coordinates ($SE(3)_W^A$) and an evaluation of the grasp outcome (E). The evaluation can theoretically take any value, but for the grasping case in this paper, we utilise a binary description. Other parameters such as preshape joint angles of the gripper could also be added to get a more elaborated action description.

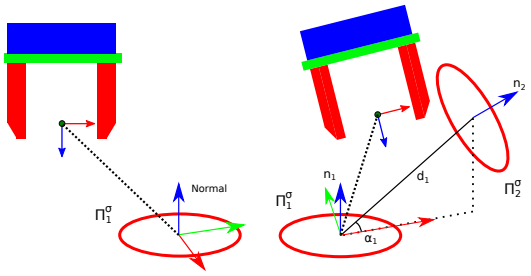


Fig. 7. Illustration of the linkage between action and perception for the first order case (left) and the second order case (right), essentially being a linkage (the dotted line) between the frame of the perception descriptor and the frame of the action.

3.3 Linking perception and action

In the final step, we link the perception part with the action part. Instances of the combined representation will be referred to as *particles* and denoted ρ as depicted in equation 4 and described in a condensed form using ρ's with superscript A (for action) and P (for perception) respectively.

$$\rho_i = \{\rho_i^P \times \rho_i^A\} \quad (4)$$

A linked particle based on the previous examples of perception, equation 2, and action, equation 3, is presented in equations 5 to 6, where $SE(3)_P^A$ is a condensation of the poses from the different domains into a single pose, where the action is described in terms of the coordinate system of the perception side. In Fig. 7, an illustration of a particle is shown for two different levels of perception.

$$\rho = \{SE(3)_W^P, \alpha_1, \alpha_2, \alpha_3, d_1, SE(3)_W^A, E\} \quad (5)$$

$$\rho = \{SE(3)_P^A, \alpha_1, \alpha_2, \alpha_3, d_1, E\} \quad (6)$$

4 Learning algorithm

In this section, the algorithm for learning and applying the visually predicted action affordances will be explained. An overview of the process is shown in Fig. 8. The figure covers the steps from the Object/Action environment through a data-creation process, a learning process of which the results are stored in an Action Perception database, and finally a prediction step where the knowledge is used to predict actions to be performed in the Object/Action environment.

In the following subsections, the different components shown in the overview diagram will be covered. First we describe the data creation process, (section 4.1), next the learning phase will be explained (section 4.2) and finally the utilisation of the learned knowledge for predicting actions will be described in (section 4.3).

4.1 Data creation

The data creation process is relying on the formalism defined in section 3.1, where the two domains, action and perception, are combined. From the Object/Action environment, we acquire evaluated action information as well as visual information in terms of extracted 3D surfling features, for training set objects. From features, we compute feature relations and then link the two domains together such that that the action is defined with respect to the feature combination (see equation 6).

The procedure for doing the linking process is explained in algorithm 1. Note, that for every particle, ρ, a random action and feature relation is chosen and combined into a particle. The random selection is introduced due to the intractability of exhaustively combining feature relations and actions. In the combination step, additional constraints such as, e.g., locality (the action target pose should be close to the feature relation pose) could be added.

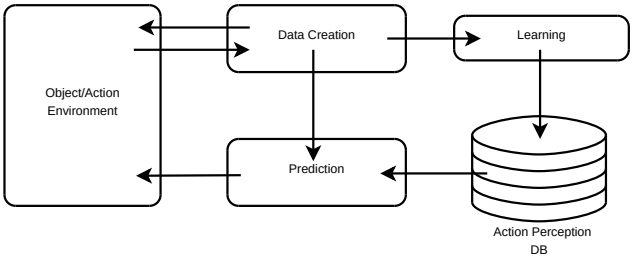


Fig. 8. Overview diagram of the data creation, learning, storing and prediction of action affordances.

ALG. 1: Combining feature relations with actions.

Input: FeatureRelations $\rho^{\mathbf{P}}$, Actions $\rho^{\mathbf{A}}$
Output: Particles, ρ

```
N ; // Number of particles we use
i = 0;
while i < N do
    ρ_j^A = random ρ^A;
    ρ_k^P = random ρ^P;
    ρ_i = {ρ_k^P × ρ_j^A};
    ρ.push_back(ρ_i);
    i++;
```

A fundamental part of the data creation process is the input actions. Such actions could be provided from various sources, e.g., real world experiments, simulation, hand labelled data or through human demonstration. The desirable properties of the input actions are that they provide a reasonable coverage and success rate for a given situation. In this work, we approach the data creation with a simulated environment that allows for a more explorative approach as compared to real world experiments. We utilise visually extracted surfling features as a bias for proposing a input action set. In Fig. 3a, a number of examples are shown of how features can act as a bias for proposing candidate actions for the grasping case. That said, the action candidate creation is likely to be very dependent on the type of action. The input actions are then evaluated in simulation. Hereby we retain some control over the amount of input actions while we also can guide the rate of success.

4.2 Neighbourhood analysis

In this section, the foundation for learning will be described in terms of the different components. First the learning approach is presented, next a two-stage extension is introduced and finally an optimisation of the learning outcome is considered.

4.2.1 Algorithm outline

The overall outline of the learning process is depicted in Fig. 9. This illustration encapsulates the steps from the feature extraction, action creation to the establishment

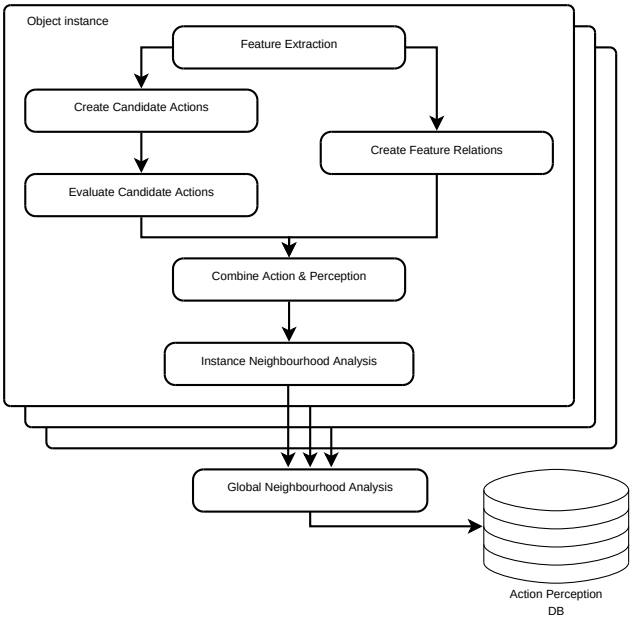


Fig. 9. Overview of the learning process, note the two-stage neighbourhood analysis, initially on instance level and finally on the combined set.

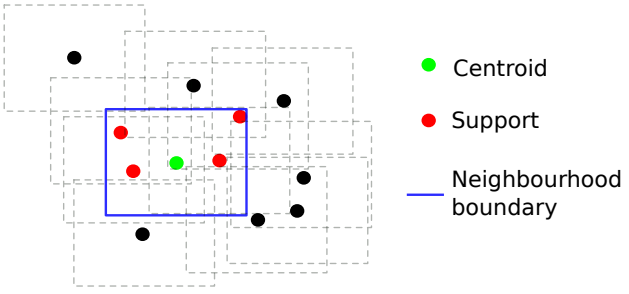


Fig. 10. 2D illustration of the neighbourhood analysis around a particle, highlighted in green.

of an action perception database, in terms of particles ρ.

The core of the learning process is a neighbourhood analysis, which is illustrated in Fig. 10. The first step is to find the set of particles in the neighbourhood, which is formally described by, A_k, in equation 7. Based on the set of particles, the two measures probability and support are computed. The support, s_k, is given as the size of the set inside the neighbourhood (equation 8) and the probability, P_k, is defined as the average success probability within the neighbourhood (equation 9).

As we will show in the result section, both variables are essential for the efficient prediction of affordances.

$$A_k = \{\rho_i | Dist(\rho_i, \rho_k) < \mathbf{t}\} \tag{7}$$

$$s_k = |A_k| \tag{8}$$

$$P_k = \frac{1}{|A_k|} \sum_{\rho_i \in A_k} E_i \tag{9}$$

Given these two measures, we have a description of the action perception space in terms of success-outcome likelihood and the support for this likelihood. The latter can also be seen as the particle density in the neighbourhood. From a formal point of view, we go from particles in the form of equation 5 to *evaluated particles* of the form expressed in equation 10.

$$\rho_i^E = \{\rho_i^P \times \rho_i^A, P_i, s_i\} \tag{10}$$

The elementwise $Dist$ function in equation 7, is used to decide whether the particle, ρ_k, is in the neighbourhood of ρ_i. For the distance computation, we split $SE(3)_P^A$, from equation 6, into a rotational part described by a quaternion $\mathbf{q}$ and a positional part (x, y, z) described by three components:

$$SE(3)_P^A = \{x, y, z, \mathbf{q}\} \tag{11}$$

The distance is computed in the individual dimensions of the parametrisation, with the exception of the orientation part of the $SE(3)_P^A$ pose, which is computed as the shortest angular distance between the orientation of ρ_k and ρ_i. Using a quaternion representation, the computation can be done with the formula in equation 12, where $\langle \mathbf{q_1}, \mathbf{q_2} \rangle$ depicts the inner product of the two quaternions $\mathbf{q_1}$ and $\mathbf{q_2}$.

$$dist(\mathbf{q_1}, \mathbf{q_2}) = 2 \arccos(\langle \mathbf{q_1}, \mathbf{q_2} \rangle) \tag{12}$$

In equation 13, the distance computation is expressed between two particles of the type described in equation 6.

$$\begin{aligned} Dist(\rho_i, \rho_k) = \{&x_i - x_k, y_i - y_k, z_i - z_k, dist(\mathbf{q_i}, \mathbf{q_k}), \\ &\alpha_{1,i} - \alpha_{1,k}, \alpha_{2,i} - \alpha_{2,k}, \alpha_{3,i} - \alpha_{3,k}, d_{1,i} - d_{2,k}\} \end{aligned} \tag{13}$$

It should be noted that the comparison operator ($<$) in equation 7 is an element wise comparison of the distance vector (see equation 13) and the threshold vector ($\mathbf{t}$). For it to be true, all the elementwise comparisons should be true.

The basic process for performing a neighbourhood analysis is captured by algorithm 2. The decisive parameter when doing a neighbourhood analysis is the choice of "neighbourhood" or vicinity, expressed as the

ALG. 2: Neighbourhood analysis.

Input: Particles ρ

Output: ActionPerceptionDB, $\rho_{\mathbf{DB}}$

$\mathbf{t}$ =Compute threshold;

for ρ_k ***in*** ρ **do**

  $A_k = \{\rho_i | Dist(\rho_k, \rho_i) < \mathbf{t}\}, \rho_i \in \rho$;

  $P_k = \frac{1}{|A_k|} \sum_{A_k} E_i$;

  $s_k = |A_k|$;

  $\rho_k^E = \{\rho_k, P_k, s_k\}$;

  $\rho_{\mathbf{DB}}.push\_back(\rho_k^E)$;

threshold vector $\mathbf{t}$ in equation 7. The argument is that a too large neighbourhood will over-smooth the data resulting in no or little gain in information and predictive power. In a similar sense, a too narrow neighbourhood will result in no generalisation at all. In order to have a reasonable basis for choosing the neighbourhood, we propose two options for setting the threshold, $\mathbf{t}$, a manual choice and an automatic choice. Using a manual approach to set the parameters involves setting a fixed threshold of each individual dimension based on common sense and then enable a scaling of the fixed parameter vector $\mathbf{t}$ by a scalar multiplier, M_m (see equation 14).

$$\mathbf{t}_{manual,M} = M_m \mathbf{t}_{manual} \tag{14}$$

The manual parameter setting can make use of the semantics in the feature spaces (e.g., a distance measure for position can be chosen relative to the gripper opening). An alternative to the manual setting is to utilise a rule of thumb from Kernel Density Estimation to find a suitable threshold. Scott [22] proposed such a rule (see equation 15). The estimated threshold or bandwidth, t_{scott} is depending on the number of instances in the data, n, the dimensionality of the space, d, and the estimated standard deviation of the data-points within the dataset, $\hat{\sigma}$. It should be noted that the dimension of the vector $\mathbf{t}$ and $\hat{\sigma}$ depend on the parametrisation used for the particles ρ.

$$\mathbf{t}_{scott} = n^{\frac{-1}{d+4}} \hat{\sigma} \tag{15}$$

We can then use Scott's rule as a guideline for the ratio between the distances in the different dimensions. To adjust the neighbourhood-distance, we introduce an additional scaling parameter, M_s, similar to the multiplier mentioned for the manual defined threshold.

$$\mathbf{t}_{scott,M} = M_s \mathbf{t}_{scott} \tag{16}$$

The potential risk of using Scott's rule for bandwidth computation is that it does not take the semantic of the parameters into account. Given the data has the property of having a large variance but very narrow discriminative areas, an automatic threshold will result in suboptimal interpretation of potential good areas as it will work as an smoothing operator on the data.

In the Appendix, a comparison of an automatic- versus a manually set threshold is carried out. Here it it becomes apparent, that there might be a gain in prediction performance by choosing an appropriate manual threshold. Although there is a little gain, it is unlikely that the effort is worth it, especially when considering even more advanced visual representations of higher dimension.

4.2.2 Two-stage neighbourhood analysis

As displayed in the overview diagram (see Fig. 9), the neighbourhood analysis is performed in a two-stage process. This is motivated by the urge to decrease the computation time. The cost for performing the neighbourhood analysis is related to the number of particles n, due to reliance on the KD-tree data structure. The computation cost for performing a search is $O(\log n)$, and when we take into account that we need to perform a search for every particle, the computational cost adds up to $O(n \cdot \log n)$. We can reduce the computational complexity by decreasing the amount of particles on which we are performing the neighbourhood analysis.

In an initial stage, we perform a neighbourhood analysis on the particles from the individual objects in the full dataset. By splitting in terms of object instances rather than doing a random split of the full dataset, we ensure that the smaller problems covers the same areas of the action perception space and hence allow for generalisation. The partitions provides us with a set of significantly smaller neighbourhood problems, instead of a single large problem. Having a set of smaller problems, that are independent, we also facilitate a parallelisation of the first stage. The second stage in the analysis (global neighbourhood analysis), is a neighbourhood analysis performed on the outcome of the set of smaller first stage problems. In order for the two-stage approach to have an effect, the first stage should work as a filter, such that only "promising" particle candidates are taken into account.

One way of filtering away "un-promising" particles, is to set up a criteria for the minimum support that a particle should have for it to be taken into account.

Such a filter could be expressed in absolute, average or median values of the support in the dataset. There are however some pitfalls when using support as a filtering parameter, namely the risk for filtering away the diversity in the particles. This aspect of the learning is addressed in the results (section 6.4), where different levels of support filtering has been applied to verify the effect on the prediction outcome.

In practice, an introduction of support filtering in the neighbourhood analysis includes a small extension that removes particles below a certain support threshold for the final dataset.

4.3 Prediction

In order to apply the learned data in novel situations, two different methods have been applied. One method where we look for similarities on the perception side and use these as direct cues for proposing actions denoted as "direct action proposition" and secondly a method, denoted as "voting scheme", where we suggest a candidate list of actions from the *ActionPerceptionDB* to vote for the actions. The two approaches will be explained in the following subsections.

4.3.1 Direct action propositions

The direct action proposition approach is based on the assumptions, that our learned high probability and high support action perception particles are descriptive enough for predicting actions. In Fig. 11, an overview of the involved steps is shown. We extract feature relations, the ρ^P part of the particles, from the novel object and search for similar ρ^P parts in the *ActionPerceptionDB*. If we find a similar perception part with a high probability for success and high level of support, we take its action part, ρ^A, and attach to our ρ^P part. This means, if we find an action described in terms of the perception part from the novel object, we have a proposed action.

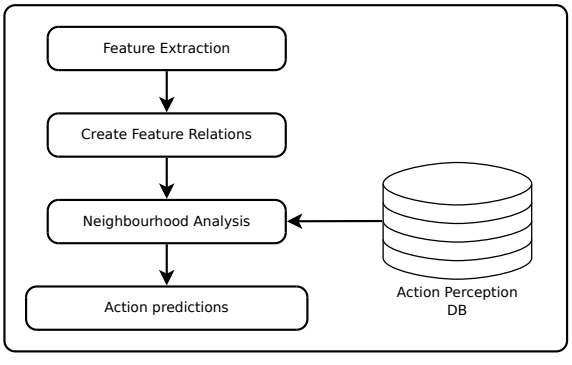


Fig. 11. Overview diagram of the steps involved in the direction action proposition method.

Given the simplicity of the direct action proposition approach, it has some limitations. The main problem is, that the approach relies heavily on a discriminative perceptual representation in order to make reliable predictions. The potential problem arises when we use a too simple perceptual representation, namely that a particular simple relation can predict very different actions depending on the object it was learned from. This problem should eventually disappear if we utilise a more descriptive perception representation. Therefore we introduce a second approach, the voting scheme. For comparison, experiments have been carried out with the direct action proposition method (see Appendix), where the prediction performance and limitation in the method are presented.

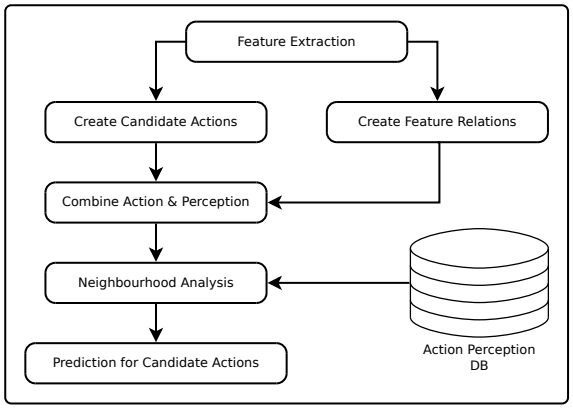


Fig. 12. Overview diagram of the voting scheme.

4.3.2 Voting scheme

The principle behind the voting scheme is that we want to utilise our learned *ActionPerceptionDB* as a means to vote for a set of candidate actions. Hereby we utilise multiple perception descriptors to predict the action outcome of a single candidate action, and by that improve the robustness of the prediction. In Fig. 12, an overview of the process involved in the voting scheme is

shown. Note that the candidate action creation is identical to the one described in section 4.1.

The voting procedure has been formalised in algorithm 3. The process is very similar to the actual learning phase, however where we in the learning phase "forget" the origin actions when we combine them with the perception part, ρ_P, we remember them in the voting scheme. This allows for a final step in which we can project a prediction probability back to the origin candidate action, and thereby give a prediction based on multiple perception action particles. In Fig. 13, an example is presented, where we utilise multiple feature relations (Figs. 13d to 13g), to vote for a single candidate action (Fig. 13h).

ALG. 3: Voting scheme.

Input: ActionPerceptionDB $\rho_{\mathbf{DB}}$, Features
Output: Candidate Actions with prediction $\rho^{\mathbf{A}}_{\mathbf{C},\mathbf{E}}$

$\rho^{\mathbf{A}}_{\mathbf{C}}$ = Create Candidate action through visual bias;
$\rho^{\mathbf{P}}_{\mathbf{C}}$ = Compute feature relations;
$\rho_{\mathbf{C}}$= Combine feature relations with candidate actions as in ALG. 1;
for $\rho_{C,k}$ ***in*** $\rho_{\mathbf{C}}$ **do**
  $A_k = \{\rho_{C,i} | Dist(\rho_{C,i}, \rho_k) < \mathbf{t}\}, \rho_{C,i} \in \rho_{\mathbf{DB}}$;
  $P_k = \frac{1}{|A_k|} \sum_{A_k} E_i$;
  $s_k = |A_k|$;
  $\rho^E_{C,k} = \{\rho_{C,k}, P_k, s_k\} = \{\rho^P_{C,k}, \rho^A_{C,k}, P_k, s_k\}$;
// Backproject probabilities to origin actions
for $\rho^A_{C,l}$ ***in*** $\rho^{\mathbf{A}}_{\mathbf{C}}$ **do**
  $B_o = \{\rho^{A,E}_{C,i} | \rho^{A,E}_{C,i} == \rho^A_{C,l}\}, \rho^A_{C,i} \in \rho^{\mathbf{A}}_{\mathbf{C}}$;
  $P_{avg} = \frac{1}{|B_o|} \sum_{B_o} P_i$;
  $\rho^A_{C,l} = \{\rho^A_{C,l}, P_{avg}\}$;
  $\rho^{\mathbf{A}}_{\mathbf{C},\mathbf{E}}.push\_back(\rho^A_{C,l})$

5 Setting

In this section, the settings for the experimental work will be explained. It involves the object data set (section 5.1), the simulation environment (section 5.2), the feature extraction (section 5.3), the visual biased action sampling (section 5.4) and details regarding action and perception parametrisation (section 5.5).

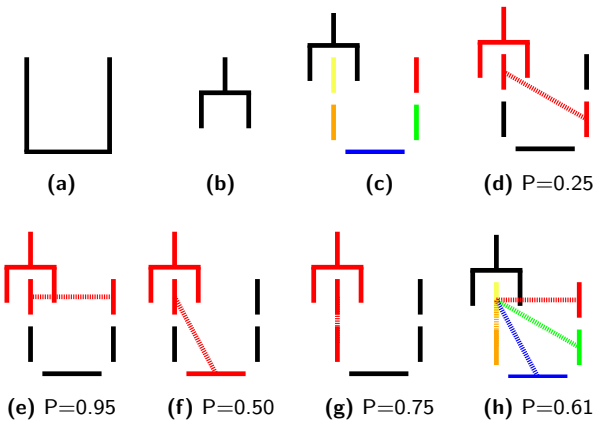


Fig. 13. A 2D example illustration of the voting scheme. (a) 2D container, (b) a two-finger gripper, (c) a feature representation with a candidate grasp. Figures (d), (e), (f) and (g) show feature relations that are used to vote for the candidate action. Probabilities are shown below which would be the probabilities found in the database. Given the example probabilities, the combined probability for the candidate grasp is shown in (h).

5.1 Object set

In Fig. 14 an overview of the different objects used in the experiments is given. The objects are split into three different categories, namely box-like objects, curved/cylindrical objects and open/container objects. The objects in the set are partly taken from the KIT object database [23] and partly from the online database archive3D [24]. The KIT objects are digitalised real objects which potentially simplifies the transfer from a simulated environment to the real world. Furthermore they add realism to the feature extraction as the objects are textured based on the real objects. However due to the lack of open/container objects in the KIT set, we needed to extend the object set with objects from other sources, which are not digitalised real objects.

5.2 Simulation environment

The experiments in this paper are all performed in a simulated environment utilising the robotic library RobWork [25]. RobWork is used to create a realistic environment, that facilitates simulated sensors (such as RGB-D sensors and Stereo cameras) as well as a dynamics simulator [26]. Fig. 15 shows a view of a dynamic grasp simulation with the Schunk SDH-2 hand and a pitcher from the visualisation tool. The grasping simulations are performed in a free-floating world where gravity is not taken into account since it facilitates grasping from every direction.

Fig. 14. Visualisation of the three different categories of objects. (Top), box objects, (middle), round objects and (bottom) open objects.

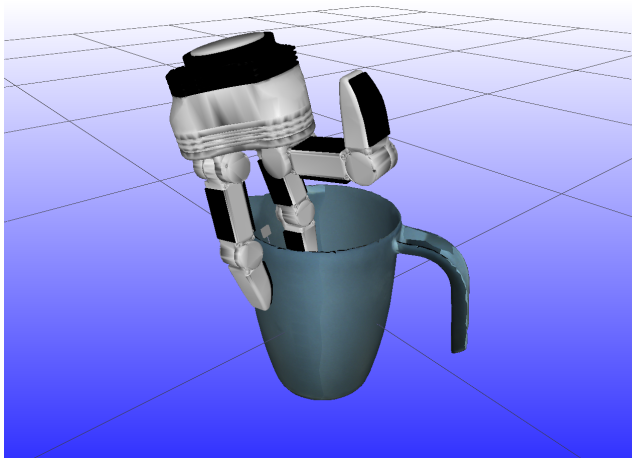


Fig. 15. Visualisation from RobWork showing a grasping action with the Schunk SDH-2 hand.

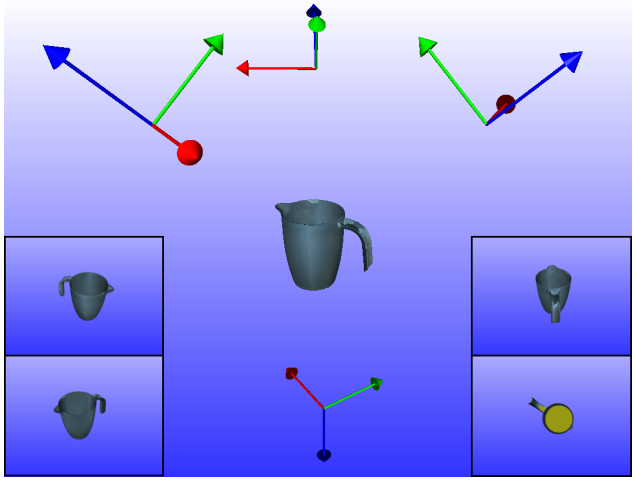


Fig. 16. Visualisation of the four simulated RGB-D sensor views, illustrated with the four coloured frames, and the object of interest in the centre. The frames depict the position and the camera-view are along the negative z-axis, coloured blue. The views from the four cameras are shown in the small images.

5.3 Feature extraction

An essential part of the setting is the feature extraction from the simulated environment. In Fig. 16, our setup of RGB-D sensors is displayed. Having a setup of three sensors surrounding the object and an additional sensor from below gives an approximated full view of the objects in the centre.

Based on the simulated setup in RobWork, we are able to extract the 3D surfling features at different granularities and with added semantic. An example of the feature extraction of surflings at four different granularity levels is visualised in Fig. 17. Furthermore the extracted features are shown both with and without the added semantic for boundary features. The boundary features are shown with an additional vector depicting the direction of the boundary.

5.4 Action sampling

The action sampling biased through the visually extracted features is a prerequisite for learning the grasp affordances in an automatic way since it ensures a reasonable chance of success as well as a limit to the amount of considered actions. We propose two template grasp types for the sampling. The two types are visualised in Fig. 18, one is denoted the SidePinchGrasp and the

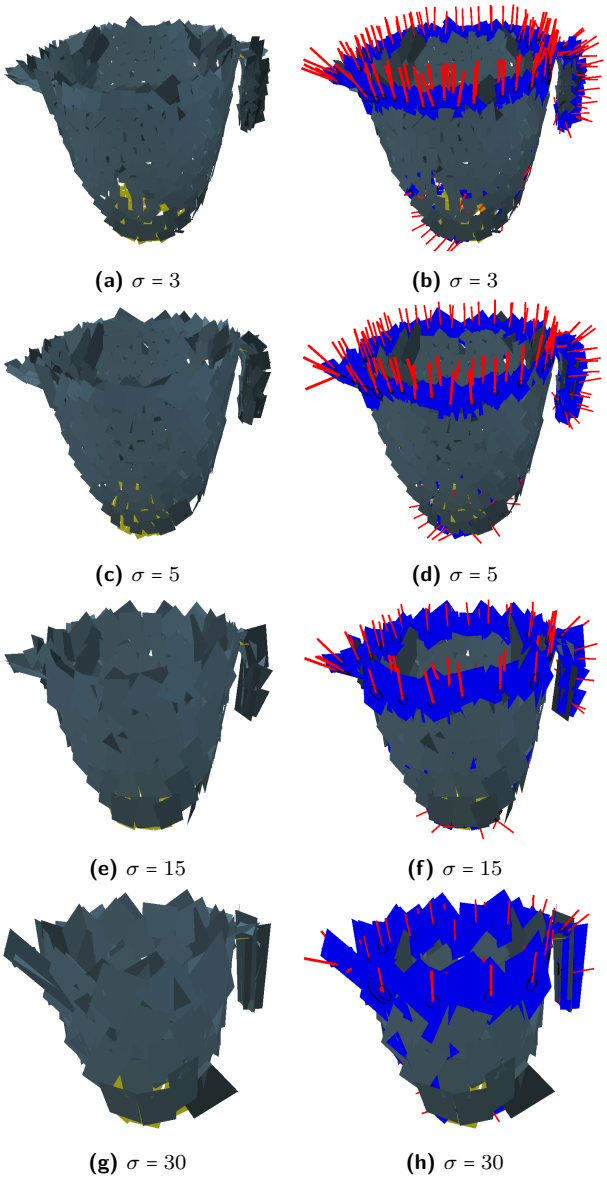


Fig. 17. Visualisation of extracted features at four different granularities with (right column) and without (left column) boundary semantic.

other is denoted TopGrasp. The SidePinchGrasp has a rather narrow opening between the two fingers such that it can grasp within a container and the TopGrasp have wide open fingers to make an encompassing grasp of larger objects. We create a set of candidate grasps by means of extracted 3D surfling features with a small feature size such that we can achieve a reasonable coverage of the objects. Based on the features, we propose a set of template grasps by rotating them in 32 steps around the feature normal. From this sampling we achieve an average success-rate between 10% and 50% depending on the object set (see the random chance as dashed horizontal lines in the results plots Figs. 22, 23 and 24).

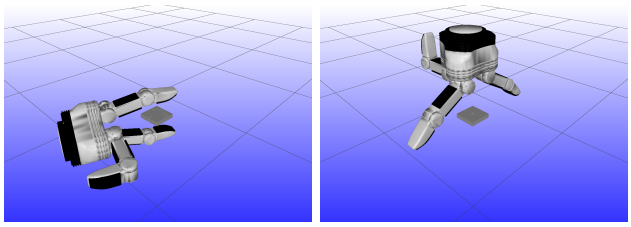


Fig. 18. Visualisation the two different basic grasp types, SidePinchGrasp (left) and TopGrasp (right)

5.5 Parametrisation of feature relations

Throughout the experiments, we will rely on a limited set of different feature relation types, namely of first and second order relation with different levels of boundary semantics. In equations 17 to 22 the different parametrisations are presented.

$$\Upsilon_1^{\sigma} = f(\Pi^{\sigma}) = \{SE(3)\} \tag{17}$$

$$\Upsilon_1^{\sigma,\hat{\beta}} = f(\Pi_1^{\sigma,\hat{\beta}}) = \{SE(3)\} \tag{18}$$

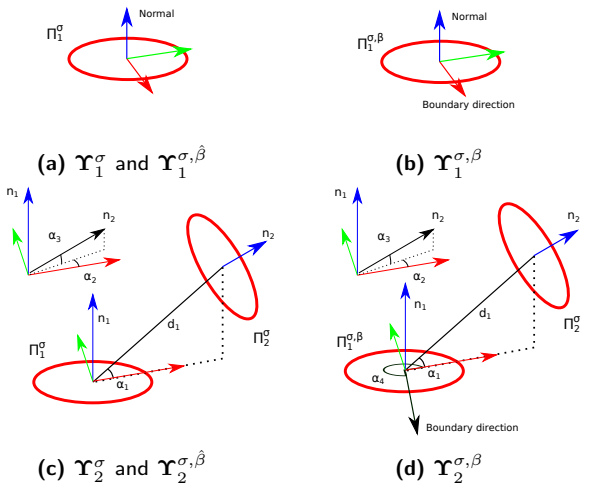
$$\Upsilon_1^{\sigma,\beta} = f(\Pi_1^{\sigma,\beta}) = \{SE(3)\} \tag{19}$$

$$\Upsilon_2^{\sigma} = f(\Pi_1^{\sigma}, \Pi_2^{\sigma}) = \{SE(3), \alpha_1, \alpha_2, \alpha_3, d_1\} \tag{20}$$

$$\Upsilon_2^{\sigma,\hat{\beta}} = f(\Pi_1^{\sigma,\hat{\beta}}, \Pi_2^{\sigma}) = \{SE(3), \alpha_1, \alpha_2, \alpha_3, d_1\} \tag{21}$$

$$\Upsilon_2^{\sigma,\beta} = f(\Pi_1^{\sigma,\beta}, \Pi_2^{\sigma}) = \{SE(3), \alpha_1, \alpha_2, \alpha_3, \alpha_4, d_1\} \tag{22}$$

In Fig. 19 visualisations are shown of the different types of feature relations used in the experiments. Note that only four different feature relations are visualised. The reason is that the parameters for equations 17 and 18 are similar with the only difference being that we know the feature in equation 18 is a boundary feature. The same holds for the two cases in equation 20 and 21. The parametrisation covers three first order cases: one plain feature (Υ_1^{σ}), one where we know the feature is a boundary feature ($\Upsilon_1^{\sigma,\hat{\beta}}$) and one were we utilise the boundary semantic with direction ($\Upsilon_1^{\sigma,\beta}$). As for first order, we introduce a parametrisation for three second order cases: One without semantic (Υ_2^{σ}), one with the knowledge of a boundary but not the direction ($\Upsilon_2^{\sigma,\hat{\beta}}$) and finally one with boundary semantic and direction ($\Upsilon_2^{\sigma,\beta}$).



(a) Υ_1^{σ} and $\Upsilon_1^{\sigma,\hat{\beta}}$ (b) $\Upsilon_1^{\sigma,\beta}$

(c) Υ_2^{σ} and $\Upsilon_2^{\sigma,\hat{\beta}}$ (d) $\Upsilon_2^{\sigma,\beta}$

Fig. 19. Visualisation of the utilised feature relations and the associated parameters.

6 Results

The result section is divided into four subsections. In section 6.1, we will present the outcome of the learning phase in terms of associated support and probability of the evaluated particles. In section 6.2, we will present the core results comparing the prediction performance when features at different granularities, different levels of abstraction and different semantics are input to the voting scheme. Subsequently (section 6.3), a qualitative analysis is presented of the results. Finally (section 6.4), we will present results regarding the impact of support filtering. In the experimental work, the different object sets have been split into two classes such that the learning from the first class and is applied on the second and vice versa.

6.1 Learning outcome

In order to examine the learning outcome before it is used for prediction, we visualise the frequency of occurrence of the evaluated particles (see equation 6) in terms of support and probability. Fig. 20 shows the distributions in 2D histogram for the different parametrisations described in equations 17 to 22, where the colour depicts the frequency. The colouring is based on the log_{10} transform of the actual frequency in the area to allow for a visible distinction. A histogram corresponding to Fig. 20a but without performing a log_{10} transformation of the frequency is shown in Fig. 21 as a comparison. In this plot, we only see that the majority of the particles have low support and probability.

When assessing the 2D histograms in Fig. 20, we can acquire indications about the predictive power of the different visual representations. We see a shift towards the higher probability areas when the order is raised or semantic is added to the feature relation, e.g., compare Fig. 20a towards Fig. 20f. This change is reflected in the later presented prediction results (see Fig. 24).

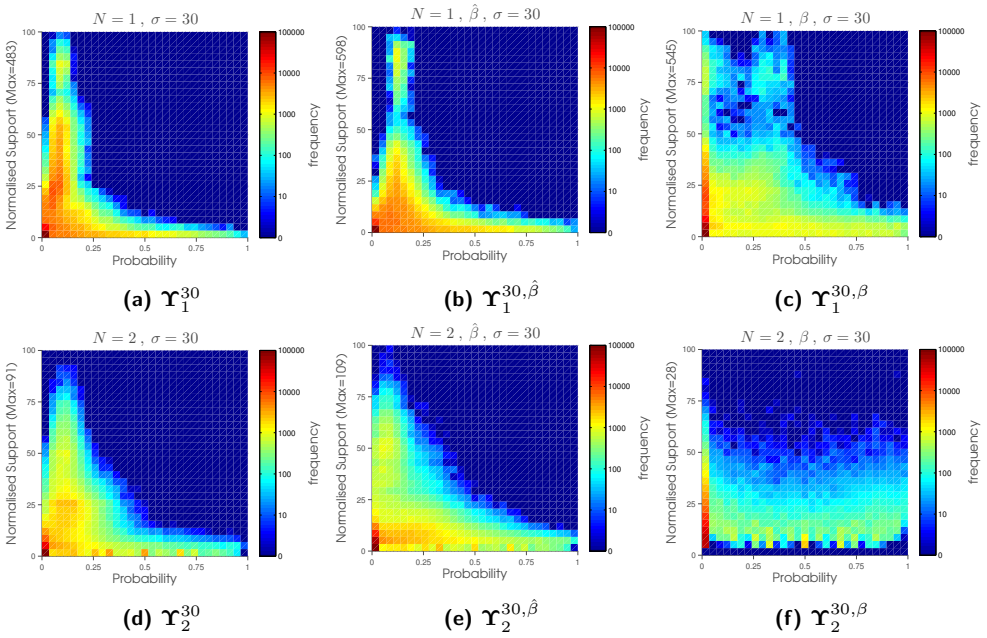
6.2 Core experiments

The outcome of the voting method (section 4.3.2) is a set of candidate actions with associated predicted probability. To discretise these outcomes, which allows for a comparison to the binary grasp outcome from simulation and hence to quantify the performance, we introduce a probability selection threshold. We vary the actual value of the threshold between the extremes. This results in the plots in Figs. 22–24.

In order to assess the prediction results, we present two different average measures of the prediction success over the object set.

- **Avg-1** - An average computed over all the objects in the set, independent of whether feature combinations leading to any grasp prediction were found for a certain object. If no predictions was found the object contribute to the average with a success rate of zero. This average type is plotted with a full line.
- **Avg-2** - An average computed over the average success prediction for only the set of the object instances, where a prediction was found. This average type is plotted with a dashed line.
- **random** - The average chance on the object set for randomly getting a successful outcome given the candidate actions. This measure is plotted with a dashed black line.

When assessing the result plots, there are multiple aspects that one need to consider when we want to identify a good result. One aspect is the difference between the random chance and the top point of the predictions, another is how well a change in the moving threshold to a higher value is reflected as a higher rate of success prediction. Finally one should note the difference between the dotted lines and the full lines as it can be seen as a measure of how well the object set is covered, because the first average will get lower the more objects no grasp affordances can be found for.



(a) Υ_1^{30} (b) $\Upsilon_1^{30,\hat{\beta}}$ (c) $\Upsilon_1^{30,\beta}$

(d) Υ_2^{30} (e) $\Upsilon_2^{30,\hat{\beta}}$ (f) $\Upsilon_2^{30,\beta}$

Fig. 20. Visualisation of the particle distribution for the open object set in terms of support and probability for the learned *ActionPerceptionDB*. The number of particles in the databases ranges from $\sim 250,000$ to $\sim 400,000$.

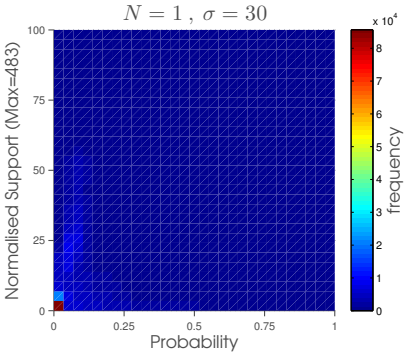


Fig. 21. Visualisation of the particle distribution in terms of support and probability for a learned *ActionPerceptionDB*, where the particle frequency is shown without any modifications.

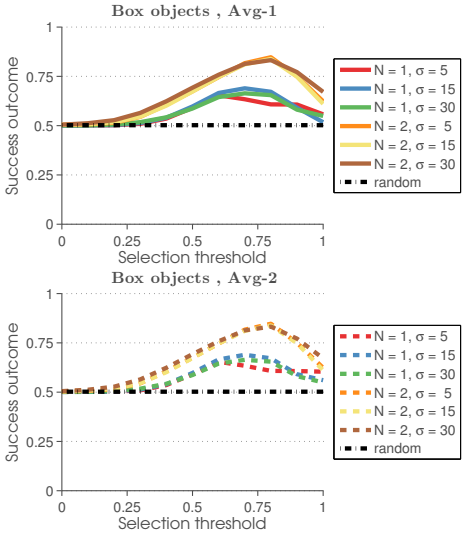


Fig. 22. Box objects prediction results. See equations 17 and20 for the utilised parametrisation and see text for further details.

Box objects: The results for the box object set are presented in Fig. 22. The plots show results where the two dimensions "order" (denoted N, equation 1) and "feature granularity" (denoted σ equation 1), were varied. From the results we derive: (1) When the order is increased, we see a clear improvement of the prediction rates and (2), when the feature size is changed, small changes in the performance are observed. For the first order case, we see the best performance with a medium sized feature whereas there is no or little difference when we compare the second order cases at different granularities.

Round objects: The experimental results acquired for the round object set are shown in Fig. 23. As above, the plots show results where the two dimensions "order" and "feature granularity" where varied. We see: (1) When the order is increased a clear improvement is seen in the predictions and (2), when feature size is varied, we see small changes in the performance for the first order case, whereas we see a clear drop in performance when we use the largest feature size for the second order case. The

last result is in line with the expected result, namely that a large surfling patch is a bad reflection of a round object and hence should be less descriptive as compared to a feature of smaller size.

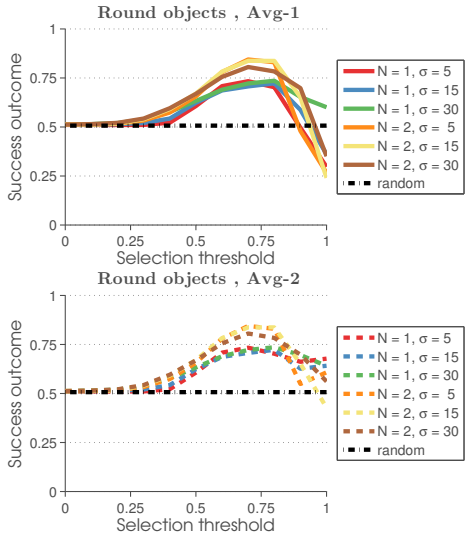


Fig. 23. Round objects prediction results. See equations 17 and20 for the used parametrisation, and see text for further details.

Open objects: The experimental results for the open object set are displayed slightly differently compared to the round and box object sets, since we observed that for open objects the semantic information in terms of boundary information is crucial. The introduction of boundary features allows for all the parametrisations described in section 5.5. The results are presented in Fig. 24 for three different granularities, respectively 5, 15 and 30. In each of the figures, results for the order and level of abstraction through semantic are shown. We see, that the higher order we use and the more semantic we add, the prediction results improve. A significant improvement is observed when we go to second order relations as compared to first order, however we do not see a significant improvement in the prediction power when we add the semantic of a boundary without direction, although we have a better object set coverage as the full line is resulting in a higher success probability. A significant improvement of success prediction rating is achieved for second order relations with boundary and direction. We see however a small drop when we reach the higher end of the selection filter. This can be explained with the fact that the voting method act as a smoothing operator hence high prediction areas will be in general occurring rarely. When we compare the results acquired for the different granularities, we see a similar outcome as in Fig. 22 and 23.

6.3 Qualitative analysis of the power of semantic information

In order to illustrate the performance gain we get when we introduce the boundary semantic, we present a visualisation of the *ActionPerceptionDB* for the three first order cases. The visualisations are shown in Fig. 25. In the centre, a surfling feature is placed and the coloured area around the feature represents how the actions are distributed with respect to the pose of the feature. The colour coding of the actions depicts the likelihood of success for that particular particle.

For $\boldsymbol{\Upsilon}_1^5$ we see a uniform distribution of success probability, whereas for $\boldsymbol{\Upsilon}_1^{5,\hat{\beta}}$ we see two rather uniformly coloured areas. Noticeable is an inner part with a higher success likelihood as compared to the outer part. This is explained with the added knowledge of the boundary, specifically by the fact that, at the boundary, a successful action will be closer to the feature, hence the inner circle captures both the successful boundary grasp as well as unsuccessful, whereas the outer part mostly capture the non-boundary action.

When assessing $\boldsymbol{\Upsilon}_1^{5,\beta}$, it becomes obvious what we gain by introducing the direction towards the boundary. The visualisation shows a high likelihood of success along the direction of the boundary and the further the grasp are located orientational wise from the boundary direction a lower success likelihood is observed.

To visualise how the power of semantic constitute itself when applied for predicting actions, a visualisation of the distribution of predicted grasps for an object is shown in Fig. 26. The figure shows the prediction result for a pitcher, where the order and level of semantic are varied. One can easily notice how the introduction of boundary and direction information for both first and second order cases allow for high success areas at the boundary of the pitcher.

6.4 Support filtering

In order to investigate the impact of the support filter, a series of experiments based on the open object set have been performed, in which the amount of particles used from the first stage of the neighbourhood analysis is varied. We filter by choosing the 0th to the 10th decile of the particles based on their support, e.g., split the first

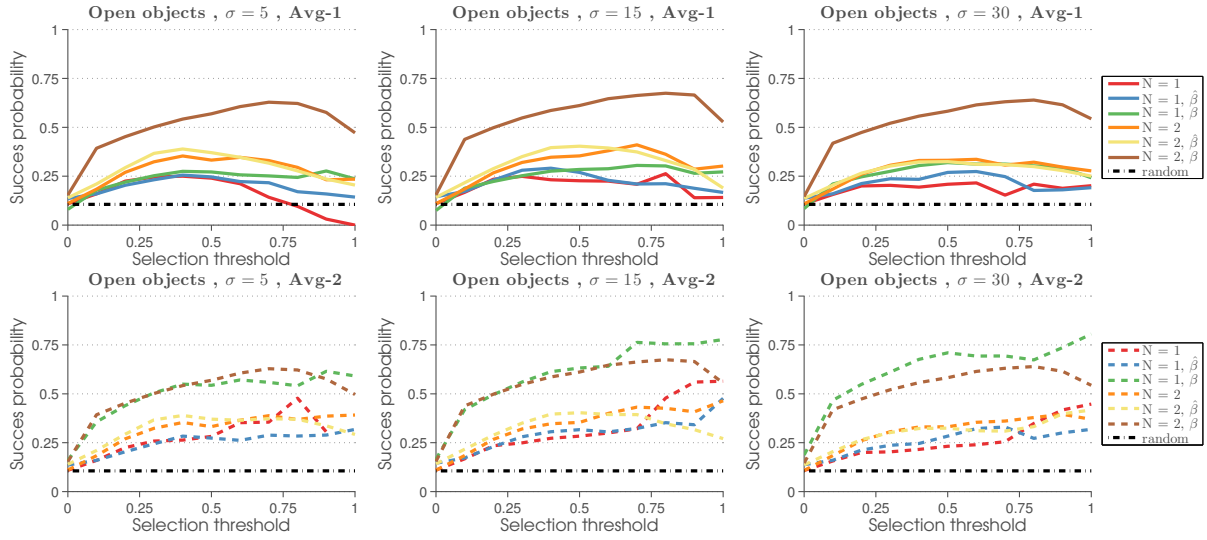


Fig. 24. Prediction result for open objects of granularity 5, 15 and 30. See equations 17–22 for the used parametrisation, and see text for further details.

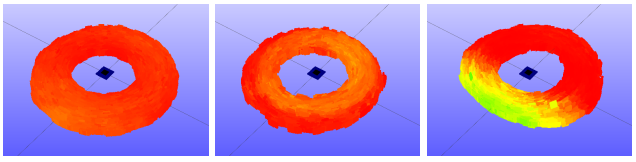


Fig. 25. The three visualisations show how the learned particles are distributed, when the feature part of the particles is positioned in the centre. The three cases are Υ_1^5 (left), $\Upsilon_1^{5,\hat{\beta}}$ (middle) and $\Upsilon_1^{5,\beta}$ (right). Red colour depict a success likelihood of 0.0 and green a success likelihood of 1.0.

decile lowest supported particles from the highest supported particles and then utilise the highest supported part. Hereby we cover the extreme situations, from using every particle to using very few. The acquired results are presented in Figs. 28 and 27. Note the support level is described as a measure between zero and 1.0.

From the results, three main points are derived:

(1) When assessing the results for **Avg-1** for the four cases, Υ_1, $\Upsilon_1^{\hat{\beta}}$, Υ_2 and $\Upsilon_2^{\hat{\beta}}$, the observed pattern shows, that a lower support filter results in higher success rate, although only at lower selection threshold. When comparing the results of **Avg-1** with **Avg-2** for the same four cases, it is noticed that a larger support level result in a higher success rate for the instances that are found. This is in particular seen for Υ_1 and Υ_2, as the selection threshold increases towards 1.0. This result indicates, that with a higher support level very good prediction for a subset of the objects can be derived.

(2) When assessing the Υ_1^{β} results the pattern is significantly different. For **Avg-1** the prediction results show similar performance independent of the applied support level, with the only exception being the highest support level, where the performance is degrading at a low selection threshold. The results for **Avg-2** show that if a prediction is found, then a higher success rate is achieved when a high support level is used.

(3) When assessing the results for Υ_2^{β} the recognised pattern for both the averages, **Avg-1** and **Avg-2**, show similar performance with a small advantage at the higher support levels. Especially at the two highest support levels, an improved performance is noticed.

To summarise the outcome of the support filter experiment, it can be observed that for the less elaborated feature representations, good predictions can be found for individual instances of objects at a high support level, whereas generalisation is in general not observed when utilising a lot of instances (a low support level). For the more elaborated visual representations, it becomes evident, that we are able to achieve an improved performance and still retain the generalisation when using a higher support level. This result indicate, that there indeed exists particular feature relations, which are predictive for grasping in the provided visual representation.

7 Summary and conclusion

In this paper, we have introduced a method for finding combinations of visual features that are predictive for actions. The method has been exemplified for the prob-

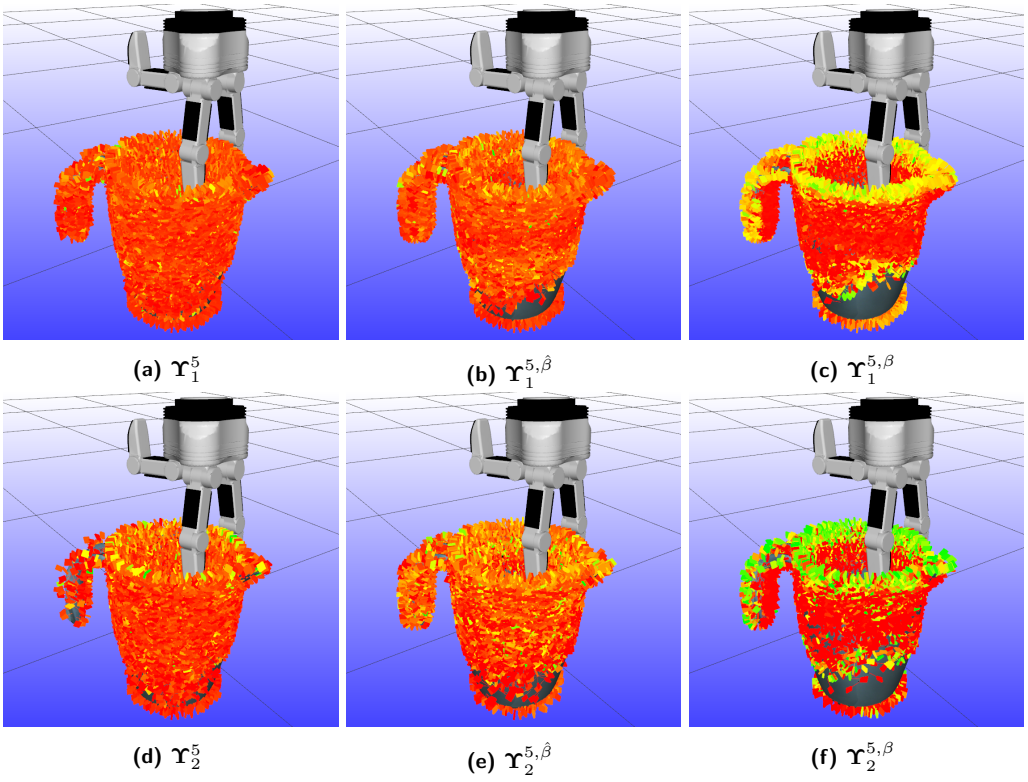


Fig. 26. Visualisation of the grasp predictions for a pitcher object with feature relations of different order and with different semantic. The colour depict the predicted likelihood for success. Green meaning a success likelihood of 1.0 and red meaning a success likelihood of 0.0.

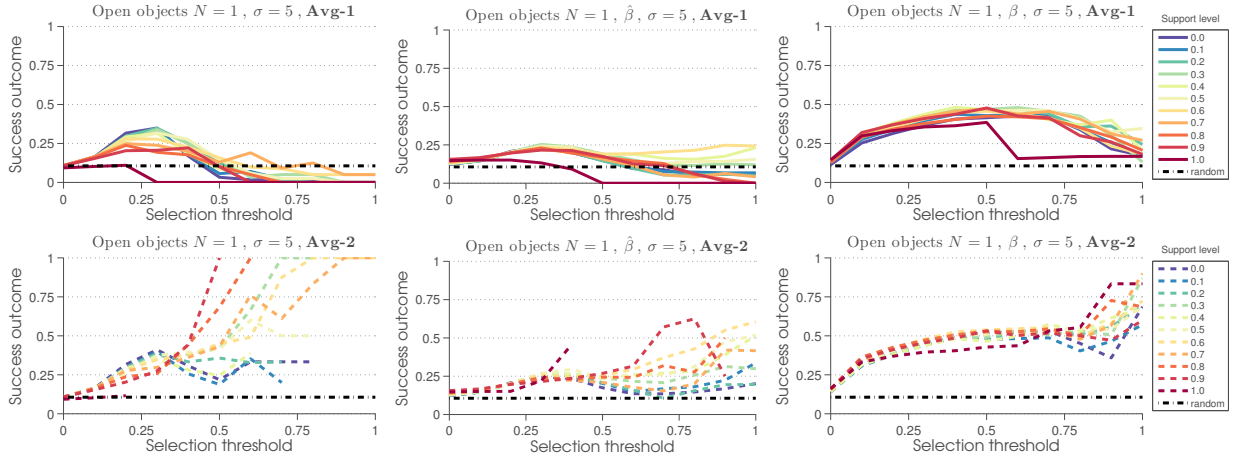


Fig. 27. Prediction results for the open object set, with a feature size of 5 and different support filters, see equations 17–22 for the used parametrisations, and see text for further details.

lem of learning grasping actions. We have performed an analysis of the cross space of perceptual features and grasping actions with special focus on how an enrichment of the perception side leads to improvements of the derived prediction.

Through the performed investigations, we have been able to learn actions with a high likelihood of success for three different object classes, namely box like, round and open objects. For the box and round object set we were able to reach a grasp prediction success of up to 0.90 and 0.80 respectively, when utilising a second order feature constellation as a perceptual descriptor. This high success rate should be seen in the context that grasping of those objects is a rather simple task. For the more

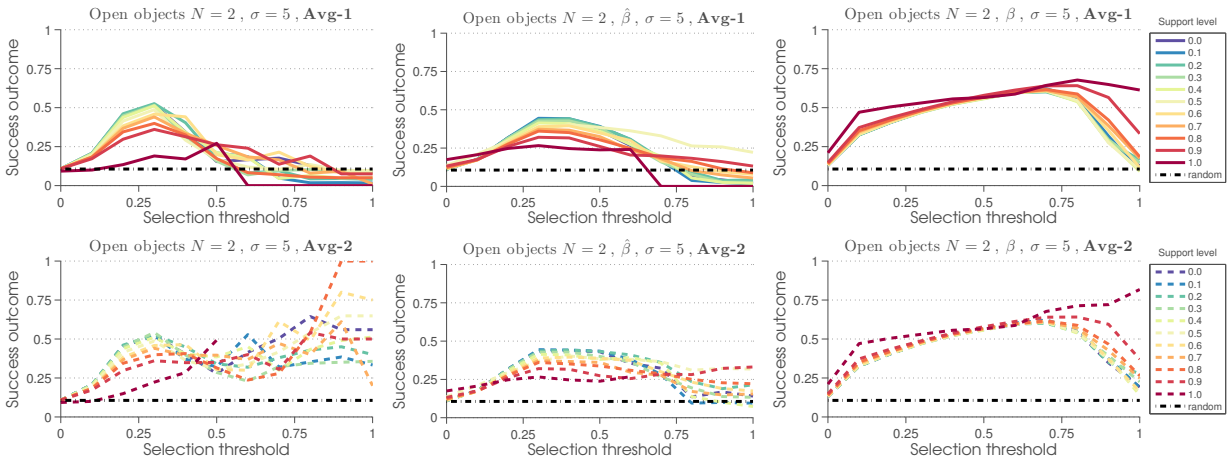


Fig. 28. Prediction results for the open object set, with a feature size of 5 and different support filters, see equations 17–22 for the used parametrisations, and see text for further details.

difficult open object set, we investigated in addition to granularity and order of feature combination also the impact of additional semantic information attached to the features through boundary information. From these results, we were able to achieve a success-rate of up to 0.75, when second order features with added semantic where utilised on the perception side.

By that we have replaced manual design if affordances as done in [4] by learning. We could confirm that relatively high success rates for action feature associations built by means of rather basic features is possible. Moreover and most importantly, we showed how the structure of the feature space influences the results of the algorithm. For that we investigated three important dimensions of a feature space motivated by the visual hierarchy of the human visual system: granularity, order of features and semantic abstraction. Since our approach is not restricted to grasping, in future work we plan to apply our algorithm to other action affordances

A Learning methodology experiments

In the following subsections, two aspects of the learning approach will be investigated. (1) The prediction results when the direct action proposition approach (see section 4.3.1) is applied, and (2) the difference between an automatically- and a manually set threshold (see section 4.2.1).

A.1 Direct action proposition approach

As a comparison to the voting scheme (see section 4.3.2), a number of experiments were performed using the direct action proposition method. The experimental results are presented in table 1. Compared to the results presented when utilising the voting method (see section 6.1), these results are evaluated with a single measure depicting the success prediction. In the experiments, the order and granularity were varied for the box- and round object classes, whereas the level of semantic in addition were varied for the open object class.

For the box- and round objects, two things are observed, (1) A larger feature size improve the success rate for the first order cases, whereas it degrades for the second order cases and (2), the success rate is, in general, higher for the second order cases. The improvement, due to a larger feature, is explained by the increased object knowledge that it brings. This information gain however seem to counteract the added knowledge of two combined features, resulting in a degrade in prediction performance, when a larger feature is used in second order combination.

For the open objects, three things are observed. (1) The performance when utilising the representations without semantic is very low, however an improvement is noticed when going from 1st order cases to second order cases. (2) For the first order cases, a larger feature results in a better prediction rate. This is not the case for the second order cases, where the highest prediction rate is achieved at a feature size of 15. (3) The highest overall prediction rate is achieved at a representation based on Υ_1^{30}. This essentially tell us, that the information gain from a larger feature is superior to adding

FeatureSize	ObjectSet	Order + Abstraction					
σ		N=1	N=1, $\hat{\beta}$	N=1, β	N=2	N=2, $\hat{\beta}$	N=2, β
5	Box objects	0.51	-	-	0.65	-	-
	Round objects	0.57	-	-	0.66	-	-
	Open objects	0.05	0.10	0.44	0.12	0.14	0.45
15	Box objects	0.54	-	-	0.61	-	-
	Round objects	0.62	-	-	0.63	-	-
	Open objects	0.06	0.08	0.52	0.12	0.18	0.49
30	Box objects	0.54	-	-	0.60	-	-
	Round objects	0.67	-	-	0.56	-	-
	Open objects	0.08	0.08	0.53	0.11	0.13	0.45

Table 1. Prediction results when utilising the direct action proposition method. The used parametrisations are found in equations 17–22, and see text for further details.

another feature when used in connection with the direct action proposition method.

Finally, when comparing the results with the voting method results, the direct action approach does not show similar performance, which is explained by the direct attachment of an action to a perceptual representation. This compares to the multiple particles, that are used to vote for a single action in the voting method.

A.2 Automatic vs. manual threshold

In this experiment we show the impact of an automatically chosen threshold as compared to a manually chosen threshold (see equations 14, 15 and 16 in section 4.2.1). In Fig. 29, the outcome of the experiments are shown for the three different object classes. We focus on the results with highest abstraction and order, meaning Υ_2^β for the open objects and Υ_2 for the box and round objects. For the open objects, we see an improved performance when the manual threshold is used. Both the top point of the curve and the consistency between the selection threshold and the prediction rate on the high end of the selection threshold show superior performance compared to the automatic threshold. For the box- and round objects, the automatic threshold results show slightly better performance as the top point has a higher success rate, although the curve drops earlier than the manually selected threshold.

From these results, it can be derived, that an automatically chosen threshold shows a tendency to smooth the data more. Hence, the correspondence between the selection threshold and the actually success outcome is suboptimal close to the selection threshold of 1.0. However, although the manual chosen threshold shows better consistency between the selection threshold and the actual prediction, it comes with the cost of a lower top point and the need to manually define the threshold for the individual dimensions of the parametrisation.

References

[1] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2008. [Online]. Available: http://ijr.sagepub.com/content/27/2/157.abstract

[2] G. Granlund, "The complexity of vision," *Signal Processing*, vol. 74, 1999.

[3] N. Krüger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. H. Piater, A. J. Rodríguez-Sánchez, and L. Wiskott, "Deep hierarchies in the primate visual cortex: What can we learn for computer vision?" *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1847–1871, 2013.

[4] G. Kootstra, M. Popovic, J. Jørgensen, K. Kuklinski, K. Miatliuk, D. Kragic, and N. Kruger, "Enabling grasping of unknown objects through a synergistic use of edge and surface information," *The International Journal of Robotics Research*, vol. 31, no. 10, pp. 1190–1213, 2012. [Online]. Available: http://ijr.sagepub.com/content/31/10/1190.abstract

[5] A. Treisman and G. Gelade, "A feature integration theory of attention," *Cognitive Psychology*, vol. 12, pp. 97–136, 1980.

[6] L. Montesano and M. Lopes, "Learning grasping affordances from local visual descriptors," in *Proceedings of the 2009 IEEE 8th International Conference on Development and Learning*, ser. DEVLRN '09. Washington, DC, USA: IEEE Computer Society, 2009, pp. 1–6. [Online]. Available: http://dx.doi.org/10.1109/DEVLRN.2009.5175529

[7] M. Stark, P. Lies, M. Zillich, J. Wyatt, and B. Schiele, "Functional object class detection based on learned affordance cues," in *Computer Vision Systems*, ser. Lecture Notes in Computer Science, A. Gasteratos, M. Vincze,

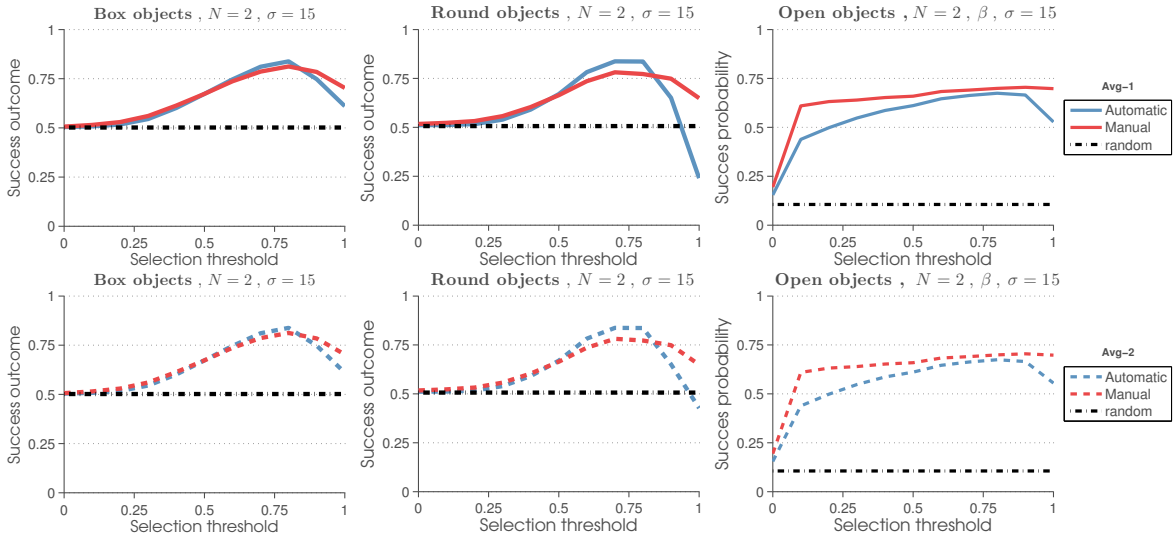


Fig. 29. Automatic vs. manual set threshold for the three different object set, see equations 17–22 for the used parametrisations, and see text for further details.

and J. Tsotsos, Eds. Springer Berlin Heidelberg, 2008, vol. 5008, pp. 435–444. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-79547-6_42

[8] G. Fritz, L. Paletta, M. Kumar, G. Dorffner, R. Breithaupt, and E. Rome, "Visual learning of affordance based cues," in *Proceedings of the 9th International Conference on From Animals to Animats: Simulation of Adaptive Behavior*, ser. SAB'06. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 52–64. [Online]. Available: http://dx.doi.org/10.1007/11840541_5

[9] D. Rao, Q. V. Le, T. Phoka, M. Quigley, A. Sudsang, and A. Y. Ng, "Grasping novel objects with depth segmentation," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 2578–2585.

[10] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Şucan, "Towards reliable grasping and manipulation in household environments," in *Experimental Robotics*. Springer, 2014, pp. 241–252.

[11] J. Stückler, R. Steffens, D. Holz, and S. Behnke, "Efficient 3d object perception and grasp planning for mobile manipulation in domestic environments," *Robotics and Autonomous Systems*, vol. 61, no. 10, pp. 1106–1115, 2013.

[12] M. Richtsfeld and M. Zillich, "Grasping unknown objects based on 21/2d range data," in *Automation Science and Engineering, 2008. CASE 2008. IEEE International Conference on*. IEEE, 2008, pp. 691–696.

[13] K. Huebner, S. Ruthotto, and D. Kragic, "Minimum volume bounding box decomposition for shape approximation in robot grasping," in *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, 2008, pp. 1628–1633.

[14] N. Curtis, J. Xiao, and S. Member, "Efficient and effective grasping of novel objects through learning and adapting a knowledge base," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2008, pp. 2252–2257.

[15] R. Detry, C. H. Ek, M. Madry, and D. Kragic, "Learning a dictionary of prototypical grasp-predicting parts from grasping experience," in *IEEE International Conference on Robotics and Automation*, 2013.

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

[17] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, T. Asfour, and S. Schaal, "Template-based learning of grasp selection," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2379–2384.

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

[19] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *CoRR*, pp. –1–1, 2013.

[20] Y. Jiang, S. Moseson, and A. Saxena, "Efficient grasping from rgbd images: Learning using a new rectangle representation," in *ICRA'11*, 2011, pp. 3304–3311.

[21] S. Fidler, M. Boben, and A. Leonardis, "Learning hierarchical compositional representations of object structure," in *Object Categorization: Computer and Human Vision Perspectives*, S. Dickinson, A. Leonardis, B. Schiele, and M. Tarr, Eds. Cambridge University Press, 2009, pp. 196–215.

[22] D. W. Scott, *Multivariate Density Estimation: Theory, Practice, and Visualization (Wiley Series in Probability and Statistics)*, 1st ed. Wiley, Sept. 1992. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0471547700

[23] R. Becher, P. Steinhaus, R. Zöllner, and R. Dillmann, "Design and implementation of an interactive object modelling system," in *Proceedings Conference Robotik/ISR 2006, München*, May 2006.

[24] archvied3D, "Archive3d free online cad model database," http://www.archive3d.net.

[25] L.-P. Ellekilde and J. A. Jørgensen, "Robwork: A flexible toolbox for robotics research and education," *Robotics (ISR), 2010 41st International Symposium on and 2010 6th*

German Conference on Robotics (ROBOTIK), pp. 1 –7, june 2010.

[26] J. A. Jørgensen, L.-P. Ellekilde, and H. G. Petersen, "RobWorkSim - an Open Simulator for Sensor based Grasping," *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*, pp. 1 –8, june 2010.