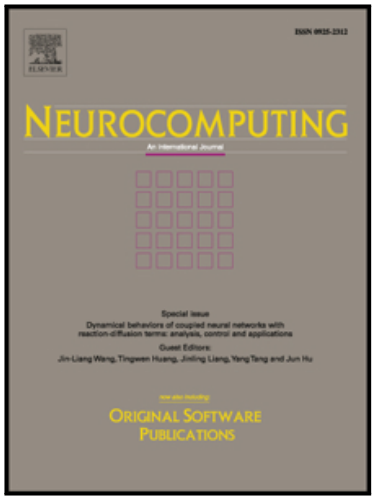
Communicated by Prof. Zidong Wang

Accepted Manuscript



Towards Lifelong Assistive Robotics: A Tight Coupling between Object Perception and Manipulation

S.Hamidreza Kasaei, Miguel Oliveira, Gi Hyun Lim,
Luís Seabra Lopes, Ana Maria Tomé

PII: S0925-2312(18)30232-7
DOI: 10.1016/j.neucom.2018.02.066
Reference: NEUCOM 19370

To appear in: *Neurocomputing*

Received date: 15 June 2015
Revised date: 21 February 2018
Accepted date: 21 February 2018

Please cite this article as: S.Hamidreza Kasaei, Miguel Oliveira, Gi Hyun Lim, Luís Seabra Lopes, Ana Maria Tomé, Towards Lifelong Assistive Robotics: A Tight Coupling between Object Perception and Manipulation, *Neurocomputing* (2018), doi: 10.1016/j.neucom.2018.02.066

Towards Lifelong Assistive Robotics: A Tight Coupling between Object Perception and Manipulation

S. Hamidreza Kasaei[a], Miguel Oliveira[a,b], Gi Hyun Lim[a], Luís Seabra Lopes[a,c], Ana Maria Tomé[a,c]

[a]*IEETA - Instituto de Engenharia Electrónica e Telemática de Aveiro, Universidade de Aveiro, Portugal*
[b]*Instituto de Engenharia de Sistemas e Computadores, Tecnologia Ciência R. Dr. Roberto Frias, 465, 4200 Porto, Portugal*
[c]*Departamento de Electrónica, Telecomunicações e Informática, Universidade de Aveiro, Portugal*

Abstract

This paper presents an artificial cognitive system tightly integrating object perception and manipulation for assistive robotics. This is necessary for assistive robots, not only to perform manipulation tasks in a reasonable amount of time and in an appropriate manner, but also to robustly adapt to new environments by handling new objects. In particular, this system includes perception capabilities that allow robots to incrementally learn object categories from the set of accumulated experiences and reason about how to perform complex tasks. To achieve these goals, it is critical to detect, track and recognize objects in the environment as well as to conceptualize experiences and learn novel object categories in an open-ended manner, based on human-robot interaction. Interaction capabilities were developed to enable human users to teach new object categories and instruct the robot to perform

*Corresponding author - S. Hamidreza Kasaei
Email addresses: seyed.hamidreza@ua.pt (S. Hamidreza Kasaei), mriem@ua.pt (Miguel Oliveira), lim@ua.pt (Gi Hyun Lim), lsl@ua.pt (Luís Seabra Lopes), ana@ua.pt (Ana Maria Tomé)

complex tasks. A naive Bayes learning approach with a Bag-of-Words object representation are used to acquire and refine object category models. Perceptual memory is used to store object experiences, feature dictionary and object category models. Working memory is employed to support communication purposes between the different modules of the architecture. A reactive planning approach is used to carry out complex tasks. To examine the performance of the proposed architecture, a quantitative evaluation and a qualitative analysis are 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 complex tasks.
Keywords: Assistive robots; 3D object perception; open-ended learning; interactive learning; object manipulation.

1. Introduction

Assistive robots are extremely useful because they can help elders or people with motor impairments to achieve independence in everyday tasks[1][2]. Elderly, injured, and disabled people have consistently attributed a high priority to object manipulation tasks[3]. Object manipulation tasks consist of two phases: the first is the perception of the object and the second is the planning and execution of arm or body motions which grasp the object and carry out the manipulation task. These two phases are closely related: object perception provides information to update the model of the environment, while planning uses this world model information to generate sequences of arm movements and grasp actions for the robot. In addition, assistive robots must perform the tasks in reasonable time. It is also expected that the competence of the robot increases over time, that is, robots must robustly adapt to new environments by being capable of handling new objects. However,

it is not reasonable to assume that one can pre-program all necessary object categories for assistive robots. Instead, robots should learn autonomously from novel experiences, supported in the feedback from human teachers. In order to incrementally adapt to new environments, an autonomous assistive robot must have the ability to process visual information and conduct learning and recognition tasks in a concurrent and interleaved fashion. Several state-of-the-art assistive robots use traditional object category learning and recognition approaches [4][5][6]. These classical approaches are often designed for static environments in which it is viable to separate the training (off-line) and testing (on-line) phases. In these cases, the world model is static, in the sense that the representation of the known categories does not change after the training stage. Therefore, these robots are unable to adapt to dynamic environments [7]. This leads to several shortcomings such as the inability to detect/recognize new or unknown categories. To cope with these issues, several cognitive robotics groups have started to explore how robots could learn incrementally from their own experiences as well as from interaction with humans [8][9][10].

In this paper, a cognitive 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. In particular, we present an adaptive object perception system based on environment exploration and Bayesian learning. The objective is that the robotic system is capable of continuously learning new object categories while carrying out manipulation tasks in the environment. This work focuses on learning, recognizing and manipulating table-top objects.

The contributions proposed in this work are the following: *(i)* an integrated

framework for object manipulation incorporating perception and planning capabilities for manipulation tasks; *(ii)* unsupervised object exploration methodology that produces a dictionary of visual words used for representing objects (Bag-of-Words model); *(iii)* interactive categorization (labelling) of physical objects, in which a human user playing the role of tutor provides category labels for objects under shared attention; *(iv)* open-ended learning of object category models from experiences. The fourth contribution follows our previous works on open-ended learning for object recognition [11] [12] [13] [14]. These previous approaches are instance-based, i.e. a set of features is stored for each object view. In contrast, the present work uses a Naive Bayes learning method to compute category models from the observed views of instances of the categories. Furthermore, manipulation experiments are carried out for validating the approach.

The remainder of the paper is organized as follows: section 2 describes the related work; an overview of the developed system is presented in section 3; sections 4, 5, 6 and 7 describe in detail the proposed methodologies. Finally, results are presented and discussed in section 8 and conclusions are presented in section 9.

2. Related Work

Although an exhaustive survey of assistive robotics as well as object perception and manipulation techniques is beyond the scope of this paper, representative works will be reviewed in this section.

2.1. Assistive and Service Robots

Daily tasks such as setting a table for a meal or cleaning a table are difficult for disabled or elder people [2]. Over the past decade, several researches have

been conducted to develop robots to assist those people in order to enable them to maintain an active life less dependent on others [1]. In the ARMEN project, Leroux et al. [4] proposed a mobile assistive robotics approach providing advanced functions to help maintaining elderly or disabled people at home. Similar to our system, this project involves object manipulation, knowledge representation and object recognition. The authors also developed an interface to facilitate the communication between the user and the robot. Jain et al. [3] presented an assistive mobile manipulator named EL-E that can autonomously pick objects from a flat surface and deliver them to the users. They used a multi-step control policy that is not suitable to achieve real time performance. In our approach we can achieve real-time performance through the use of ROS nodelets and multiplexing mechanisms [12]. Furthermore, in [3], the user provides the location of the object to be grasped by the robot by briefly illuminating a location with a laser pointer. In this work, objects are detected and recognized autonomously. Therefore it is enough for the user to specify the category of the object to be picked up.

In another work [15], a multi-robot assistive system, consisting of a Segway mobile robot with a tray and a stationary Barrett WAM robotic arm, was developed. The Segway robot navigates through the environment and collects empty mugs from people. Then, it delivers the mugs to a predefined position near the Barrett arm. Afterwards, the arm detects and manipulates the mugs from the tray and loads them into a dishwasher rack. This work is similar to ours in that it integrates perception and motion planning for pick and place operations. However there are some differences: their vision system is designed for detecting a single object type (mugs), while our perception system not only tracks the pose of different types of objects but also recognizes their categories. Furthermore, because

there is a single object type (i. e. mug), they computed the set of grasp points off-line. In our approach, grasping must handle a variety of objects never seen before.

In the RACE project (Robustness by Autonomous Competence Enhancement), a PR2 robot demonstrated effective capabilities in a restaurant scenario including the ability to serve a coffee, set a table for a meal and clear a table [16] [17] [18]. The aim of RACE was to develop a cognitive system, embodied by a service robot, which enabled the robot to build a high-level understanding of the world by storing and exploiting appropriate memories of its experiences. Other examples of assistive robot platforms that have demonstrated perception and action coupling include TUM Rosie robot [5], HERB [19] and ARMAR-III [6].

2.2. Object Manipulation

In most cases, prior works on object manipulation requires a complete geometric description of the objects [20][21]. However, in real scenarios, it is not possible to have complete knowledge of the geometric properties of all possible objects in advance. That information has to be extracted online from the experiences of the robot. In neuroscience and neurocomputing literature, it has been demonstrated that visual processing in the ventral and dorsal pathways is based on classifying the grasped objects into three groups: *known*, *familiar* and *unknown* objects [1][22][23][24][25]. This classification has been adopted in robotics [20].

The underlying reason for this classification is that prior knowledge about objects determines how grasp candidates are generated and ranked. For *known* objects, i.e., when there is complete knowledge of the geometric properties of objects, grasping is limited to solving the problems of recognition and pose estimation. In the case of *familiar* objects, an object comparison procedure may

be used to compare the given object with known objects, and to define grasping strategies based on that [26]. For *unknown* objects, heuristic methods are used to extract grasps in run-time from 3D sensor data. Commonly, the heuristic methods work based on both the overall shape of the object and its features. For more details on grasp synthesis, we refer the reader to the surveys of J. Bohg et al. [20] and Sahbani [21]. Similar to our grasping approach, Ciocarlie et al. [1] and Stuckler et al. [27] have considered grasps on objects either from above or from the side based on the overall shape of the object and the global characteristics such as center of mass and bounding box obtained from RGBD data. The intuition behind this approach is that many domestic objects are graspable by aligning the grippers with the (estimated) principal axes of the object. They follow a standard train and test procedure for object recognition, while our approach can incrementally update its knowledge based on new observations.

2.3. Object Perception and Learning

Interactive open-ended object category learning and recognition are key capabilities in assistive and service robotics. This means that a robot should be capable of continuously learning new objects in order to perform different tasks in domestic domains.

Aldoma et al. [28] reviewed properties, advantages and disadvantages of several state-of-the-art 3D shape descriptors available from the Point Cloud Library (PCL) to develop 3D object recognition and pose estimation system. They also proposed two pipelines for object recognition systems using local and global 3D shape descriptors from PCL. Martinez et al. [29] described a fast and scalable perception system for object recognition and pose estimation. The authors employed the RANSAC and Levenberg Marquardt algorithms to segment objects

and represented them based on SIFT descriptors. In [30], an object classification approach was proposed, in which the object representation was based on SIFT, SURF and color histograms. All these features were compacted into a histogram of visual words for optimizing the recognition process, as well as memory usage. In this case, authors used a naive Bayes classifier in the recognition stage. Yeh et al. [31] integrated the bag-of-words methodology to propose an efficient method for concurrent object localization and recognition. In most of the proposed systems described above, training and testing are separate processes, i.e., they do not occur simultaneously. However, in open-ended applications, data is continuously available and the target object categories are not known in advance. In these cases, traditional object recognition approaches are not well suited, because those systems are limited to using off-line data for training and are therefore unable to adapt to new environments / objects.

There are some approaches which support incremental learning of object categories. In these approaches, the set of classes is predefined and the models of known object categories are enhanced (e.g., augmented, improved) over time, while in open-ended approaches the set of categories is also continuously growing. Haibo et al. [10] proposed an incremental multiple-object recognition and localization (IMORL) framework using a multilayer perceptron (MLP) structure as the base learning model. The authors claimed that the proposed framework can incrementally learn from accumulated experiences and use such knowledge for object recognition. Yeh and Darrell [32] developed novel methods for efficient incremental learning of SVM-based visual category classifiers, and showed that, using their framework, it is possible to adapt the classifiers incrementally.

Kirstein et al. [33] proposed a lifelong learning approach for interactive lear-

ning of multiple categories based on vector quantization and a user interface. Collet et al. [34] proposed a graph-based approach for lifelong robotic object discovery. Similar to our approach, they used a set of constraints to explore the environment and to detect object candidates from raw RGB-D data streams. In contrast, their system does not interactively acquire more data to learn and recognize the object. Seabra Lopes and Chauhan [9] approached the problem of object experience gathering and category learning with a focus on open-ended learning and human-robot interaction. In their approach, learning is based on multiple representations as well as combinations of classifiers. They showed a system that starts with an empty vocabulary and can incrementally acquire object categories through the interaction with a human user. They used RGB data whereas we used depth data. Moreover, their object detection, learning and recognition approaches are completely different from our approach.

3. Overall System Architecture

The overall system architecture is depicted in Fig. 1. It is a reusable framework, with all modules developed in Robot Operating System (ROS)[35]. The current architecture is an evolution of the architecture developed in previous work for object perception and open-ended perceptual learning [36, 14]. Information exchange is performed using standard ROS mechanisms (i.e. either publish / subscribe or server/client). Therefore, any new module can be easily added to the system. The architecture includes two memory systems, namely the *Working Memory* and the *Perceptual Memory*. Both memory systems have been implemented

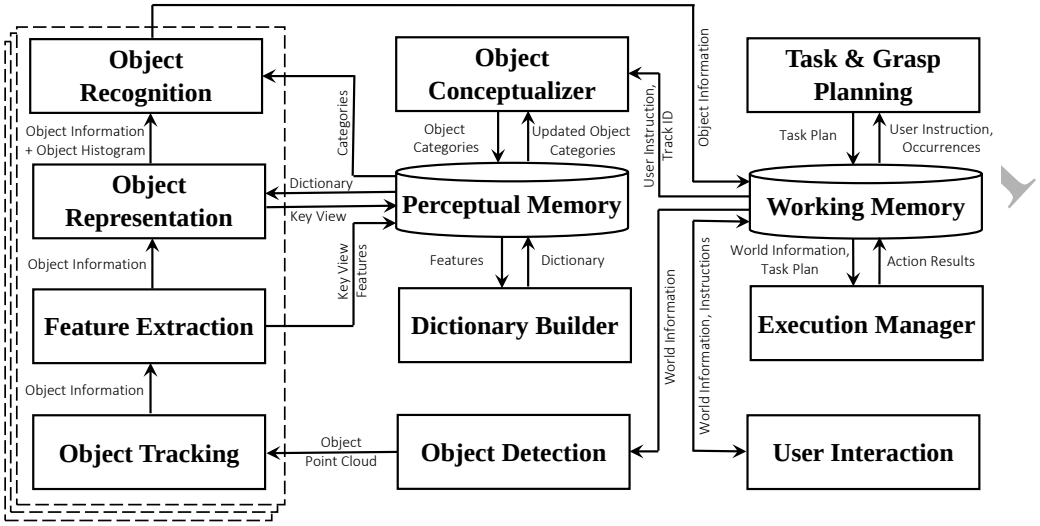


Figure 1: Overall architecture of the proposed system.

using a lightweight NoSQL database called LevelDB[1]. LevelDB is a fast key-value storage database that provides an ordered mapping from string keys to string values. The *Working Memory* is used for temporarily storing information as well as for communication among different modules. It keeps track of the evolution of both the internal state of the robot and the events observed in the environment (i.e. world model). The object features, dictionary of visual words, object representation data and object category models are stored into the *Perceptual Memory*. The goal of *Grasp Planning* is to extract a grasp pose (i.e. a gripper pose relative to the object) either from above or from the side of the object, using global characteristics of the object. The *Execution Manager* works based on a Finite-State-Machine (FSM) paradigm. It retrieves the task plan and the world model information from *Working Memory* and computes the next action (i.e. a primitive operator) based

[1]LevelDB has been developed by Google: https://code.google.com/p/leveldb/

on the current context. Then, it dispatches the action to the robot platform as well as records success or failure information in the *Working Memory*.

Whenever the robot captures a scene, the first step is preprocessing which includes three filtering procedures, namely distance filtering, a filter to remove the robot's body from sensor data, and a downsampling filter for reducing the size of the data. *Object Detection*, responsible for detecting objects in the scene, launches a new perception pipeline for each detected object. Each pipeline includes *Object Tracking*, *Feature Extraction*, *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 and color data [12]. The *Feature Extraction* module extracts features of the current object view and stores them in the *Perceptual Memory*. Based on the extracted features and on a visual dictionary, 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 [37]. *User Interaction* is essential for supervised experience gathering. A graphical user interface has been developed to teach the robot new object categories or to instruct the robot to perform a complex task.

The developed architecture, shown in Fig. 1, includes two perceptual learning modules. One of them, the *Dictionary Builder*, is concerned with building a dictionary of visual words for object representation. The dictionary plays a prominent role because it is used for category learning as well as recognition. The second learning module is the *Object Conceptualizer*. Whenever the instructor provides a category label for an object, the *Conceptualizer* retrieves the probabilistic models of the current object categories as well as the representation of the labeled object

in order to improve an existing object category model or to create a new category model. In recognition situations, a probabilistic classification rule is used to assign a category label to the detected object. The system is run in two stages. The first stage is dedicated to environment exploration. In this stage, unsupervised object discovery is carried out in the environment while the robot operates. The robot seeks to segment the world into "object" and "non-object". Afterwards, a pool of shape features is created by computing local shape features for the extracted objects. The pool of features is then clustered by the *Dictionary Builder* leading to a set of visual words (dictionary). Only the modules directly involved in object discovery and dictionary building are active in this stage. The second stage corresponds to the normal operation of the robot, with object category learning, recognition, planning and execution. In the following sections, the characteristics of each module are explained in detail.

4. Environment Exploration and Dictionary Construction

Comparing 3D objects by their local features would be computationally expensive. To address this problem, a Bag-of-Word (BoW) approach is adopted for object representation, i.e. objects are described by histograms of local shape features. This approach requires a dictionary of visual words. Usually, this dictionary is created off-line through clustering of a given training set. In open-ended learning scenarios, there is no predefined set of training data available at the beginning of the learning process. To cope with this limitation, we look at human cognition, in particular at the fact that human babies explore their environment in a playful (arbitrary) way [8]. Therefore, we propose that the robot freely explores several scenes and collects several object experiences. Gathering object experi-

Table 1: List of used constraints with a short description for each one.

Constraints	Description	Section
C_{table}: *"is this candidate on a table?"*	The target object candidate is placed on top of a table.	4
C_{track}: *"is this candidate being tracked?"*	Storing all object views while the object is static would lead to unnecessary accumulation of highly redundant data. This constraint is used to infer that the segmented object is already being tracked or not.	4
C_{size}: *"is this candidate manipulatable?"*	Reject large object candidate	5
$C_{\text{instructor}}$: *"is this candidate part of the instructor's body?*	Reject candidates that are belong to the user's body	4
C_{robot}: *"is this candidate part of the robot's body?"*	Reject candidates that are belong to the robot's body	4
C_{edge}: *"is this candidate near to the edge of the table?"*	Reject candidates that are near to the edge of the table	5
$C_{\text{key\_view}}$: *"is this candidate a key view?"*	For representing an object, only object views that are marked as key-views are stored in the database. An object view is selected as a key view whenever the tracking of an object is initialized, or when it becomes static again after being moved. In case the hands are detected near the object, storing key views is postponed until the hands are withdrawn.	5

ences by exploration has the advantage of not requiring any human annotation of individual objects. This (non goal-directed) exploration provides chances to discover new objects. In general, object exploration is a challenging task because of the dynamic nature of the world and ill-definition of the objects [34].

Since a system of boolean equations can represent any expression or any algorithm, it is particularly well suited for encoding the world and object candidates. Similar to Collet's work [34], we use boolean algebra[2], using three logical operators, namely AND ($\wedge$), OR ($\vee$) and NOT ($\neg$). A set of boolean constraints, C, was then defined based on which boolean expressions, ψ, were established to encode object candidates for the process of constructing the dictionary of visual words as well as for interactive object category learning and recognition. The definition of *"object"* in the exploration stage is more general than in the normal operation stage (see equations 1 and 4). In both cases, we assume that interesting objects are on tables and the robot seeks to detect tabletop objects (i.e. C_{table}). Due to memory size concerns, a representation of an object should only contain distinc-

[2]http://mathworld.wolfram.com/BooleanAlgebra.html

tive views. A view which is different from the current view may appear after the object is moved (i.e. the pose of the object relative to the sensor changes). An object view is selected as a key view (i.e. C_{key_view}) whenever the tracking of an object is initialized (C_{track}), or when it becomes static again after being moved. Therefore, the C_{key_view} constraint is used to optimize memory usage and computation while keeping potentially relevant and distinctive information. Moreover, $C_{\text{instructor}}$ and C_{robot} are used to filter out object candidates which are part of the instructor's body or robot's body. Accordingly, the resulting object candidates are less noisy and include only data corresponding to the environment:

$$\psi_{\text{exploration}} = C_{\text{table}} \wedge C_{\text{track}} \wedge C_{\text{key\_view}} \wedge \neg (C_{\text{instructor}} \vee C_{\text{robot}}), \qquad (1)$$

In our current setup, a table is detected by finding the dominant plane in the point cloud. This is done using the RANSAC algorithm [38]. Extraction of polygonal prisms is used for collecting the points which lie directly above the table. Afterwards, an Euclidean Cluster Extraction[3] algorithm is used to segment each scene into individual clusters. Every cluster that satisfies the exploration expression, $\psi_{\text{exploration}}$, is selected. The output of object exploration is a 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. Subsequently, to construct a pool of features, spin-images[4] are computed for the selected points extracted from the pool of object candidates. We use a PCL function to compute spin-images[5]. These

[3] http://www.pointclouds.org/documentation/tutorials/cluster_extraction.php

[4] The default spin-image parameters are the following: $SL = 5mm$, $A = \pi/2$ and $IW = 4$.

[5] In this work, we computed around 32000 spin-images from the point cloud of the 194 objects.

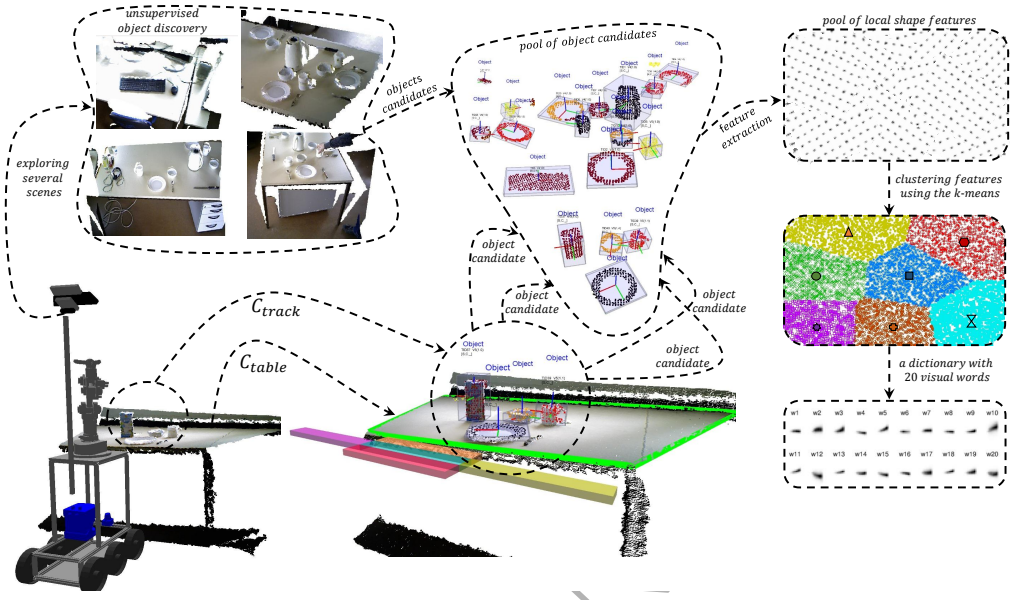


Figure 2: Dictionary construction: (*left*) the robot moves through an office to extract tabletop objects; (*center*) the captured scenes are processed to produce a pool of object candidates; (*right*) a pool of local shape features is obtained by computing spin-images from the pool of object candidates; the dictionary is subsequently constructed by clustering the features using the k-means algorithm; finally, a dictionary with 20 visual words is built.

capabilities are implemented in the *Object Detection*, *Object Representation* and *Feature Extraction* modules (see fig.1). Finally, the dictionary is constructed by clustering the features using the k-means algorithm [39]. The centers of the N generated clusters are treated as visual words, $\mathbf{w}_i$ ($1 \leq i \leq N$). Figure 2 shows a dictionary containing 20 words. In the implementation, we tested different dictionary sizes (see section 8.1). In the context of the RACE project [16], the University of Osnabruck provided us with a rosbag collected by one of their robots while exploring an office environment. A video of this exploration is available at:

http://youtu.be/MwX3J6aoAX0. The exploration stage was run on this rosbag.

5. Object Detection and Representation

This section presents the *Object Detection*, *Feature Extraction* and *Object Representation* modules as they are used in the normal operation stage.

5.1. Object Detection and Tracking

A common way for fast processing of massive point clouds is to use some mechanisms for removing unnecessary or irrelevant data. For this purpose, two filters are used that discard large quantities of 3D points from the original point cloud. The first step is to define a cubic volume in 3D (distance filtering), which defines the region of interest. The second filter reduces the spatial resolution of points (downsampling) using a voxelized grid approach[6]. Furthermore, the points corresponding to the body of the robot are filtered out from the original point cloud by retrieving the knowledge of the positions of the arm joints relative to the camera pose from the working memory.

After preprocessing, the next step is to find objects in the scene using the preprocessed point cloud. 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*

[6]http://pointclouds.org/documentation/tutorials/voxel_grid.php

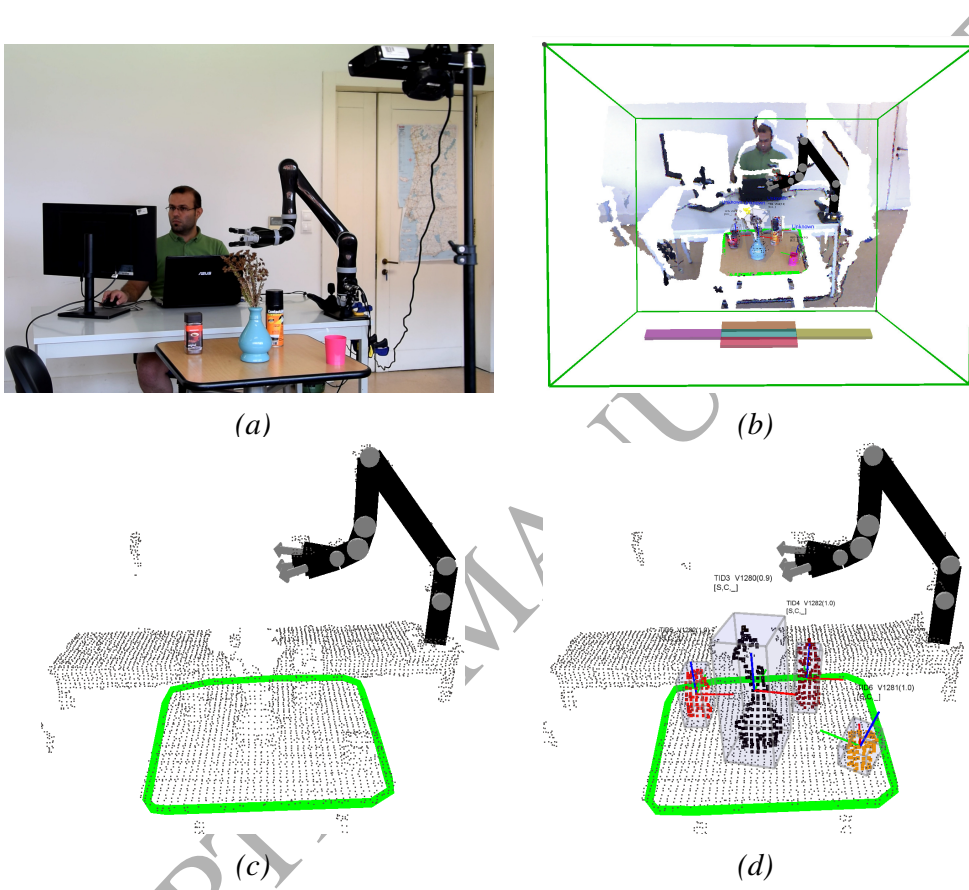


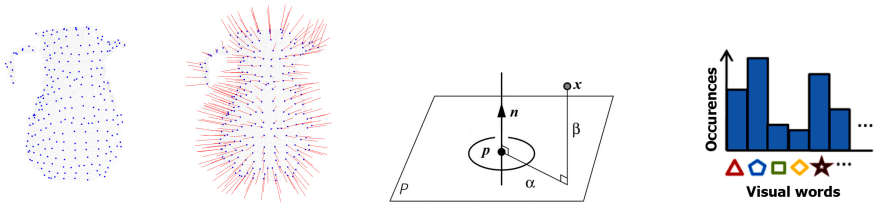
Figure 3: An example of preprocessing and *Object Detection*: *(a)* experiment setup; the JACO robotic arm performs manipulation tasks to clear the table; *(b)* distance filtering; *(c)* result of the second preprocessing step and table detection; *(d)* the position of the arm joints are used to filter out the points corresponding to robot's body from the original point cloud. The object candidates are shown by different bounding boxes and colors. The red, green and blue lines represent the local reference frame of the objects.

Detection module then assigns a new TrackID to each newly detected object and launches an object perception pipeline for the object. Finally, the object detection module pushes the segmented object candidate into the respective pipeline for subsequent processing steps. An example of the proposed detection approach is shown in Fig. 3.

The *Object Tracking* module is responsible for keeping track of the target object over time while it remains visible. It receives the point cloud of the detected object and computes an oriented bounding box aligned with the point cloud's principal axes. The center of the bounding box is considered as the pose of the object. The module sends out the tracked object information to the *Feature Extraction* module.

5.2. Feature Extraction and Object Representation

Object representation is critical to any object recognition system. In the present work, we adopt an approach to object representation in which object views (instances) are described by histograms of frequencies of visual words. The input is the set of features of an object candidate, **O**, computed by the *Feature Extraction* module. The *Feature Extraction* module involves keypoint extraction and computation of a spin image for each keypoint. Finally, the *Object Representation* module represents these features as a histogram of visual words. For keypoint extraction, first a voxelized grid approach is used to obtain a smaller set of points. The nearest neighbor point to each voxel center is selected as a keypoint[11]. 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 spin image is assigned to a visual word. Finally, each object is represented as a histogram of occurrences of visual words:



(a) *(b)* *(c)* *(d)*

Figure 4: Object representation for a flask: *(a)* keypoint extraction; *(b)* surface normal estimation for the keypoints; *(c)* a schematic of how spin-image is computed for a keypoint p; *(d)* histogram of visual words that represents the object view.

$$\mathbf{h} = [h_1\ h_2\ \ldots\ h_n], \tag{3}$$

where the i^{th} element of $\mathbf{h}$ is the count of the number of features assigned to a visual word, $\mathbf{w}_i$ and n is the size of the dictionary. Figure 4 illustrates the *Feature Extraction* and *Object Representation* processes for an object. The obtained histogram is dispatched to the *Object Recognition* module and is recorded in *Perceptual Memory*. To optimize the *Perceptual Memory*, some object views are marked as key views and only these are recorded into the memory. Key object views are selected by the *Object Tracking* module when the object is not moving and the user's hands are far away from the object [37]. In other words, key views are defined as follows:

$$\begin{aligned}\psi_{\text{key\_view}} = C_{\text{table}} \wedge C_{\text{track}} \wedge C_{\text{size}} \wedge C_{\text{key\_view}} \wedge \\ \neg\,(C_{\text{instructor}} \vee C_{\text{robot}} \vee C_{\text{edge}}),\end{aligned} \tag{4}$$

6. Interactive Object Category Learning and Recognition

The key idea for fast 3D object recognition is to use mechanisms for representing objects in a uniform and compact format. Estimating a robust model for each object category is more promising than template matching. In this section, first, a user interface for supervised experience gathering is presented. The interface is used not only for teaching new object categories in situations where the robot encounters with new objects but also for providing corrective feedback in the case there is a misclassification. The Bag-of-Words representation combined with the Naive Bayes approach are used to incrementally learn probabilistic models of object categories.

6.1. User Interaction

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 category learning and recognition system will be more flexible if it is able to learn new categories using the feedback of a human user. The *User Interaction* module provides a graphical menu to facilitate the collection of supervised object experiences and to instruct the robot to perform a task. In the case of supervised object experiences, two alternative interactions with an instructor are supported: gesture recognition or the usage of a graphical menu interface. In the first case, the instructor points to an object and then selects the desired label from a menu. In the second case the instructor can select the category label for an object based on its *TrackID*. Further details on supervised object experience gathering are available in [37]. An example of object labelling is depicted in Fig.5. The instructor puts a *'Vase'* on the table. Tracking is initialized with TrackID 1. The gray

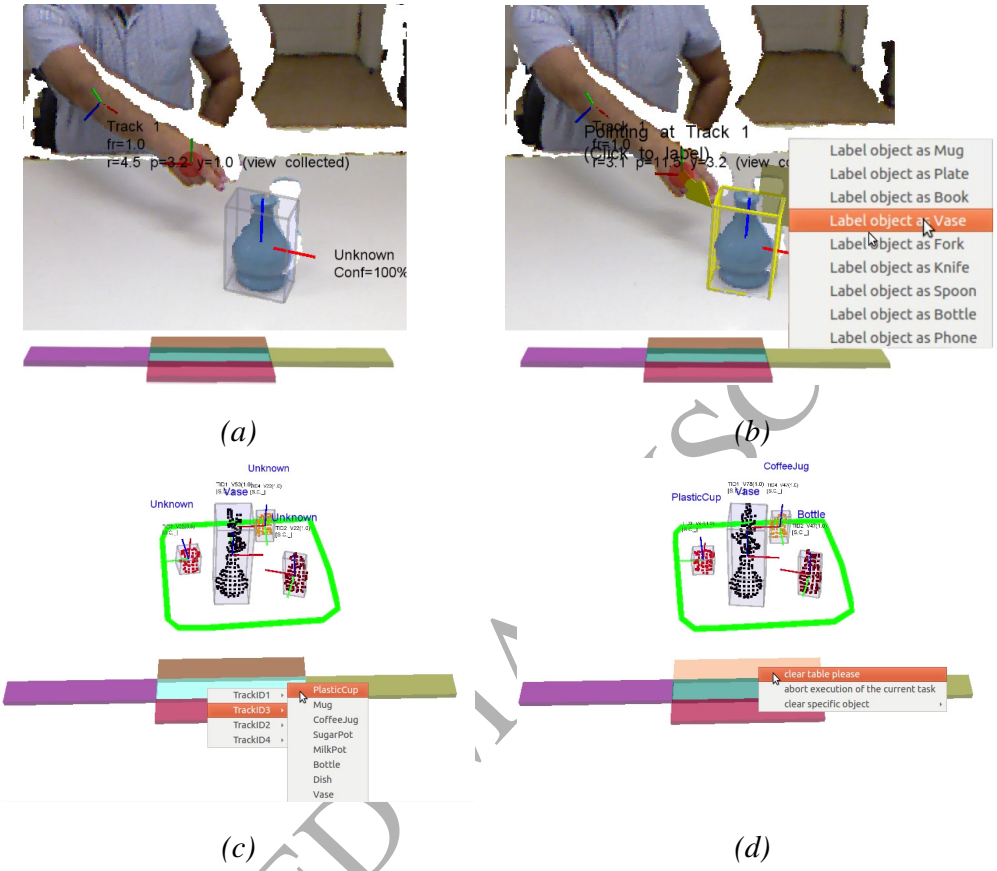


Figure 5: A 3D visualization of an object labelling event: (*a*) pointing to object by the instructor; (*b*) associating a label to the object that is currently being pointed; (*c*) labelling object categories by associating a label to a TrackID; (*d*) instructing the robot to perform the *clear_table* task;

bounding box signals the pose of the object as estimated by the tracker. TrackID 1 is classified as *'Unknown'* because vases are not yet known to the system; the instructor points at TrackID 1. The system recognizes the pointing gesture and the corresponding menu is activated. The instructor labels the object as *'Vase'*. The *Object Conceptualizer* (category learning) module is activated when the instructor provides a category label for the object. In addition, the *User Interaction* module

provides a menu to request the robot to perform a task or to abort the current task.

6.2. Object Conceptualizer

Learning methods used in most of the classical object recognition systems are not designed for open-ended domain, since those methods do not support an incremental update of the internal robot's knowledge based on new experiences. On the contrary, open-ended learning approaches can incrementally update the acquired knowledge (category models) and extend the set of categories over time, which is suitable for real-world scenarios. 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. In this section, we propose an open-ended 3D object category learning approach, which considers category learning as a process of updating a probabilistic model for each object category using the Naive Bayes approach. There are two reasons why Bayesian learning is useful for open-ended learning. One of them is the computational efficiency of the Naive Bayes approach. In fact, this model can be easily updated when new information is available, rather than retrained from scratch. Second, instance-based open-ended systems have continuously growing memory since they are constantly storing new object view representations (instances). Therefore, these systems must resort to experience management methodologies to discard some instances and thus prevent the accumulation of a too large set of experiences. In Bayesian learning, new experiences are used to update category models and then the experiences are forgotten immediately. The category model encodes the information collected so far. Therefore, this approach con-

sumes a much smaller amount of memory when compared to any instance-based approach. 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 $P(C_k)$. It should be noted that the parameters of the likelihood are the probabilities of each visual word given the object category $P(\mathbf{w}_t|C)$. In this work, we consider the probability of each visual word occurring in the object independently, regardless of any possible correlations with the other visual words (Naive Bayes approach). 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:

$$\begin{aligned} P(C_k|\mathbf{w}_1, \ldots, \mathbf{w}_n) &\propto P(C_k, \mathbf{w}_1, \ldots, \mathbf{w}_n) \\ &\propto p(C_k)\ P(\mathbf{w}_1|C_k)\ P(\mathbf{w}_2|C_k) \cdots\ P(\mathbf{w}_n|C_k) \\ &\propto P(C_k)\ \prod_{i=1}^{n} P(\mathbf{w}_i|C_k), \end{aligned} \tag{5}$$

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.

$$P(\mathbf{w}_i|C_k) = \frac{s_{ik}\ + 1}{\sum_{j=1}^{n} (s_{jk} + 1)}, \tag{6}$$

where s_{ik} is the number of times that word $\mathbf{w}_i$ was seen in objects from category C_k. Note, the probabilities are estimated with Laplace smoothing, by adding one to every counter, in order to prevent $P(w_i|C_k) = 0$. On each newly seen object of this category with x_i features of type $\mathbf{w}_i$, the following update is carried out:

$$s_{ik} \leftarrow s_{ik} + x_i, \tag{7}$$

The prior probability of category C_k is estimated as follows:

$$P(C_k) = \frac{N_k}{N}, \tag{8}$$

where N is the total number of seen objects of all categories and N_k is the number of seen objects from category k.

6.3. Object Category Recognition

The last step in object perception is object category recognition. To classify an object $\mathbf{O}$, which is represented as a histogram of occurrences of visual words $\mathbf{h} = [h_1\ h_2\ ...\ h_n]$, the posterior probability for each object category is approximated using the Bayes theorem as:

$$P(C_k|\mathbf{O}) = P(C_k|\mathbf{h}) = \frac{P(\mathbf{h}|C_k)P(C_k)}{P(\mathbf{h})} \approx P(\mathbf{h}|C_k)P(C_k), \tag{9}$$

Because the denominator does not depend on C_k, and the values of the features are given as a histogram of occurrences of visual words, the denominator is constant. Equation 9 is re-expressed based on equation 5 and multinomial distribution assumption:

$$P(\mathbf{h}|C_k)P(C_k) \approx P(C_k)\prod_{i=1}^{n} P(\mathbf{w}_i|C_k)^{h_i}, \tag{10}$$

In addition, to avoid underflow problems, the logarithm of the likelihood is computed:

$$\approx \log P(C_k) + \sum_{i=1}^{n} h_i\ \log P(\mathbf{w}_i|C_k), \tag{11}$$

The category of the target object $\mathbf{O}$ is the one with highest likelihood:

$$Category(\mathbf{O}) = \underset{C_k \in C}{\operatorname{argmax}}\ P(C_k|\mathbf{O}). \tag{12}$$

7. Planning and Execution

Figure 6 shows a schematic representation of the planning and execution framework. In this framework, task planning is triggered when a user instructs the robot to achieve a task (e.x. *clear_table*). This is handled by the *User Interaction* module. The current state of the system, including world model information, global characteristics of the object of interest (i.e. overall shape, main axis, center of bounding box) and robot pose is retrieved from the working memory. Then, a task plan would be generated. A plan is a sequence of primitive operators to be performed to achieve the given goal. It should be noted that *Task Planning* is not in the scope of this paper. Previously, we showed how to conceptualize successfully executed task plans and how to use these conceptualized experiences for task planning [18]. In the present work, a predefined task plan is used. In order to be executed, a task plan must be complemented with end-effector poses. A pose is represented as a tuple $G = (x\ ,\ y\ ,\ z\ ,\ roll\ ,\ pitch\ ,\ yaw)$, specified relative to the base reference frame of the robot.

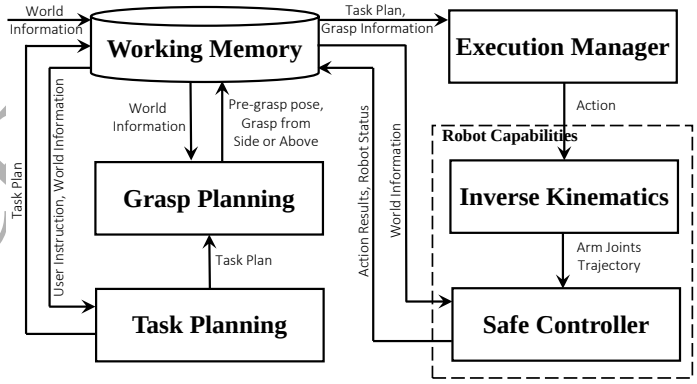
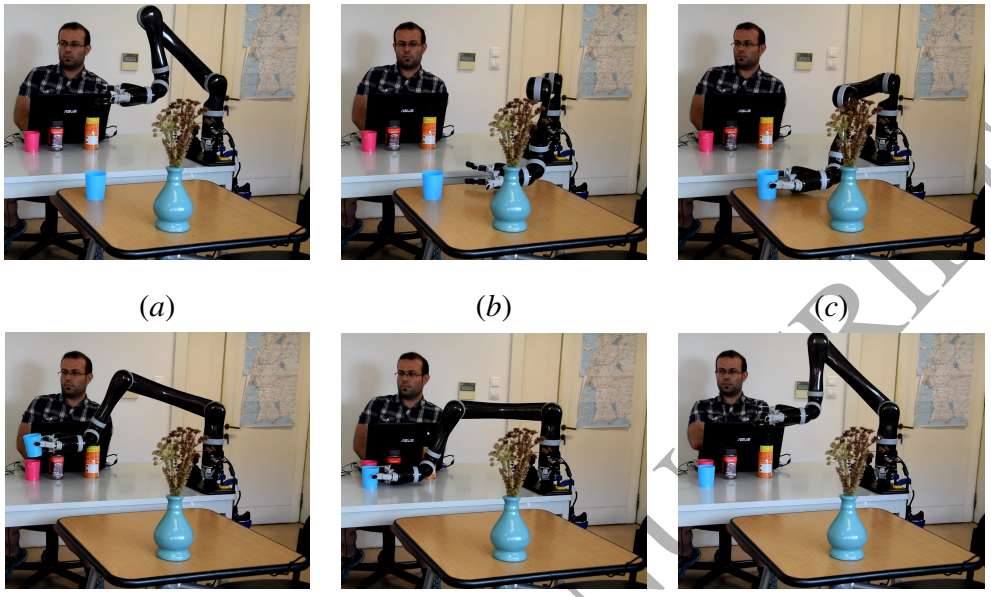


Figure 6: Schematic representation of task planning, grasp planning and execution manager.

The *Grasp Planning* module receives the task plan and chooses a grasp point either from above or from the side as well as a pre-grasp pose using the world model information and global characteristics of the object. In the current setup, the pre-grasp pose is placed at a fixed distance ($d_{pre-grasp} = 0.15\ m$) behind or above the center of bounding box of the object. The intuition behind this assumption is that many domestic objects are graspable by aligning grippers with the principal axes of the object [1][27]. In another work, we proposed an advanced grasping approach to learn how to grasp familiar objects using interactive object view labeling and kinesthetic grasp teaching [26]. Afterwards, the *Execution Manager* retrieves the plan and grasp information from the *Working Memory*. The *Execution Manager* uses a Fine State Machine to reactively execute the plan. The actions are dispatched to the *Robot Capabilities* module. Inverse kinematics and safe controller, integrated from the JACO arm driver[7], are used to transform a given end-effector pose goal into joint-space goals.

Whenever the object is grasped, the height of the robot's end-effector relative to the robot's base is recorded into *Working Memory* and it is used as the desired height for placing the grasped object. The *Execution Manager* computes a new trajectory to navigate the robot's end-effector to the placing area and sends out the action. After executing each action, the current state of the robot is updated in the *Working Memory*. Since world model information is updated by different modules (i.e. *Object Detection*, *Execution Manager* and etc.), the *Execution Manager* can abort execution when an unpredictable situation happens along expected execution path such as new obstacles move into the planned path of the robot arm. It

[7]http://wiki.ros.org/JACO.



(*a*) (*b*) (*c*)

(*d*) (*e*) (*f*)

Figure 7: Sequence of snapshots showing the JACO robotic arm performing a constrained pick and place task to clean the table; In this task, the orientation of the grasped object must be kept consistent throughout the plan; (*a*) the JACO robotic arm goes to the initial pose and extracts object (i.e. '*PlasticCup*') pose and shape properties; (*b*) a side grasp is selected and the robot goes to pre-grasp position; (*c*) the robot approaches and grasps the *PlasticCup*; (*d*) picking up the *PlasticCup* and moving it to the side; (*e*) placing the object and (*f*) going back to the initial position.

should be noted that an orientation constraint on the end-effector is used to grasp and move an object parallel to the support plane. In addition, objects outside of the arm's workspace are not considered. Figure 7 illustrates the result of a constrained pick and place plan executed on the robot.

Table 2: Average object recognition performance 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 Accuracy	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. Experimental Results

Three types of experiments were performed to evaluate the proposed approach. First, an off-line quantitative evaluation for the object recognition system is presented (section 8.1). Second, in section 8.2, a *"simulated teacher"* was developed to assess the performance and scalability of the proposed object perception system. Finally, a qualitative analysis of the complete interactive open-ended object recognition system is shown in the context of a real-life use case (section 8.3). In this case, a seven-minute demonstration session is described, where a user interacts with the system by teaching several objects to the robot and instructing the robot to perform a *"clear_table"* task.

8.1. Off-Line Evaluation of the Perceptual Learning Approach

An object dataset has been acquired for off-line evaluations, which contains 339 views of 10 categories of objects [11]. The system has four different parameters that must be tunned to provide a good balance between recognition performance, memory usage and computation time. To examine the accuracy of different configurations of the proposed approach, 10-fold cross validation was carried out. A total of 120 experiments were performed for different values of the four system parameters 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 2. The object recognition performance for each system configuration is depicted in figure 8 where the system parameters are represented as a tuple (VS, DS, IW, SL).

The parameters that obtained the best average accuracy were selected as the default system parameters. They are the following: VS = 0.01, DS = 90, IW = 4 and SL = 0.05. The accuracy of the system with the default parameters was 79 percent. Results show that the overall performance of the recognition system is promising. Spin images are capable of collecting distinctive traits of the local

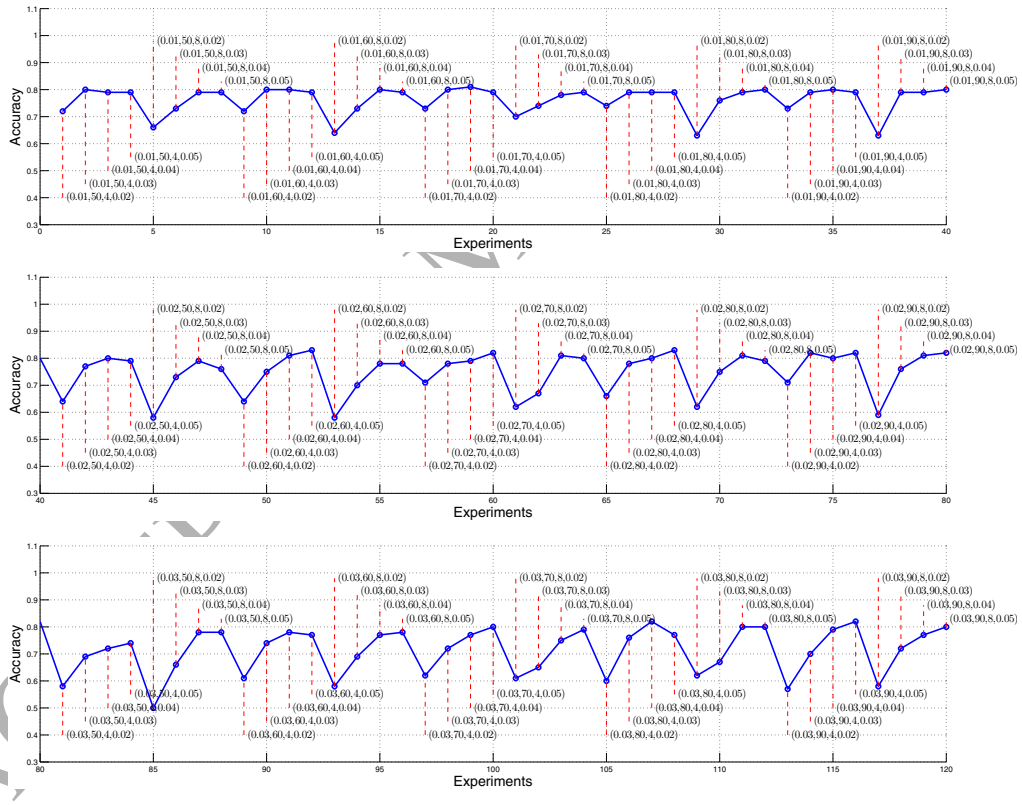


Figure 8: Object recognition performance for different values of four parameters of the system; the system parameters are represented as a tuple (VS, DS, IW, SL).

surface patches of each object. The results presented in sections 8.3 and 8.2 are computed using this configuration.

8.2. Open-Ended Evaluation

The off-line evaluation methodologies are not well suited to evaluate open-ended learning systems, because they do not abide to the simultaneous nature of learning and recognition and also those methodologies imply that the set of categories must be predefined. Therefore, an open-ended teaching protocol [9] [40] [11] is adopted in this evaluation. A simulated teacher was developed to assess the performance and scalability of the proposed object perception system by following the teaching protocol.

The *simulated teacher* autonomously interacts with the learning system using *teach*, *ask* and *correct* actions. 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 for checking whether the system can recognize them. The simulated teacher also provides corrective feedback in case of misclassification. Experiments were run on the largest publicly available 3D object dataset namely Washington RGB-D Object Dataset consisting of 250,000 views of 300 common household objects [41]. In the experiments that will be presented, the system begins with zero knowledge and the training instances become gradually available according to the teaching protocol. Therefore, the system learns new object categories as well as incrementally updates the existing object category models. Average Protocol Accuracy (*APA*) is computed using a sliding window of size $3n$, where n is the number of categories that have already been introduced. If the number of iterations k, since the last time a new category was introduced, is less than $3n$, all results are used. *APA* is used to determine if

a new category can be taught. According to the protocol, the system is ready to learn a new object category when *APA* is higher than a certain threshold (marked by the horizontal line in fig 9), and at least one instance of every known category has been tested ($k \geq n$). When an experiment is carried out, learning performance is evaluated using several measures, including:

- The number of learned categories at the end of an experiment (*LC*), an indicator of **How much does it learn?**;
- 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; i.e. **How fast does it learn?**
- Global classification accuracy (*GCA*), computed using all predictions in a complete experiment, and the Average Protocol Accuracy (*APA*), indicators of **How well does it learn?**.

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. Figure 9 (*top*) shows the performance of the system in the initial 200 iterations of the first experiment. The introduced categories are signaled by vertical red lines and category labels in the plot.

In the additional nine experiments, these categories were used again with different introduction sequences, the results of which are reported in Table 3. By comparing all experiments, it is visible that 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

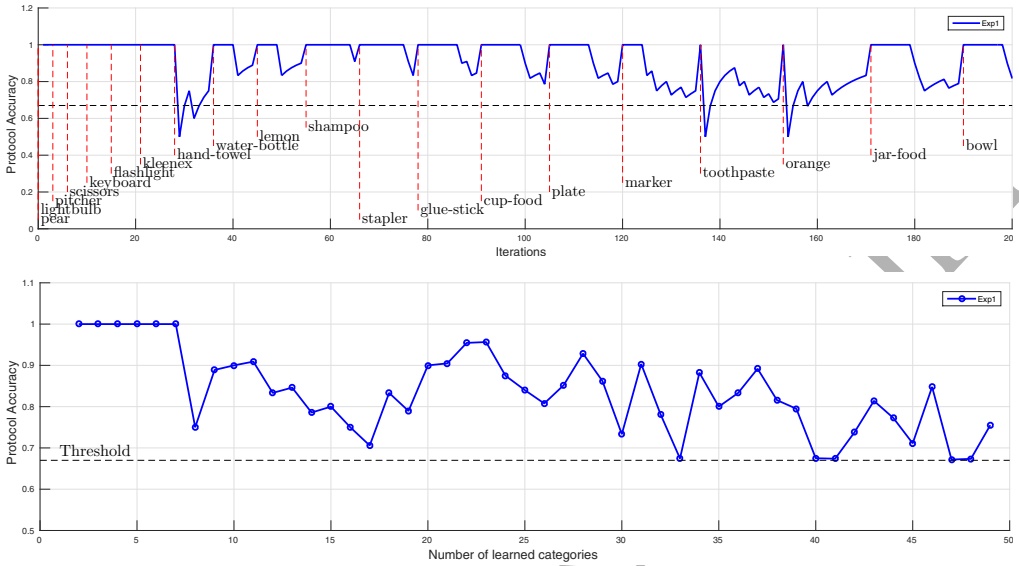


Figure 9: (*top*) Evolution of teaching protocol accuracy versus number of question/correction iterations in the first 200 iterations of the simulated teacher experiment 1 with the protocol accuracy threshold set to 0.67; (*bottom*) protocol accuracy versus the number of learned categories, for the same experiment.

other experiments. The underlying reason for different performances of these experiments is that categories were introduced to the system in a different order, which has a significant influence on the evolution of the learning performance.

Figure 9 (*top*) and Fig. 10 show the evolution of the teaching protocol accuracy in experiments 1, 3, 5, 7 and 9. Figure 9 (*bottom*) shows the protocol accuracy as a function of the number of learned categories. Figure 11 (*left*) shows the global classification accuracy (i.e. the accuracy since the beginning of the experiment) 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

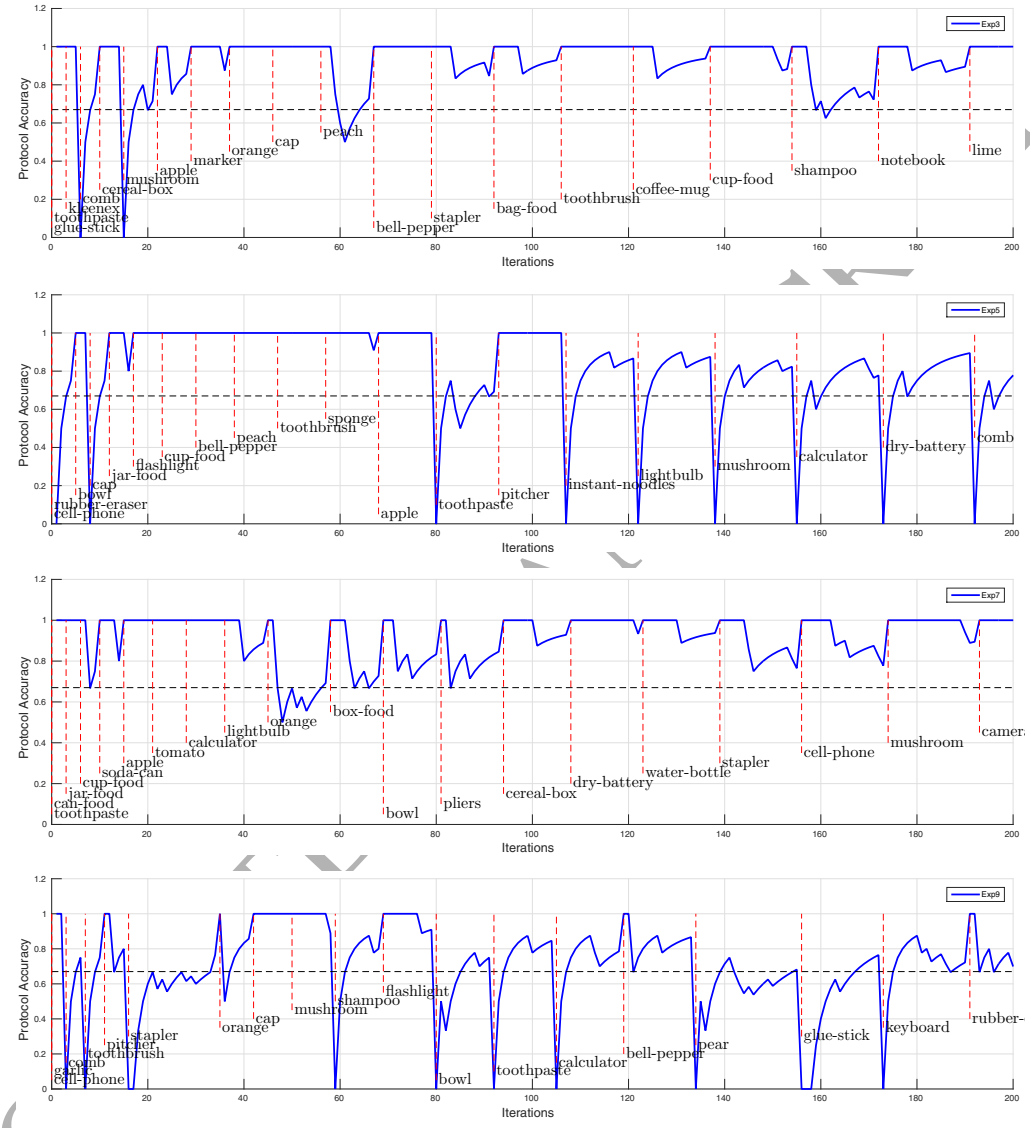


Figure 10: Evolution of teaching protocol accuracy versus number of question/correction iterations in simulated teacher experiments #3, 5, 7 and 9 with the protocol accuracy threshold set to 0.67.

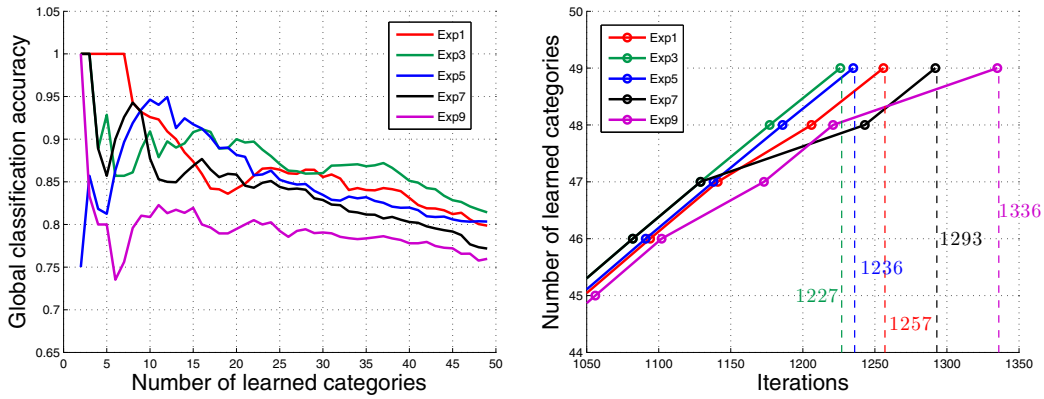


Figure 11: System performance during simulated user experiments: (left) global accuracy versus number of learned categories, a measure of how well the system learns; (right) number of learned categories versus number of question/correction iterations, represents *how fast* the system learned object categories.

classification task more difficult. To cope with this issue, memory management mechanisms [42], including salience and forgetting, can be considered. Finally, Fig. 11 (*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.

8.3. A Real Life Use-Case: Clear Table

In this section, we present and discuss a "*Clear Table*" use-case to show all the functionalities of the system. In this use-case, the system works in a scenario where a table is in front of the robot, and a user interacts with the system. In this task, the robot must be able to detect and recognize different objects and transport all objects except decorative table-top objects (e.g., *Vase*) to predefined areas.

The experimental setup is shown in Fig. 12. It consists of a computer for

Table 3: Summary of experiments[(1)].

EXP#	#QCI	#LC	#AIC	GCA (%)	APA (%)
1	1257	49	8.16	79	83
2	1228	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

[(1)] EXP#: experiment number; QCI: Question/Correction Iterations; LC: Learned Categories; AIC: Average Instances per Category; GCA: Global Classification Accuracy; APA: Average Protocol Accuracy.

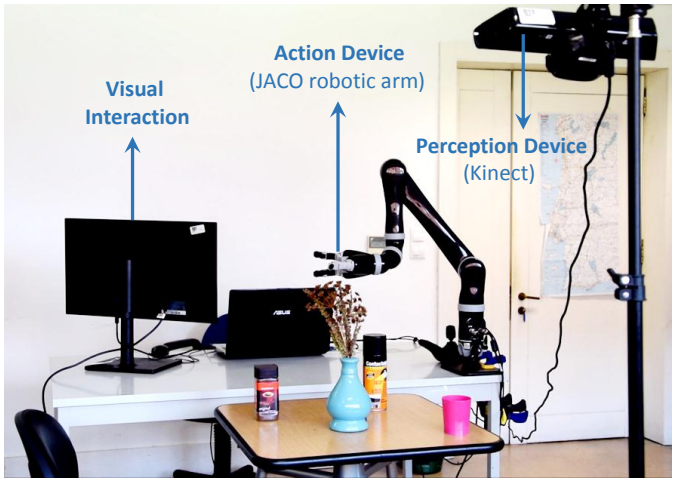


Figure 12: Our experimental setup consists of a computer for human-robot interaction purposes, a Kinect sensor and a JACO robotic-arm as the primary sensory-motor embodiments for perceiving and acting upon its environment.

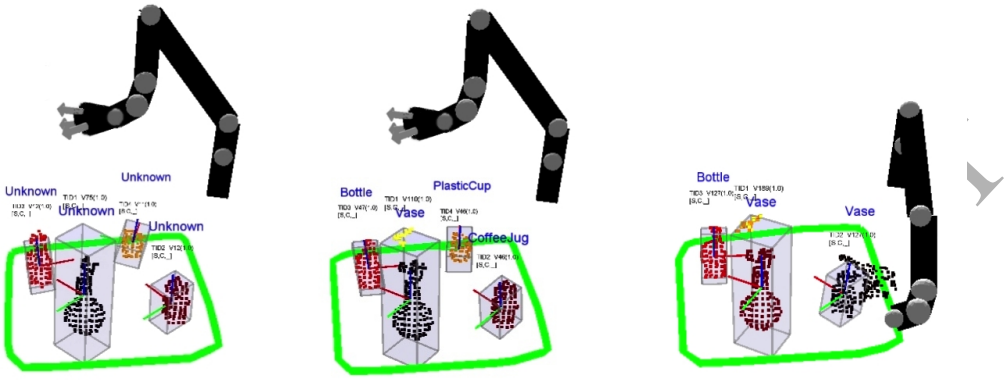


Figure 13: System performance during the clear_table use-case; (*left*): Initially, the system starts with no knowledge of any object. The position of the arm joints are retrieved from *Working Memory* and visualized by grey spheres and black lines. The table is then detected as shown by the green rectangle. Afterwards, the object candidates are detected and highlighted by different colors. The grey bounding boxes and the local reference frames represent the pose of the objects as estimated by object tracking module. (*center*): A user then teaches all the active objects to the system and all objects are correctly recognized, i.e., the output of object recognition is shown in blue on top of each object. (*right*): When grasping and manipulating an object, the shape of the object is partially changed and, as a consequence, a misclassification might happen.

human-robot interactions, a Kinect sensor for perceiving users and environment and a JACO robotic arm. The JACO arm has six degrees of freedom and a three fingers gripper. Since the JACO arm can carry up to 1.5kg[8], it is ideal for manipulating everyday objects. Moreover, infinite rotation around the wrist joints allows for flexible and effective interaction in a domestic environment.

At the beginning of the session, there is a *Vase* object on top of the table. Later, a user places three more objects including *Bottle*, *CoffeeJug* and *PlasticCup* on

[8]http://www.kinovarobotics.com

the table. Note that, at the start of the experiment, the set of categories known to the system is empty and therefore, the system recognizes all table-top objects as *Unknown* (see Fig.13 *left*). Afterwards, the user labels TrackID1 as a *Vase*. The system conceptualizes the *Vase* category and the category of TrackID1 is correctly recognized. Similarly, the user teaches all the other objects to the robot by pro-

Figure 14: The sequence of snapshots showing the JACO robotic arm performing a clear_table task; (*First row*): *PlasticCup* is the closest object to the arm's base. Therefore, the robot picks it up first from the table, transports it into the first predefined area and then, places the *PlasticCup* down. (*Second row*): *CoffeeJug* is selected as the second closest object. The robot goes to the pre-grasp area and then grasps the *CoffeeJug*. The robot moves the object into the second placing area and places it down. (*Third row*): Similarly, *Bottle* object is picked-up, moved and placed.

viding the respective category labels. As depicted in Fig.13 (*center*), the system could recognize all objects properly. Afterwards, the user instructs the robot to perform a *clear_table* task (i.e. puts the table back into a clear state). While there are active objects on the table, the robot retrieves the world model information from the *Working Memory*, including label and position of all active objects. The robot then selects the object closer to the arm's base and clears it from the table (see figure 14). As it is shown in the Fig.13 (*right*), whenever the robot grasps an object, the shape of the object is partially changed and therefore a misclassification might happen. This real life use-case shows that the developed system is capable of detecting new objects, tracking and recognizing them, as well as manipulating objects in various positions. In other words, it shows the important role of robust object recognition and manipulation in performing tasks in human environments. Moreover, it shows how human-robot interaction is currently supported. A video of this session is available online at: https://youtu.be/cTK10iNyYXg.

9. Conclusions

In this paper, we have presented a cognitive architecture designed to support a tight coupling between perception and manipulation for assistive robots. In particular, an interactive open-ended learning approach for grounding 3D object categories has been presented, which enables robots to adapt to different environments and reason out how to behave in response to the request of a complex task such as *clear_table*.

Unsupervised object exploration is used to construct a feature dictionary based on which objects are represented and object categories are learned. A Bayesian approach to category learning is proposed. We have assumed that the set of object

categories to be learned is not known in advance and the training instances are extracted from actual experiences of a robot rather than being available at the beginning of the learning process.

The proposed approach starts with the construction of a local 3D shape dictionary (visual words); each object is represented as a histogram of visual words and then the system creates or updates the probabilistic object category models based on Bayesian learning. For recognition, a probabilistic classification rule was used to assign a category label to the detected object. Results showed that the system can incrementally learn new object categories and perform manipulation tasks in reasonable time and appropriate manner. 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. In the continuation of this work, we are investigating the possibility of improving performance by topic modelling based on Latent Dirichlet Allocation (LDA) and also using other 3D shape descriptors (e.g. GOOD [43] and VFH [44]). Some results obtained with LDA have already been published [45]. Moreover, we would like to integrate compliance into the arm to provide comfortable interaction with the arm.

Acknowledgements

This work was funded by the EC 7th FP theme FP7-ICT-2011-7, grant agreement no. 287752 (project RACE - Robustness by Autonomous Competence Enhancement) and by National Funds through FCT project PEst-OE/EEI/UI0127/2014 and FCT scholarship SFRH/BD/94183/2013. We would like to thank the other RACE project partners for their efforts in the integration and the demonstrations,

and especially to the Knowledge-Based Systems Group, Institute of Computer Science, University of Osnabruck for providing the ROS bag used for dictionary building in the exploration stage.

References

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

[2] D.-J. Kim, R. Hazlett-Knudsen, H. Culver-Godfrey, G. Rucks, T. Cunningham, D. Portee, J. Bricout, Z. Wang, A. Behal, How autonomy impacts performance and satisfaction: Results from a study with spinal cord injured subjects using an assistive robot, Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 42 (1) (2012) 2–14.

[3] A. Jain, C. C. Kemp, El-e: an assistive mobile manipulator that autonomously fetches objects from flat surfaces, Autonomous Robots 28 (1) (2010) 45–64.

[4] C. Leroux, O. Lebec, M. B. Ghezala, Y. Mezouar, L. Devillers, C. Chastagnol, J.-C. Martin, V. Leynaert, C. Fattal, Armen: Assistive robotics to maintain elderly people in natural environment, IRBM 34 (2) (2013) 101–107.

[5] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mosenlechner, D. Pangercic, T. Ruhr, M. Tenorth, Robotic roommates making pancakes, in: Humanoid Robots (Humanoids), 2011 11th IEEE-RAS International Conference on, IEEE, 2011, pp. 529–536.

[6] N. Vahrenkamp, M. Do, T. Asfour, R. Dillmann, Integrated grasp and motion planning, in: Robotics and Automation (ICRA), 2010 IEEE International Conference on, IEEE, 2010, pp. 2883–2888.

[7] S. Jeong, M. Lee, Adaptive object recognition model using incremental feature representation and hierarchical classification, Neural Networks 25 (0) (2012) 130 – 140.

[8] L. Smith, M. Gasser, The development of embodied cognition: Six lessons from babies, Artificial life 11 (1-2) (2005) 13–29.

[9] A. Chauhan, L. Seabra Lopes, Using spoken words to guide open-ended category formation, Cognitive processing 12 (4) (2011) 341–354.

[10] H. He, S. Chen, Imorl: Incremental multiple-object recognition and localization, Neural Networks, IEEE Transactions on 19 (10) (2008) 1727–1738.

[11] S. H. Kasaei, M. Oliveira, G. H. Lim, L. Seabra Lopes, A. M. Tomé, Interactive open-ended learning for 3D object recognition: An approach and experiments, Journal of Intelligent & Robotic Systems 80 (3) (2015) 537–553.

[12] M. Oliveira, G. H. Lim, L. Seabra Lopes, H. Kasaei, A. Tomé, A. Chauhan, 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, pp. 2216–2223.

[13] M. Oliveira, L. Seabra Lopes, G. H. Lim, H. Kasaei, A. Sappa, A. Tomé,

Concurrent learning of visual codebooks and object categories in open-ended domains, in: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2015, pp. 2488–2495.

[14] M. Oliveira, L. Seabra Lopes, G. H. Lim, S. H. Kasaei, A. M. Tomé, A. Chauhan, 3D object perception and perceptual learning in the RACE project, Robotics and Autonomous Systems 75, Part B (2016) 614 – 626.

[15] S. Srinivasa, D. I. Ferguson, M. Vande Weghe, R. Diankov, D. Berenson, C. Helfrich, H. Strasdat, The robotic busboy: Steps towards developing a mobile robotic home assistant, in: International Conference on Intelligent Autonomous Systems, 2008, pp. 2155–2162.

[16] J. Hertzberg, J. Zhang, L. Zhang, S. Rockel, B. Neumann, J. Lehmann, K. Dubba, A. Cohn, A. Saffiotti, F. Pecora, M. Mansouri, Š. Konečný, M. Günther, S. Stock, L. Seabra Lopes, M. Oliveira, G. Lim, H. Kasaei, V. Mokhtari, L. Hotz, W. Bohlken, The race project, KI - Künstliche Intelligenz 28 (4) (2014) 297–304.

[17] S. Rockel, et al., An ontology-based multi-level robot architecture for learning from experiences, in: Designing Intelligent Robots: Reintegrating AI II, AAAI Spring Symposium, Stanford (USA), 2013, pp. 52–57.

[18] V. Mokhtari, L. Seabra Lopes, A. J. Pinho, Experience-based robot task learning and planning with goal inference, in: Twenty-Sixth International Conference on Automated Planning and Scheduling, 2016, pp. 509–517.

[19] S. S. Srinivasa, D. Ferguson, C. J. Helfrich, D. Berenson, A. Collet, R. Di-

ankov, G. Gallagher, G. Hollinger, J. Kuffner, M. V. Weghe, Herb: a home exploring robotic butler, Autonomous Robots 28 (1) (2010) 5–20.

[20] J. Bohg, A. Morales, T. Asfour, D. Kragic, Data-driven grasp synthesis-a survey, Robotics, IEEE Transactions on 30 (2) (2014) 289–309.

[21] A. Sahbani, S. El-Khoury, P. Bidaud, An overview of 3D object grasp synthesis algorithms, Robotics and Autonomous Systems 60 (3) (2012) 326–336.

[22] E. Chinellato, A. P. Del Pobil, The neuroscience of vision-based grasping: a functional review for computational modeling and bio-inspired robotics, Journal of integrative neuroscience 8 (02) (2009) 223–254.

[23] S. Monaco, A. Sedda, C. Cavina-Pratesi, J. C. Culham, Neural correlates of object size and object location during grasping actions, European Journal of Neuroscience 41 (4) (2015) 454–465.

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

[25] J. C. Culham, C. Cavina-Pratesi, A. Singhal, The role of parietal cortex in visuomotor control: what have we learned from neuroimaging?, Neuropsychologia 44 (13) (2006) 2668–2684.

[26] N. Shafii, S. H. Kasaei, L. Seabra Lopes, Learning to grasp familiar objects using object view recognition and template matching, in: Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on, IEEE, 2016, pp. 2895–2900.

[27] J. Stückler, R. Steffens, D. Holz, S. Behnke, Efficient 3D object perception and grasp planning for mobile manipulation in domestic environments, Robotics and Autonomous Systems 61 (10) (2013) 1106–1115.

[28] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, M. Vincze, Point cloud library: Three-dimensional object recognition and 6 DoF pose estimation, IEEE Robotics & Automation Magazine 19 (3) (2012) 80–91.

[29] M. Martinez Torres, A. Collet Romea, S. Srinivasa, Moped: A scalable and low latency object recognition and pose estimation system, in: Robotics and Automation, (ICRA 2010) IEEE International Conference on, 2010, pp. 2043–2049.

[30] M. Islam, F. Jahan, J.-H. Min, J. hwan Baek, Object classification based on visual and extended features for video surveillance application, in: Control Conference (ASCC 2011), 8th Asian, 2011, pp. 1398–1401.

[31] T. Yeh, J. J. Lee, T. Darrell, Fast concurrent object localization and recognition, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, 2009, pp. 280–287.

[32] T. Yeh, T. Darrell, Dynamic visual category learning, in: Computer Vision and Pattern Recognition, (CVPR 2008). IEEE Conference on, 2008, pp. 1–8.

[33] S. Kirstein, H. Wersing, H.-M. Gross, E. Körner, A life-long learning vector quantization approach for interactive learning of multiple categories, Neural Networks 28 (2012) 90 – 105.

[34] A. Collet, B. Xiong, C. Gurau, M. Hebert, S. S. Srinivasa, Herbdisc: Towards lifelong robotic object discovery, The International Journal of Robotics Research 34 (1) (2015) 3–25.

[35] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, ROS: an open-source robot operating system, in: ICRA workshop on open source software, Vol. 3, 2009, pp. 5–11.

[36] S. Hamidreza Kasaei, M. Oliveira, G. H. Lim, L. Seabra Lopes, A. Tomé, An interactive open-ended learning approach for 3D object recognition, in: Autonomous Robot Systems and Competitions (ICARSC), 2014 IEEE International Conference on, 2014, pp. 47–52.

[37] G. H. Lim, M. Oliveira, V. Mokhtari, S. Hamidreza Kasaei, A. Chauhan, L. Seabra Lopes, A. Tomé, Interactive teaching and experience extraction for learning about objects and robot activities, in: Robot and Human Interactive Communication, 2014 RO-MAN: The 23rd IEEE International Symposium on, 2014, pp. 153–160.

[38] M. A. Fischler, R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM 24 (6) (1981) 381–395.

[39] J. A. Hartigan, M. A. Wong, Algorithm as 136: A k-means clustering algorithm, Applied statistics (1979) 100–108.

[40] L. Seabra Lopes, A. Chauhan, How many words can my robot learn?: An approach and experiments with one-class learning, Interaction Studies 8 (1) (2007) 53 – 81.

[41] K. Lai, L. Bo, X. Ren, D. Fox, A large-scale hierarchical multi-view rgb-d object dataset, in: Robotics and Automation (ICRA), 2011 IEEE International Conference on, 2011, pp. 1817–1824.

[42] A. D. Baddeley, Human memory: Theory and practice, Psychology Press, 1997.

[43] S. H. Kasaei, A. M. Tomé, L. Seabra Lopes, M. Oliveira, Good: A global orthographic object descriptor for 3d object recognition and manipulation, Pattern Recognition Letters 83 (2016) 312–320.

[44] R. B. Rusu, G. Bradski, R. Thibaux, J. Hsu, Fast 3d recognition and pose using the viewpoint feature histogram, in: Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on, IEEE, 2010, pp. 2155–2162.

[45] S. H. Kasaei, A. M. Tomé, L. Seabra Lopes, Hierarchical object representation for open-ended object category learning and recognition, in: Advances in Neural Information Processing Systems (NIPS), 2016, pp. 1948–1956.

Hamidreza Kasaei Hamidreza Kasaei is a Ph.D. student at the Uni- versity of Porto (MAP-i), Portugal. Currently, he is a researcher at the IEETA, University of Aveiro, Portugal, where he works on 3D object cate- gory learning and recognition in open-ended domains. His main research in- terests focus on the intersection of robotics, machine learning, and machine vision. He is interested in developing algorithms for an adaptive percep- tion system based on interactive environment exploration and open-ended learning, which enables robots to learn from past ex- periences and interact with human users. He investigates active perception, where robots use their mobility and manipulation capabilities not only to gain the most useful per- ceptual information to model the world, also to predict the next best view for improving object detection and manipulation performances.

Miguel Oliveira a received the Mechanical Engineering and M.Sc. in Mechanical Engineering degrees from the University of Aveiro, Portugal, in 2004 and 2007, where later in 2013 he obtained the Ph.D. in Mechanical Engineering specialization in Robotics, on the topic of autonomous driv- ing systems. Currently he is a researcher at the Institute of Electronics and Telematics Engineering of Aveiro, Portugal, where he works on visual object recognition in open-ended domains. His research interests include multimodal sensor fusion, computer vision and robotics.

Gi Hyun Lim received the B.S. degree in metallurgical engineering and the M.S. and Ph.D degrees in electronics and computer engineering from Hanyang University, Seoul, Korea, in 1997, 2007 and 2010, respectively. He is currently a post-doctoral researcher at the Institute of Electronics and Telematics Engineering of Aveiro, Portugal. His research interests lie in the area of intelligence and learning for robots, including perception and seman- tics.

Luís Seabra Lopes is Associate Professor of Informatics in the Depart- ment of Electronics, Telecommunications and Informatics of the University of Aveiro, Portugal. He received a PhD in Robotics and Integrated Manu- facturing from the New University of Lisbon, Portugal, in 1998. Lus Seabra Lopes has long standing interests in robot learning, cognitive robotic archi- tectures, and human-robot interaction.

Ana Maria Tomé is an Associate Professor of electrical engineering with the DETI/IEETA of the University of Aveiro. Her research interests include digital and statistical signal processing, independent component analysis, and blind source separation, as well as machine learning applications.