Object Learning and Grasping Capabilities for Robotic Home Assistants

S. Hamidreza Kasaei[1(✉)], Nima Shafii[1], Luís Seabra Lopes[1,2], and Ana Maria Tomé[1,2]

[1] IEETA - Instituto de Engenharia Electrónica e Telemática de Aveiro, Universidade de Aveiro, Aveiro, Portugal
{seyed.hamidreza,nima,lsl,ana}@ua.pt
[2] Departamento de Electrónica, Telecomunicações e Informática, Universidade de Aveiro, Aveiro, Portugal

Abstract. This paper proposes an architecture designed to create a proper coupling between perception and manipulation for assistive robots. This is necessary for assistive robots, not only to perform manipulation tasks in reasonable amounts of time, but also to robustly adapt to new environments by handling new objects. In particular, this architecture provides automatic perception capabilities that will allow robots to, (*i*) incrementally learn object categories from the set of accumulated experiences and (*ii*) infer how to grasp household objects in different situations. To examine the performance of the proposed architecture, quantitative and qualitative evaluations have been carried out. Experimental results show that the proposed system is able to interact with human users, learn new object categories over time, as well as perform object grasping tasks.

Keywords: Assistive robots · Object grasping · Object learning and recognition

1 Introduction

Assistive robots are extremely useful because they can help elderly adults or people with motor impairments to achieve independence in everyday tasks [3]. Elderly, injured, and disabled people have consistently put a high priority on object manipulation [6]. On the one hand, a robot capable of performing object manipulation tasks in domestic environments would be worthwhile. On the other hand, this type of end-users expect robots to improve the task performance and to robustly adapt to new environments by handling new objects. In other words, it is not feasible to assume one can pre-program everything for assistive robots. Instead, robots should infer and learn autonomously from experiences, including feedback from human teachers. In order to incrementally adapt to new environments, an autonomous assistive robot must have the abilities to process visual information, infer grasp points and learn and recognize object categories in a concurrent and interleaved fashion.

S. Behnke et al. (Eds.): RoboCup 2016, LNAI 9776, pp. 279–293, 2017.
https://doi.org/10.1007/978-3-319-68792-6_23

However, several state of the art assistive robots employ traditional object grasping and object category learning/recognition approaches, often resorting to complete geometric models of the objects [9,16]. These traditional approaches are often designed for static environments in which it is viable to separate the training (off-line) and testing (on-line) phases. Besides, the knowledge of this kind of robots is static, in the sense that the representation of the known categories does not change after the training stage.

In this paper, a framework for assistive robots is presented which provides a tight coupling between object perception and manipulation. The approach is designed to be used by an assistive robot working in a domestic environment similar to the RoboCup @Home league environment. This work focuses on grasping and recognizing table-top objects. In particular, we present an adaptive object recognition system based on environment exploration and Bayesian learning. Moreover, the robot should manipulate detected objects while working in the environment.

The contributions presented here are the following: *(i)* an integrated framework for assistive robots that incorporates capabilities for object perception, learning and manipulation. *(ii)* unsupervised object exploration for constructing a dictionary of visual words for object representation using the Bag-of-Words model; *(iii)* open-ended learning of object category models from experiences; *(iv)* a data driven grasp pose detection approach for household objects including flat ones.

2 Related Work

Over the past decade, several researches have been conducted to develop assistive robots for motor impairments or elderly people that enable them to stay active and less dependent on others [3]. Jain et al. [6] presented an assistive mobile manipulator, EL-E, that can autonomously pick objects from a flat surface and deliver them to the user. Unlike our approach, the user provides a 3D location of the target object to the robot by pointing on the object with a laser pointer. In another work, a busboy assistive robot has been developed by Srinivasa et al. [14]. This work is similar to ours in that it integrates object perception and grasping for pick and place objects. However there are some differences. Their vision system is designed for detecting mugs only, while our perception system not only tracks the pose of different objects but also recognizes their categories.

Robotic grasping approaches are usually organized in three groups according to the knowledge available about objects: objects can unknown, known or familiar [1]. Grasping approaches for known objects typically use complete geometry of objects to improve grasp quality [1]. However, in real scenarios the complete knowledge about geometry and other properties of objects are not known in advance (objects are initially unknown). Thus, grasping approaches for unknown objects are needed. Hsiao et al. [5] proposed a data driven algorithm that searches among feasible top and side grasp candidates for unknown objects based on their partial view. Its advantage is that no supervised learning is required. In a similar

approach [15] shows, a service robot is capable of grasping different household objects at the RoboCup @Home competition. However, unlike our approach, these grasp approaches are not able to grasp flat objects e.g., plates. Another goal in the field of assistive and service robots is to achieve interactive object category learning and recognition. Martinez et al. [11] described a fast and scalable perception system for object recognition and pose estimation.

In most of the proposed systems described above, training and testing are separate processes, which do not occur simultaneously. There are some approaches which support incremental learning of object categories. Hamidreza Kasaei et al. [7] and Oliveira et al. [13] approached the problem of object experience gathering and category learning with a focus on open-ended learning and human-robot interaction. They used instance-based learning approach to describe object categories whereas we employ a model based approach in which a Naive Bayes learning method is used.

3 Overall System Architecture

The overall system architecture is depicted in Fig. 1. It is a reusable framework and all modules were developed over Robot Operating System (ROS). The current architecture is an evolution of the architecture developed in previous work for object perception and open-ended perceptual learning [7,13]. The architecture consists of two memory systems, namely the *Working Memory* and the *Perceptual Memory*. Both memory systems have been implemented as a lightweight NoSQL database namely LevelDB developed by Google. *Working Memory* is used for temporarily storing and manipulating information and communications of all modules. A visual words dictionary, object representation data and object category knowledge are stored into the *Perceptual Memory*. The goal of the *Grasp Selection* is to extract a grasp pose (i.e. a gripper pose relative to the

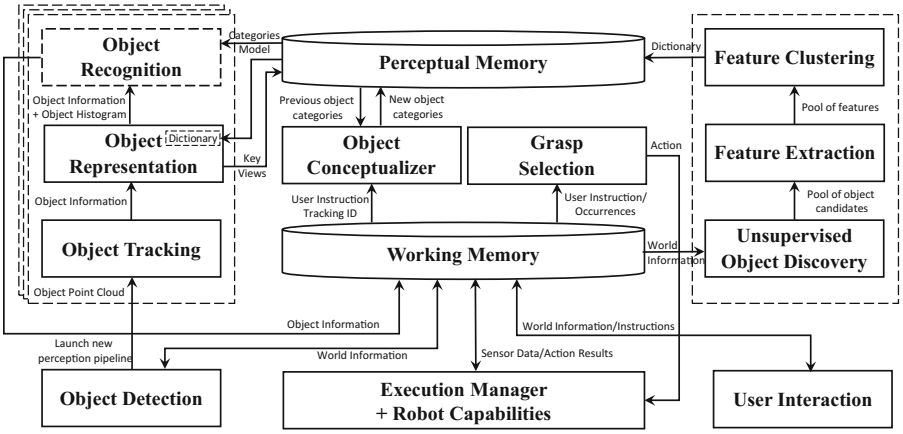

Fig. 1. Overall architecture of the proposed system

object) using the presented grasp approach (Sect. 7). The *Execution Manager* module receives the action and dispatches it to the robot platform as well as records success or failure information into the *Working Memory*.

The proposed architecture includes two perceptual learning components. The first component is concerned with building a visual words dictionary for object representation. The dictionary plays a prominent role because will be used for learning as well as recognition. The second component focuses on interactive object category learning and recognition. After constructing the dictionary, when the robot captures a scene, the first step is preprocessing, which employs three filtering procedures for removing unnecessary data. The *Object Detection* module is responsible for detecting objects in the scene. It creates a new perception pipeline for every detected object. Each pipeline includes *Object Tracking*, *Object Representation* and *Object Recognition* modules. The *Object Tracking* module estimates the current pose of the object based on a particle filter, which uses shape as well as color data [12]. The *Object Representation* module describes objects as histograms of visual words and stores them into the *Perceptual Memory*. A user can provide category labels for these objects via the *User Interaction* module [10]. Whenever the instructor provides a category label for an object, the *Object Conceptualizer* improves or creates a new object category model. In recognition situations, a probabilistic classification rule is used to assign a category label to the detected object. In the following sections, the characteristics of the object perception, learning and grasping modules are explained in detail.

4 Dictionary Construction

Comparing 3D objects by their local features would be computationally expensive. To address this problem, an approach for object representation is adopted in which objects are described by histograms of local shape features, as defined in Bag-of-Words models. A Bag-of-Words model requires a dictionary of visual words. Usually, the dictionary is created via off-line clustering of training data, while in open-ended learning, there is no predefined set of training data available at the beginning of the learning process. To cope with this limitation, we propose that the robot freely explores several scenes and collects several object experiences. In general, object exploration is a challenging task because of the dynamic nature of the world and ill-definition of the objects [4].

In the following, we used boolean expressions to specify object perception capabilities(see Eqs. 1 and 2). In both *object exploration* and *object detention* cases, we assume that interesting objects are on tables and the robot seeks to detect tabletop objects (i.e. C_{table}). On the one hand, to represent an object, it is important to store only different views, which is possible when the object is moved. On the other hand, storing all object views while the object is static would lead to unnecessary accumulation of highly redundant data. Hence, the C_{track} constraint is means the object candidate is already being tracked. Moreover, $C_{\text{instructor}}$ and C_{robot} are exploited to filter out object candidates corresponding to the instructor's body as well as robot's body. Accordingly, the

resulting object candidates are less noisy and include only data corresponding to the objects:

$$\psi_{\text{exploration}} = C_{\text{table}} \ \wedge \ C_{\text{track}} \ \wedge \ \neg \left(C_{\text{instructor}} \ \vee \ C_{\text{robot}}\right) \tag{1}$$

In our current setup, a table is detected by finding the dominant plane in the point cloud. This is done using a RANSAC algorithm. Extraction of polygonal prisms is used for collecting the points which lie directly above the table. Afterwards, an Euclidean Cluster Extraction algorithm is used to segment a scene into individual clusters. Every cluster that satisfy the exploration expression is selected. The output of this object exploration is a pool of object candidates. Subsequently, to construct a pool of features, spin-images are computed for the selected points extracted from the pool of object candidates. It should be noted that to balance computational efficiency and robustness, a downsampling filter is applied to obtain a smaller set of points distributed over the surface of the object. We use a PCL function to compute spin-images. Finally, the dictionary is constructed by clustering the features using the *k-means* algorithm. The centres of the N (i.e. $N = 90$) extracted clusters define the visual words, $\mathbf{w}_t$ $(1 \leq t \leq N)$. Figure 2 shows the procedure of constructing a dictionary of visual words. A video of the system that a robot explores an environment[1] is available in: http://youtu.be/MwX3J6aoAX0.

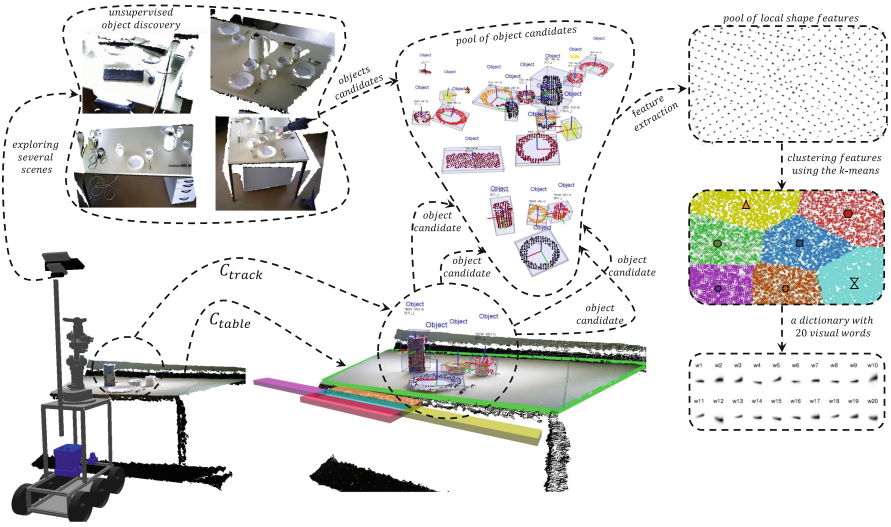

Fig. 2. The robot moves through an office to explore several scenes and extract tabletop objects to construct a dictionary of visual words.

[1] The ROS bag file used in this video was created by the Knowledge-Based Systems Group, Institute of Computer Science, University of Osnabrueck.

5 Object Detection and Representation

For fast processing of massive point clouds, two filters are used, namely distance filtering and downsampling [7]. Furthermore, knowledge of the positions of the arm joints relative to the camera pose is retrieved from the *Working Memory* and sensor data (i.e. points) corresponding to robot's body is filtered out from the original point cloud. After preprocessing, the next step is to find objects in the scene. The object detection module implements the following specification:

$$\psi_{\text{detection}} = C_{\text{table}} \wedge C_{\text{track}} \wedge C_{\text{size}} \wedge \neg (C_{\text{instructor}} \vee C_{\text{robot}} \vee C_{\text{edge}}) \quad (2)$$

The object detection uses a size constraint, C_{size}, to detect objects which can be manipulated by the robot. Moreover, a C_{edge} constraint is considered to filter out the segmented point clouds that are too close to the edge of the table. The object detection then assigns a new *Track ID* to each newly detected object and launches an object perception pipeline for the object candidate as well. The object detection pushes the segmented object candidates into the perception pipelines for subsequent processing steps.

The *Object Tracking* module is responsible for keeping track of the target object over time while it remains visible [12]. It receives the point cloud of the detected object and computes its geometric center of object view as the position of the object. The object tracking sends out the tracked object information to the *Object Representation* module.

The input to the object representation module is a point cloud of an object candidate **O**. The object representation module involves three main phases: keypoint extraction, computation of spin images for each keypoint and, finally, representing an object view as a histogram of visual words. For keypoint extraction, a voxelized grid approach is used to obtain a smaller set of points by taking only the nearest neighbor point for each voxel center [7]. Afterwards, the spin-image descriptor is used to encode the surrounding shape in each keypoint using the original point cloud. By searching for the nearest neighbor in the dictionary, each local shape is assigned to a visual word. Finally, each object is represented as a histogram of occurrences of visual words, $\mathbf{h} = [h_1\ h_2 \ldots h_N]$, where the i^{th} element of $\mathbf{h}$ is the count of the number features assigned to a visual word, $\mathbf{w}_i$. The obtained histogram of the given object is dispatched to the *Object Recognition* module and is recorded into the *Perceptual Memory* if it is marked as a key view.

6 Interactive Object Category Learning

Human-robot interaction is essential for supervised experience gathering, i.e. for instructing the robot how to perform different tasks. Particularly, an open-ended object learning and recognition system will be more flexible if it is able to learn new objects from a human user. For example, if the robot does not know how a *'Mug'* looks like, it may ask the user to show one. Such situation provides an opportunity to collect training instances from actual experiences of the robot and

the system can incrementally update it's knowledge rather than retraining from scratch when a new instance is added or a new category is defined. The details of the interaction module and supervised object experience gathering is discussed in [10]. The *Object Conceptualizer* (category learning) module is activated when the instructor provides a category label for the object.

6.1 Object Conceptualizer

In this work, object category learning is a process of computing a Bayesian model for each object category. There are two reasons why Bayesian learning is useful for open-ended learning. One of them is the computational efficiency of Bayes approaches. In fact, the model can be incrementally updated when new information is available, rather than retrained from scratch. Second, open-ended systems usually have limited amount of memory available and therefore, it must involve experience management to prevent the accumulation of experiences. In Bayesian learning, new experiences are used to update category models and then the experiences are forgotten.

The probabilistic category model requires calculating the likelihoods of the object given the category k, $p(\mathbf{O}|C_k)$, and it is also parametrized by the prior probabilities $p(C_k)$. The likelihoods of objects in each category, $p(\mathbf{O}|C_k)$, cannot be estimated directly. To make it tractable, we assume that visual words of objects are independent given the category. Therefore, the $p(C_k)p(\mathbf{O}|C_k)$ is equivalent to the joint probability model $p(C_k, \mathbf{w}_1, \ldots, \mathbf{w}_n) = p(C_k)\ p(\mathbf{w}_1, \ldots, \mathbf{w}_n|C_k)$. The joint model can be rewritten using conditional independence assumptions:

$$p(C_k|\mathbf{w}_1, \ldots, \mathbf{w}_n) \propto p(C_k) \prod_{i=1}^{n} p(\mathbf{w}_i|C_k), \tag{3}$$

where n is the size of the dictionary and $p(\mathbf{w}_i|C_k)$ is the probability of the visual word $\mathbf{w}_i$ occurring in an object of category k:

$$p(\mathbf{w}_i|C_k) = \frac{s_{ik}\ + 1}{\sum_{j=1}^{n}(s_{jk} + 1)}, \tag{4}$$

where s_{ik} is the number of times that word $\mathbf{w}_i$ was seen in objects from category C_k. On each newly seen object of this category with h_i features of type $\mathbf{w}_i$, the following update is carried out: $s_{ik} \leftarrow s_{ik} + h_i$. The prior probability of category k, $p(C_k)$, is estimated by dividing the number of seen objects from category k by the total number of seen objects in all categories.

6.2 Object Category Recognition

The last part of object perception is object category recognition. To classify an object $\mathbf{O}$, which is represented as a histogram of occurrences of visual words

$\mathbf{h} = [h_1, \ldots, h_n]$, the posterior probability for each object category $p(C_k|\mathbf{h})$ is considered as the object-category similarity (i.e. OCS(.)) and approximated using Bayes theorem as:

$$\text{OCS}(\mathbf{O}, C_k) = \frac{p(\mathbf{h}|C_k)p(C_k)}{p(\mathbf{h})}, \tag{5}$$

Since the denominator does not depend on C_k, it is constant and can be ignored. Equation 5 is re-expressed based on Eq. 3 and multinomial distribution assumption. In addition, to avoid underflow problems, the logarithm of the likelihood is computed:

$$\text{OCS}(\mathbf{O}, C_k) = \log p(C_k) + \sum_{i=1}^{n} h_i \ \log p(\mathbf{w}_i|C_k), \tag{6}$$

The category of the target object **O** is the one with highest likelihood. If, for all categories, the OCS(.) is smaller than a given threshold (e.g. CT = 0.75), then the object is classified as *Unknown*; otherwise, it is classified as the category that has the highest similarity. Consequently, object information including object recognition result, point cloud and global characteristics of the given object such as main axis, center and size of bounding box are written to the *Working Memory*, where the grasp selection module can fetch them to support object manipulation. *Grasp Selection* is triggered when a user instructs the robot to perform a task (e.x. *serve a meal*).

7 Grasp Methodology

In this work, we assume that it is possible to select suitable grasps pose for different household objects by only using partial object views. The grasp selection module retrieves the point cloud of a given object from working memory (see Fig. 1). The point cloud is then processed to determine an appropriate object's bounding box and reference frame. For constructing the object-centric reference frame, the vertical direction perpendicular to the table is assigned to the local z-axis since objects are assumed to be placed on a table. Principal Component Analysis (PCA) is used to compute the axes of minimum and maximum variance in the horizontal plane using points projected on the surface plane. The maximum variance axis is assigned to the x-axis. Although the result of PCA indicates the lines align to the eigen vectors, but the sign and direction of this line remains ambiguous to calculate x-axis. The direction of the x-axis is defined as the oposite direction to the origin of the arm base frame. Then, the bounding box of the object calculated along these axes is computed and the center of the bounding box is defined as the origin of the object's coordinate system. Since the object is only partially visible, the center of the object's bounding box is used as a proxy for the true center of mass. In this study, we use a set of heuristic grasp strategies. They are defined as follows:

Top grasp strategy: It is assumed that the object is only approached by the robot along the z-axis (perpendicular to the surface) while the gripper closing direction is aligned with the y-axis. Thus, in order to find the suitable grasp pose on the object, it is only needed to find the grasp position on x-axis. To do that, first, the object points are projected onto the xy-plane. Then, the points are clustered using equally sized intervals along x-axis. Each cluster indicates one grasp candidate. For each one, the maximum and the minimum of the positions of points along y-axis indicate the grasp width. In order to select a proper grasp candidate, since grasping around the center of mass of an object is preferable, the cluster around the origin is the first to be analyzed. This grasp minimizes the torque about the gripper axis due to the objects weight. If the grasp width of the selected candidate is larger than the width of the gripper (i.e. 12 cm), the other grasp candidates will be analyzed. The grasp candidates are then sorted based on how much their grasp width fits to the robot's palm. The size of palm of the JACO arm is 5 cm. Note, the grasp candidates that reach to the limits of the x-axis are rejected. Figure 3 (*left*) shows the process of the top grasps approach.

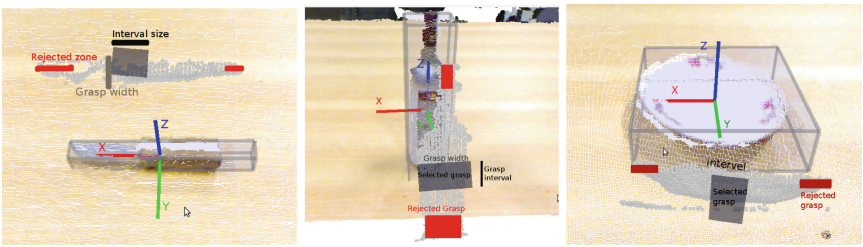

Fig. 3. The projected objects' points on the planes of the object bounding box (gray box) are analyzed to select the grasp pose (black square). The bad grasp poses (red squares) are rejected. (Color figure online)

Horizontal side grasp strategy: According to this grasp strategy, the object is approached in the horizontal plane, along the y-axis while negative y is used as approach direction and the gripper closing direction is aligned with the x-axis. In this case, the proper grasp position along z-axis should be found. To do that, first, object points are projected onto the xz-plane, then multiple grasp candidates are generated by sampling along the z-axis with equal interval size. Using a strategy similar to the one described for top grasps, the grasp candidates are analysed and one of them is selected. The grasp candidates located in the limits of the z-axis are also rejected. The process of the grasp selection using the horizontal side strategy is depicted in Fig. 3 (*center*).

Vertical side grasp strategy: In this strategy, the object is approached in the vertical plane along the y-axis. The approach direction is negative y and the gripper closing direction is aligned with the z-axis. This grasp strategy is designed to be used in grasping flat objects such as plates. In this strategy, the proper grasp position along x-axis should be inferred. To do that, first,

the object's points are projected onto the xz-plane, then multiple grasp candidates are extracted by sampling along the y-axis with equal interval size. Like above strategies, the centered grasp candidate is selected if it fits inside the gripper. Otherwise, other grasp candidates are ranked as in the top grasp strategy. The grasp candidates located in the limits of the x-axis are also rejected. Figure 3 (*right*) shows the grasp selection process using vertical side grasp strategy. In our grasp methodology, first, one of the proposed grasp strategies is selected based on the size of the bounding box of the object. To do that the following rules are presented in Algorithm 1. Where |.| returns the size of object along an specific axis. Finally, the JACO arm robot is commanded to perform actions based on the selected grasp position and strategy. Inverse kinematic integrated from the JACO arm driver is used to control the end-effector pose goal. In a grasping scenario, first the robot is commanded to go to a pre-grasp position that is 0.2 cm behind the grasp pose along with the grasp orientation. The grasp orientation is given by grasp strategy. When the pre-grasp pose is reached, the robot approaches the grasp point and then closes the gripper. Afterwards, the height of the robot's end-effector in the arm-frame is recorded by the robot in working memory to be used as the desired height for placing the grasped object. Whenever the object is grasped, the robot picks up the object and navigates it to the predefined placing pose.

Algorithm 1 rule of grasp strategy selection

```
if |z-axis| > |x-axis| and |z-axis| > |y-axis| then
    perform top grasp
else if |x-axis| > |z-axis| and |y-axis| > |z-axis| then
    perform vertical side grasp
else
    perform horizontal side grasp
end if
```

8 Experimental Results

Four types of experiments were performed to evaluate the proposed approach.

Table 1. Average object recognition performance (F1 measure) for different parameters

Parameters	VS			DS					IW		SL			
Values	0.01	0.02	0.03	50	60	70	80	90	4	8	0.02	0.03	0.04	0.05
Average F1	0.76	0.74	0.71	0.72	0.73	0.74	0.74	0.75	0.75	0.72	0.63	0.74	0.78	0.79

8.1 Off-line Evaluation of the Object Learning Approach

To examine the performance of different configurations of the proposed object learning approach (Sect. 6), a 10-fold cross validation scheme has been followed. For this purpose, an object dataset namely Restaurant Object Dataset [7] has been used which contains 339 views of 10 categories of objects. A total of 120 experiments were performed for different values of four parameters of the system, namely the voxel size (VS) which is related to number of keypoints extracted from each

object view, the dictionary size (DS), the image width (IW) and support length (SL) of spin images. Results are presented in Table 1. The combination of parameters that obtained the best average F1 score was selected as the default system parameters. They are the following: VS = 0.01, DS = 90, IW = 4 and SL = 0.05. The results presented in Sects. 8.2 and 8.4 are computed using this configuration.

8.2 Open-Ended Evaluation

The off-line evaluation methodologies (e.g. k-fold cross validation, leave- one-out, etc.) are not well suited to evaluate open-ended learning systems, because they do not abide to the simultaneous nature of learning and recognition and those methodologies imply that the number of categories must be predefined. An evaluation protocol for open-ended learning systems was proposed in [2]. A *simulated teacher* was developed to follow the teaching protocol and autonomously interact with the system using three basic actions namely ***teach***, used for teaching a new object category, ***ask***, used to ask the system what is the category of an object view and ***correct***, used for providing the system corrective feedback in case of misclassification. The idea is that, for each newly taught category, the simulated teacher repeatedly picks unseen object views of the currently known categories from a dataset and presents them to the system. It progressively estimates recognition performance of the system and, in case this performance exceeds a given threshold (CT = 0.67), introduces an additional object category. In this way, the system is trained and tested at the same time. Experiments were running the largest publicly available dataset namely RGB-D Object Dataset consisting of 250,000 views of 300 common household objects, organized into 49 categories [8].

When an experiment is carried out, learning performance is evaluated using several measures, including: (*i*) The number of learned categories at the end of an experiment (*TLC*), an indicator of ***how much the system is capable of learning***; (*ii*) The number of question/correction iterations (*QCI*) required to learn those categories and the average number of stored instances per category (*AIC*), indicators of ***time and memory resources required for learning***; (*iii*) Global classification accuracy (*GCA*), an F-measure computed using all predictions in a complete experiment, and the average classification accuracy (*ACA*), indicators of ***how well the system learns***.

Since the order of introduction of new categories may have an effect on the performance of the system, ten experiments were carried out in which categories were introduced in random sequences. In the additional nine experiments, these categories were used again with different introduction sequences, which are reported in Table 2. By comparing all experiments, it is visible that

Table 2. Summary of experiments.

EXP#	#QCI	#TLC	#AIC	GCA (%)	ACA (%)
1	1257	49	8.16	79	83
2	1238	49	7.83	80	84
3	1227	49	7.65	81	84
4	1240	49	9.08	75	78
5	1236	49	7.95	80	83
6	1346	49	9.46	76	79
7	1293	49	9.02	77	81
8	1330	49	9.79	74	79
9	1336	49	9.55	75	78
10	1225	49	8.30	78	82

in the third experiment, the system learned all categories faster than other experiments. In the case of experiment 9, the number of iterations required to learn 49 object categories was greater than other experiments.

Figure 4 (*left*) shows the global classification accuracy as a function of the number of learned categories. In this figure we can see that the global classification accuracy decreases as more categories are learned. This is expected since the number of categories known by the system makes the classification task more difficult. Finally, Fig. 4 (*right*) shows the number of learned categories as a function of the protocol iterations. This gives a measure of how fast the learning occurred in each of the experiments.

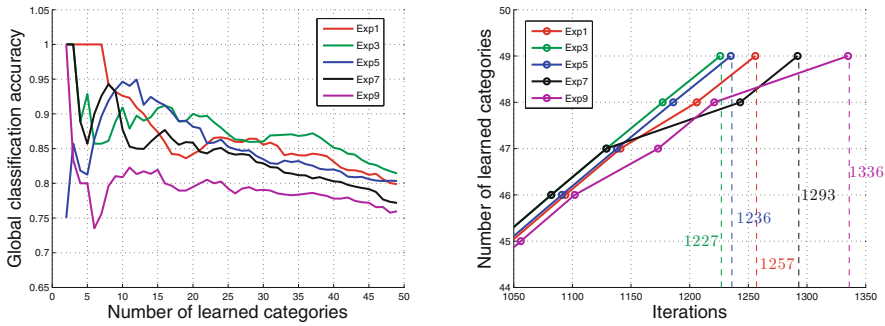

Fig. 4. System performance during simulated user experiments.

8.3 Grasp Evaluation

In order to evaluate the quality of the proposed grasp approach, a grasp scenario has been designed. In the grasp scenario, the JACO robot is instructed to pick-up an object using the proposed grasp methodology. After picking up the object, the robot carries the object to the placing position to see if the object slips due to bad grasp or not. A particular grasp is considered as success if the robot performed the scenario successfully. We analyzed the performance of our approach to grasp the household objects by evaluating the success rate. In our designed scenarios, 21 household objects were used which can be seen in Fig. 5.

As a testing scenario, the objects were placed in different orientations. In each experiment, an object was first put in the orientation shown by the object view in the Fig. 5 and robot tried to grasp it. Afterwards, we rotated the object about 60° for six times and repeated the scenario to test all viewpoints of the object. Therefore, each object was tested to be grasped by the robot 6 times, and 126 grasp trials were performed to complete the whole experiments. In these experiments, the robot could perform 111 successful grasps meaning that the overall success rate was about 88%. It was observed that the robot could grasp all objects by using the proposed strategies.

Fig. 5. The objects used as a test set to evaluate the proposed grasp approach.

8.4 System Demonstration

A *serve a meal* scenario has been designed to show all functionalities of the object recognition and grasping. In this demonstration, the system works in a scenario where a table is in front of the robot and a user interacts with the system. Note that, when the system starts, the set of categories known to the system is empty. In the session, a user presents objects to the system and provides the respective category labels. The user then instructs the robot to perform a *serve a meal* task (i.e. puts different restaurant objects on the table in front of the user). To achieve this task, the robot must be able to detect and recognize different objects and transport the objects to the predefined areas and completely serve a meal. For this purpose, the robot retrieves the world model information from the *Working Memory* including label and position of all active objects. The robot then chooses the object that is nearest to the arm's base and serves it to the user. A video of this session is available at: https://youtu.be/GtXBiejdccw. This small demonstration shows that the developed system is capable of detecting new objects, tracking and recognizing as well as manipulating objects in various positions.

9 Conclusion

In this paper, we presented an architecture designed to support a coupling between perception and manipulation for service robots. In particular, an interactive open-ended learning approach for acquiring 3D object categories and a data driven approach for object grasping have been presented, which enable robots to adapt to different environments and reason out how to behave in response to a complex task such as *serve a meal*. This paper also proposes unsupervised object exploration to construct the visual word dictionary and an incremental Bayesian learning approach for object category learning.

We have also tried to make the proposed architecture easy to integrate on other robotic systems. Our approach to object perception has been successfully tested on a JACO arm, showing the importance of having a tight coupling between perception and manipulation. For future work, we would like to investigate the possibility of improving performance of object grasping based on kinesthetic teaching and improving performance of object recognition using topic modelling.

Acknowledgement. This work was funded by National Funds through FCT project PEst-OE/EEI/UI0127/2016 and FCT scholarship SFRH/BD/94183/2013.

References

1. Bohg, J., Morales, A., Asfour, T., Kragic, D.: Data-driven grasp synthesis – a survey. IEEE Trans. Robot. **30**(2), 289–309 (2014)
2. Chauhan, A., Lopes, L.S.: Using spoken words to guide open-ended category formation. Cogn. Process. **12**(4), 341–354 (2011)
3. Ciocarlie, M., Hsiao, K., Jones, E.G., Chitta, S., Rusu, R.B., Şucan, I.A.: Towards reliable grasping and manipulation in household environments. In: Khatib, O., Kumar, V., Sukhatme, G. (eds.) Experimental Robotics. Springer Tracts in Advanced Robotics, vol. 79, pp. 241–252. Springer, Heidelberg (2014). doi:10.1007/978-3-642-28572-1_17
4. Collet, A., Xiong, B., Gurau, C., Hebert, M., Srinivasa, S.S.: Herbdisc: towards lifelong robotic object discovery. Int. J. Robot. Res. **34**(1), 3–25 (2015). doi:10.1177/0278364914546030
5. Hsiao, K., Chitta, S., Ciocarlie, M., Jones, E.G.: Contact-reactive grasping of objects with partial shape information. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1228–1235. IEEE (2010)
6. Jain, A., Kemp, C.C.: El-E: an assistive mobile manipulator that autonomously fetches objects from flat surfaces. Auton. Robot. **28**(1), 45–64 (2010)
7. Hamidreza Kasaei, S., Oliveira, M., Lim, G.H., Lopes, L.S., Tomé, A.M.: Interactive open-ended learning for 3D object recognition: an approach and experiments. J. Intell. Robot. Syst. **80**, 1–17 (2015)
8. Lai, K., Bo, L., Ren, X., Fox, D.: A large-scale hierarchical multi-view RGB-D object dataset. In: 2011 IEEE International Conference on Robotics and Automation (ICRA), pp. 1817–1824, May 2011
9. Leroux, C., Lebec, O., Ghezala, M.B., Mezouar, Y., Devillers, L., Chastagnol, C., Martin, J.C., Leynaert, V., Fattal, C.: ARMEN: assistive robotics to maintain elderly people in natural environment. IRBM **34**(2), 101–107 (2013)
10. Lim, G.H., Oliveira, M., Mokhtari, V., Hamidreza Kasaei, S., Chauhan, A., Seabra Lopes, L., Tome, A.: Interactive teaching and experience extraction for learning about objects and robot activities. In: 2014 RO-MAN: The 23rd IEEE International Symposium on Robot and Human Interactive Communication, pp. 153–160, August 2014
11. Martinez Torres, M., Collet Romea, A., Srinivasa, S.: Moped: a scalable and low latency object recognition and pose estimation system. In: IEEE International Conference on Robotics and Automation, (ICRA 2010), May 2010
12. Oliveira, M., Lim, G.H., Seabra Lopes, L., Hamidreza Kasaei, S., Tome, A., Chauhan, A.: A perceptual memory system for grounding semantic representations in intelligent service robots. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE (2014)
13. Oliveira, M., Lopes, L.S., Lim, G.H., Hamidreza Kasaei, H., Tomé, A.M., Chauhan, A.: 3D object perception and perceptual learning in the RACE project. Robot. Auton. Syst. **75**, 614–626 (2016). Part B
14. Srinivasa, S., Ferguson, D.I., Vande Weghe, M., Diankov, R., Berenson, D., Helfrich, C., Strasdat, H.: The robotic busboy: steps towards developing a mobile robotic home assistant. In: International Conference on Intelligent Autonomous Systems, pp. 2155–2162 (2008)

15. Stückler, J., Steffens, R., Holz, D., Behnke, S.: Efficient 3D object perception and grasp planning for mobile manipulation in domestic environments. Robot. Auton. Syst. **61**(10), 1106–1115 (2013)
16. Vahrenkamp, N., Do, M., Asfour, T., Dillmann, R.: Integrated grasp and motion planning. In: 2010 IEEE International Conference on Robotics and Automation (ICRA), pp. 2883–2888. IEEE (2010)