
Proceedings Paper:
Dubba, KSR, De Oliveira, MR, Lim, GH et al. (4 more authors) (2014) Grounding language in perception for scene conceptualization in autonomous robots. In: Qualitative Representations for Robots: Papers from the AAAI Spring Symposium. Technical report. AAAI 2014 Spring Symposia, 24-26 Mar 2014, Palo Alto, California. AI Access Foundation , 26 - 33. ISBN 9781577356462

Grounding Language in Perception for Scene Conceptualization in Autonomous Robots

Krishna S.R. Dubba
School of Computing, University of Leeds, Leeds, UK

Miguel R. de Oliveira and **Gi Hyun Lim** and **Hamidreza Kasaei** and **Luís Seabra Lopes** and **Ana Tomé**
Instituto de Engenharia Electrónica e Telemática de Aveiro, Universidade de Aveiro, Portugal

Anthony G. Cohn
School of Computing, University of Leeds, Leeds, UK

Abstract

In order to behave autonomously, it is desirable for robots to have the ability to use human supervision and learn from different input sources (perception, gestures, verbal and textual descriptions etc). In many machine learning tasks, the supervision is directed specifically towards machines and hence is straight forward clearly annotated examples. But this is not always very practical and recently it was found that the most preferred interface to robots is natural language. Also the supervision might only be available in a rather indirect form, which may be vague and incomplete. This is frequently the case when humans teach other humans since they may assume a particular context and existing world knowledge. We explore this idea here in the setting of conceptualizing objects and scene layouts. Initially the robot undergoes training from a human in recognizing some objects in the world and armed with this acquired knowledge it sets out in the world to explore and learn more higher level concepts like static scene layouts and environment activities. Here it has to exploit its learned knowledge and ground language into perception to use inputs from different sources that might have overlapping as well as novel information. When exploring, we assume that the robot is given visual input, without explicit type labels for objects, and also that it has access to more or less generic linguistic descriptions of scene layout. Thus our task here is to learn the spatial structure of a scene layout and simultaneously visual object models it was not trained on. In this paper, we present a cognitive architecture and learning framework for robot learning through natural human supervision and using multiple input sources by grounding language in perception.

Introduction

One of the approaches to make autonomous systems and robots more robust is to impart them the ability to learn in open-ended environments, where they interact with non-expert users through multiple input sources. But the usual approach in machine learning as well as computer vision for building knowledge is using supervision in the form of labelled (annotated) examples where the examples come from a single source. This approach is not convenient if a robot should follow a life-long learning strategy as providing labelled examples is not natural to humans. Also the robot will come across different sources of input in several forms such as videos and images (perception), verbal, gesture and text (direct or indirect human supervision) etc. and combining these inputs to form a coherent source for learning is a challenge for robots. In order to make robot learning more robust and ubiquitous, we require the robot to cope with natural human supervision (i.e. through gestures and natural language description) (Scopelliti, Giuliani, and Fornara 2005) and multiple sources of input for learning. In this paper, we present a cognitive architecture and learning framework that enable robots to adapt to novel environments. These robotic systems must consist of many tightly coupled modules and handle large size data.

We motivate this approach using a scenario from the restaurant domain where the robot is an apprentice. The robot is trained on several objects to recognize them in a natural scene environment and then instructed to approach several tables and learn the table-top layouts on those tables. The teacher gives a description of the layouts of the cutlery in natural language and the layouts have several objects that the robot has not seen before during training. Then, the robot is instructed to approach another table and check if the table-top layout on that table is correct and also recognize the never-seen-before objects during training. In this scenario, the robot is trained to recognize objects by pointing at the objects and naming them which is how humans normally teach others when introducing new objects. Also in table-layout analysis the robot has to analyse input from different sources, i.e. from perception by looking at the table and language (verbal/text description of scene by teacher).

We approach this problem by converting data from different sources into relational format using spatial and temporal relations and then converting this data into graphs. Information from different sources can be compared and contrasted using graph similarities to learn relational knowledge and models. We present the related work and our approach to learning object models and cutlery layout models from incomplete, vague but related sources. Additionally, to support these functionalities, we propose a cognitive architecture, which comprise perception, object conceptualization, scene layout learning, user interface, multiple memory system and so on.

Related Work

As robots are expected to increasingly interact and collaborate closely with humans, robotics researchers need to look at human cognition as a source of inspiration for developing robot capabilities that simplify and enhance this interaction and make robots more human-friendly. Such an ability allows the robot to adapt to new environments, improve their knowledge with accumulated experiences and conceptualize new discovered categories. Learning is closely related to memory in human cognition. In the cognitive science literature, the existence of multiple memory systems is widely accepted. Recent literature converges on five major memory systems (Tulving 1991): *Procedural memory*, for sensory-motor skills; *Perceptual representation memory*, mainly for the identification of objects; *Working memory*, to support basic cognitive activity; *Semantic memory*, mainly for spatial and relational information; and *Episodic memory*, for specific past happenings, enabling "mental time travel".

In robotics, learning about spatial layouts of scenes clearly relates to human semantic memory and learning the time-independent features of objects can be related to perceptual representation memory. The architecture presented in this paper consists of multiple memory systems, including *Semantic memory* and *Perceptual representation memory*. In support of computational efficiency, the architecture supports runtime multiplexing mechanism of object perception pipelines, one for each tracked object. The architecture can also be configured to launch modules as independent processes or as threads in a single process.

There have been attempts to combine language and perception for smooth interaction of robots with humans. For example, the Grounded Situation Model (GSM) (Mavridis and Roy 2006) was proposed as a modular architecture for combining language, visual and proprioceptive inputs but conceptualization using past experiences is missing in the proposed work. Instead it 'remembers' past which is available for resolving temporal issues. Learning a joint model for language and perception has been attempted (Matuszek et al. 2012) for the object selection task by inducing weighted groups of words where the weights correspond to the attribute classifiers the words are paired with. Note that this approach only handles attribute based concepts such as 'yellow' etc. and cannot handle relations.

In (Seabra Lopes and Chauhan 2007), an open-ended object category learning system is described based on one-class learning (with support vector data descriptions) and human-robot interaction. The authors also proposed a teaching protocol for performance evaluation in open-ended learning. In (Seabra Lopes and Chauhan 2008), a system with similar goals is described. In this case, the learning approach is based on multiple representations, classifiers and classifier combinations, and a meta-learning component monitors classifier performance, reconfigures classifier combinations and chooses the classifier to be used for prediction. This is related to *attentional selection*, a biological mechanism for focusing on specific features or representations based on recent experience (Kruschke 2005). These works are based on 2D images collected in static scenes, when a target object is selected either by the user, or by the robot. Since there is no continuous stream of data being stored, memory requirements are easily satisfied. A life-long learning approach for interactive learning of multiple categories has been proposed (Kirstein et al. 2012) which involves selecting the most crucial features from a series of high dimensional feature vectors that almost exclusively belong to that specific category. This work showed successful learning of 5 color categories and 10 shape categories observed in 56 objects, just based on 2D images. However, the authors still follow a standard train-and-test procedure, which is not plausible in open-ended scenarios. Moreover, the authors did not provide details of their software architecture.

We use graph similarity for grounding language in perception. There has been significant work in graph similarity measures and the notion of graph similarity depends on the domain of interest. Graph similarity has been used in spatial-scene similarity queries (Nedas and Egenhofer 2008) where a spatial database and the query are converted to graphs and graph similarity is used to find the answer for the query. A generic approach for finding similarity of graphs based on node similarity was proposed in (Blondel et al. 2004). Heymans and Singh (Heymans and Singh 2003) also proposed a similarity measure between nodes of two graphs that are in turn used to compute the similarity measure between the graphs. They propose that not only the similarity between nodes but also dissimilarity between the nodes of graphs is important for computing the similarity measure for graphs. We extend this algorithm to graphs with labelled edges and discard certain terms in the similarity computation because they are no longer valid in our case.

Software Architecture

The software architecture developed for the project *Robustness by Autonomous Competence Enhancement* (RACE) is illustrated in Fig. 1. The functional components of the RACE architecture are represented by boxes in Fig. 1. Each component may contain one or more modules, which are implemented as *Robot Operating System* (ROS) nodes (Cousins et al. 2010). The *Reasoning and Interpretation* component includes a high-level interpreter, a temporal reasoner, a spatial reasoner and a description logic reasoner. *Perception* contains several modules for symbolic propioception and exteroception. *Experience Management* preprocesses occurrences produced by *Perception* and extracts relevant experiences, which are then used to create new concepts by the *Conceptualization*. New concepts are added to the *Semantic Memory* and stored in an OWL ontology format. *User Interface* sends instructions that generate a goal which is then relayed to *Planning*. Planning is carried out using *SHOP2*, a Hierarchical Task Network planner (Nau et al. 2003). It outputs a plan which is collected by the *Plan Execution and Management* component. Finally, actions are conveyed to the robot. These functional components were recently extended with new modules for object perception, perceptual memory, object conceptualization and human-robot interaction (Oliveira et al. 2014).

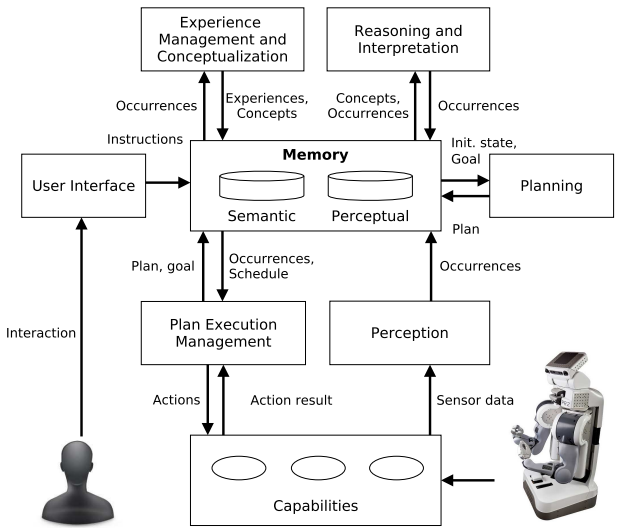


Figure 1: High level overview of the RACE architecture.

Multiple Memory Systems

The RACE software architecture accommodates two distinct memory systems: Semantic Memory and Perceptual Memory. The *Semantic Memory* (SMem) is implemented as a RDF database used as *blackboard* for all processes, the contents of which are similar to what can be found in an *ABox* in Description Logics. It is used for providing common knowledge to all high level processes. To cope with perception and data functionalities, an additional memory system,s employed, the Perceptual Memory. Unlike the SMem, which is specifically designed to store small size semantic data, the *Perceptual Memory* (PMem) database should be able to store large amounts of data and be very efficient in doing so. The implementation is carried out using a lightweight NoSQL database developed by Google called LevelDB [1]. LevelDB is a fast key-value storage database that provides an ordered mapping from string keys to string values. NoSQL databases are known for their simplicity, flexibility and scalability, which are key requisites for our *Perceptual Memory System* (Cattell 2011). In addition, LevelDB operates in memory and is copied to the file system asynchronously, which significantly improves its access speed. LevelDB allows for simultaneous access to the storage.

The multiple memory system implemented in RACE is very effective at storing and retrieving both semantic and perceptual data, especially, conceptualizing objects and scenes and grounding linguistic descriptions necessarily involves memory systems for semantic information and perceptual information. These memory systems support learning loops at multiple levels of abstraction from visual and text-based sources. In addition, there is strong evidence that human memory contains several memory systems (Tulving 1991).

[1] https://code.google.com/p/leveldb/

Perception, Learning and Interface Modules

The *Perception* functional component is composed of two software modules: *Detection* and *Multiplexed Object Perception*. The *Detection* module is responsible for detecting novel objects in the scene. This module periodically retrieves a list of all the objects currently on top of the table [2] (Rusu and Cousins 2011). Then it extracts from that list all objects that are not currently being tracked. For each of those, a *Multiplexed Object Perception* pipeline is launched. Figure 2 (*a*) shows an example where three objects are segmented, but only one is not currently being tracked, i.e., does not have a bounding box around it.

There is one *Multiplexed Object Perception* pipeline running for each object being tracked. Each pipeline provides tracking, feature extraction and object recognition functionalities. Tracking is based on a particle filter based approach [3] that uses both geometric as well as color information. Periodically, features are extracted from the tracked object and then classified by the object recognition module. To provide a 3D object description, we use regional surface descriptors as proposed in (Johnson and Hebert 1999).

The *Conceptualization* is triggered when a user provides a category label for an object. This operation is handled by the *User Interface* module, as will be discussed next. When triggered, *Object Conceptualizer* (OC) reads the current object categories from the PMem, as well as a set of features describing the labelled object. With this information, a new set of object categories is produced and written back to the PMem.

The *User Interface* functional component is responsible for the interaction with the user. The user skeleton is tracked in real-time and pointing gestures detected by computing the direction of the forearm. When a pointing direction intersects a tracked object bounding box, it is assumed that the user is pointing to that object. To produce an object category label (which is sent to the *Conceptualization*), the pointing detection must occur simultaneously with the object labelling. In this way, the simplicity of the pointing detection mechanism does not cause any false positive detections. The object labelling mechanisms are implemented as rviz interactive menu markers [4]. An example is shown in Fig. 2 (*c*).

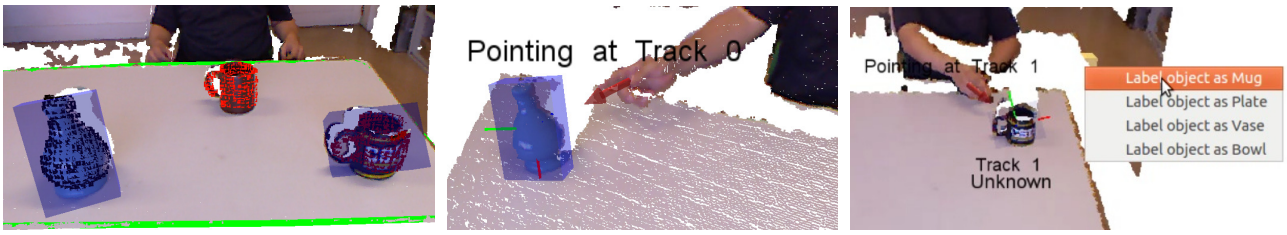
Object Conceptualization By Grounding Language In Perception

In order to adapt to new service environments, robots need to categorise physical objects and acquire vocabulary about them (Lim, Suh, and Suh 2011). We approach this problem from a long-term perspective and with emphasis on domain open-endedness, i.e. not assuming a pre-defined set of categories (Chauhan et al. 2013), (Kasaei et al. 2013). If the robot does not know what a 'mug' looks like, it may ask the user to point to one. Such situation provides an opportunity to collect a training instance (an experience) for object

[2] http://wiki.ros.org/tabletop_object_detector

[3] http://www.willowgarage.com/blog/2012/01/17/tracking-3d-objects-point-cloud-library

[4] http://wiki.ros.org/interactive_markers



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

Figure 2: Examples of some of the software modules: (*a*) Object Detection; (*c*) Gesture Recognizer; and (*d*) Interface Manager.

conceptualization. Concerning category formation, the baseline is a purely memory-based (instance-based) learning approach.

We adopt object-centered 3D shape features for object recognition. Firstly, a down-sampling filter based on octrees (Potmesil 1987), which keeps balance between computational efficiency and robustness, is applied to extract reliable feature keypoints from objects. A spin-image descriptor, which is rotation, scale and pose invariant in 3D perception (Johnson and Hebert 1999), is utilized to describe the surrounding shape in each keypoint.

In order to estimate the dissimilarity between a target object view, v^t, and an object view contained in a category model in the PMem, v^m, the following distance function is implemented:

$$D_V\left(v^t, v^m\right) = \frac{\sum_{l=1}^{q} \min_k d(v_l^t, v_k^m)}{q} \tag{1}$$

where v_l^t and v_k^m represents the spin-image of the target object and the model object, respectively. q is the number of spin-image of target object view. A kd-trees based fast approximate method is utilized to search nearest neighbour in the high-dimensional feature spaces (Bentley 1975). The minimum distance between the target object and the instances of a certain category, *C*, is considered as the object-category distance:

$$D_C\left(v^t, C\right) = \min_{v^C \in C} D_V\left(v^t, v^C\right) \tag{2}$$

where $v^C$are the view instances of object category *C*. In 3D object recognition, views in same category are more similar than those in others. Thus, $D_C()$ can be utilized to distinguish "unknown" from each "known" category with its normalized distance. Here the intra-category distance (ICD) of each object category is defined, as follows:

$$ICD(C) = \frac{\sum_{v_i \in C} \sum_{v_j \in C, v_i \neq v_j} D_V\left(v_i, v_j\right)}{n \cdot (n-1)} \tag{3}$$

where n is the number of category instances, and v_i and $v_j$are two different instances of category *C*. Finally, the normalized distance of object category, $ND()$, is computed based on dividing $D_C(v_t, C)$ by $ICD(C)$, as follows:

$$ND\left(v_t, C\right) = \frac{D_C(v_t, C)}{ICD(C)} \tag{4}$$

The normalized distance is utilized to classify target objects. If the value is larger than a given threshold for all categories, then the object is classified as "unknown". Otherwise it is classified as the category which gives the smallest value for ND.

Grounding Language In Perception With Graph Matching

In this section, we explain the grounding of language in perception using graph matching for scene layouts. We assume the robot has perceived the table layout and has a textual description of cutlery layout from a human. For our experiments, the textual description of laying out the cutlery and the perceived image of the final layout is taken from the *wikihow* website[5]. In order to concentrate on the main task of grounding, we hand annotated the objects in the image and the robot has no knowledge of the objects labels (ideally some of the object labels can be obtained from using the object conceptualization module explained in the previous section). The textual description of the cutlery layout is converted to relational form manually[6]; care was taken to preserve the vagueness and incomplete nature of human instructions. For example, in one of the scripts for laying the cutlery for a formal dinner, the instructions say that the dessert fork should be above the dinner plate and the dessert spoon is also above the dinner plate, but it is not mentioned whether the dessert spoon should be above the dessert fork or vice versa. To simplify the task, we ignored the orientation of the cutlery and alternative positions and other options mentioned in the instructions (these are addressed in future work). Also we assume that the robot knows the orientation of the scene i.e. the layout is seen and explained as if looking from the chair in front of the dinner layout.

Relational Graph From Perception A spatial relational description is obtained from the bounding boxes of the annotated objects in the image using the spatial relations in

[5]The idea of using knowledge from wikihow to guide learning follows on from this approach in the Robohow project: `http://robohow.eu/project/approach`

[6]We are currently developing a natural language parser to obtain the required relational format.

Fig. 4. Note that the textual description uses a subset of these relations to explain the layout. The relations are computed by projecting the bounding boxes to *X* and *Y* axes and then using the Rectangle Algebra (Balbiani, Condotta, and Del Cerro 1998) to obtain the relations.

Figure 3: An annotated table top. The bounding boxes for the annotated polygons are used for generating the spatial relations among the cutlery and crockery.

top, bottom, right, left, on, under,
top_right, top_left, bottom_left, bottom_right,

Figure 4: Spatial relations used for obtaining a relational graph from perception.

A graph is constructed from this spatial relational description of the layout by using each object as a vertex and the relation between a pair of objects as the label for the edge connecting the corresponding vertices. The relations used are not symmetric and hence the graph is a directed graph. Also note that every pair of objects has a spatial relation resulting in a complete graph where every vertex is connected to every other vertex in both directions. In this experiment, the robot does not know the labels of the annotated polygons, hence vertices are given unique codes which we try to replace with correct object names using the textual description.

Relational Graph From a Description The manually generated spatial relational description is used to generate the graph where each object becomes a vertex and the spatial relations are used to label the edges between vertices. Note that unlike the graph constructed from an image, this graph is not a complete graph as the instructions do not exhaustively list all the relations and it is not possible to infer them from available relations because of the inherent ambiguity in the relations possible. Relations that are possible to be inferred are derived such as: if A is to the left of B, B is to the left of C, then it can easily be inferred that A is to the left of C. For the the experiments, we used two scripts with instructions on how to set the table which we call *script1*[7] and *script2*[8]. Figure 3 corresponds to *script1* though we use the same image for *script2* to make the experiments more generic and interesting because some objects mentioned in the instructions from *script2* are not present in the image and also the number of objects mentioned in the textual description is different from the number of objects in the image. The relational description that is manually derived from *script1* is given in Fig. 7.

Similarity Scores Between Vertices In Graphs

Definition 1. *A directed graph G is a 6-tuple $G = (V, E, \lambda, \alpha, L_V, L_E)$, where V is the set of vertices (nodes), $E \subseteq V \times V$ is a set of edges, $\lambda : V \rightarrow L_V$ and $\alpha : E \rightarrow L_E$ are functions assigning labels to vertices and edges respectively.*

Definition 2. *Let G be a directed graph as defined above. Let I_v and O_v be the set of incoming and outgoing edges of a vertex v in G. We can define a labelled edge histogram for each vertex v denoted by H_v which is computed from I_v and O_v and each bucket is labelled by an edge label from the set L_E and whether it is an incoming or an outgoing edge. Let B_G be the set of buckets in H_v. Hence there are a total of $2 * |L_E|$ buckets in H_v. For example, $H_v(l_{i,1})$ and $H_v(l_{o,1})$ denote the number of incoming and outgoing edges for vertex v labelled as l_1 respectively.*

Let $G_1 = (V_1, E_1, \lambda_1, \alpha_1, L_{V_1}, L_{E_1})$, where $|V_1| = n_1$, and $G_2 = (V_2, E_2, \lambda_2, \alpha_2, L_{V_2}, L_{E_2})$, where $|V_2| = n_2$, be two directed graphs. We aim to obtain a n_1 x n_2 similarity matrix S, where the entry S(a, b) expresses the similarity between nodes a and b where $a \in V_1$ and $b \in V_2$. Since we will be dealing with two graphs from the same domain, we expect $L_{E_1} \equiv L_{E_2}$.

Two nodes from different graphs can be similar not only because of their properties alone but also because of their neighbourhood. We first define a similarity measure Sim that depends only on the properties of the individual nodes and subsequently the similarity measure of their neighbours is added. Sim is also an n_1 x n_2 matrix where the entry $Sim(a, b)$ expresses the similarity between nodes a and b where $a \in V_1$ and $b \in V_2$ computed solely from a and b without taking into account their neighbours. $Sim(a, b)$ captures the similarity by counting the types of incoming and outgoing edges for the two nodes as given below.

$$Sim(a, b) = \sum_{a \in V_1, b \in V_2, l \in B_{G_1} \cap B_{G_2}} min(H_a(l), H_b(l)) \tag{5}$$

The similarity score matrix S is first initialized with $S^1 = Sim$ and then iteratively updated (giving $S^2, S^3, ...$) using the following two scores which capture the similarity of the neighbourhood of two nodes. Equation 6 is based on the rule: the similarity of two nodes increases if they are referred

[7] http://www.wikihow.com/Set-a-Dinner-Table

[8] http://www.wikihow.com/Arrange-a-Place-Setting-for-a-Formal-Dinner

$$A_1^{(k)}(a,b) = \begin{cases} \sum\limits_{a_2 \to a, b_2 \to b, l \in L_{E_1} \cap L_{E_2}} \frac{\mathrm{S}^k(a_2,b_2)}{deg_{in}^l(a) deg_{in}^l(b)} & \text{if } deg_{in}^l(a) \neq 0 \text{ and } deg_{in}^l(b) \neq 0 \\ \sum\limits_{a_2 \in V_1, b_2 \in V_2} \frac{\mathrm{S}^k(a_2,b_2)}{n_1 \times n_2} & \text{if } deg_{in}^l(a) = deg_{in}^l(b) = 0 \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

$$A_2^{(k)}(a,b) = \begin{cases} \sum\limits_{a \to a_2, b \to b_2, l \in L_{E_1} \cap L_{E_2}} \frac{\mathrm{S}^k(a_2,b_2)}{deg_{out}^l(a) deg_{out}^l(b)} & \text{if } deg_{out}^{\alpha_i}(a) \neq 0 \text{ and } deg_{out}^{\alpha_i}(b) \neq 0 \\ \sum\limits_{a_2 \in V_1, b_2 \in V_2} \frac{\mathrm{S}^k(a_2,b_2)}{n_1 \times n_2} & \text{if } deg_{out}^l(a) = deg_{out}^l(b) = 0 \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

to (using incoming edges) by similar nodes by identically labelled edges and Eq. 7 is based on the rule: the similarity of two nodes increases if they refer (using outgoing edges) similar nodes by identically labelled edges. In the equations below, $a \to a_2$ means there is an edge with the given label from node a to node a_2 and $deg_{in}^l(a)$ and $deg_{out}^l(a)$ gives the number of incoming and outgoing edges of vertex a with label l.

The above scores are computed in an iterative fashion (starting with $k = 1$) and the similarity matrix is updated using the following equation.

$$\mathrm{S}^{(k+1)}(a,b) = \frac{A_1^{(k)}(a,b) + A_2^{(k)}(a,b)}{2} \text{ x } Sim(a,b) \quad (8)$$

Note that the similarity matrix has to be normalized (we use Frobenius norm) during every iteration and this is done as shown below:

$$\mathrm{S}^{(k+1)} \leftarrow \frac{\mathrm{S}^{(k+1)}}{\| \mathrm{S}^{(k+1)} \|_F} \quad \text{where } \| A \|_F = [\sum_{i,j} abs(a_{i,j})^2]^{1/2} \quad (9)$$

This iterative process is repeated until the similarity matrix stabilizes or until we exceed the specified number of iterations. When computing the similarity matrix is finished, the best matchings are obtained using the Hungarian algorithm (Kuhn 1955) on this similarity matrix.

There are other rules that are used by Heymans and Singh (Heymans and Singh 2003) but they are not relevant here noting the fact that the graphs in our domain are completely connected or are constructed using incomplete knowledge. Hence we cannot rule out the fact that there might be an edge between two vertices which is not present in our knowledge.

Experimental Results

This section presents a qualitative analysis of the object conceptualization module and results of graph matching for grounding language in perception for scene layouts. For object conceptualization, a typical case study is shown where the system acquires knowledge about the objects through the interaction with the user.

Scenario-Based Tests

To show the features and functioning of the complete object conceptualization system, a scenario-based test was made. The test consists of a sequence of about 1 minute long, where the user is moving several objects in front of the camera. During the experiment, the system must detect several objects and user pointing gestures that occur. It must also learn about new object categories when the user provides a category label. At the start of the experiment, the system had no prior knowledge, i.e., the PMem did not contain any data.

Figure 5 shows that the proposed architecture is capable of functioning as an open-ended system. Objects that were unknown at first were later labelled by the user. After that, the system is capable of properly recognizing the objects.

Graph Matching for Scene Layouts

Grounding language in perception is achieved through matching relational graphs obtained from textual description and the corresponding perceived scene. A similarity matrix is obtained as explained in the previous section and subsequently obtain the node matching by applying the Hungarian algorithm on this matrix. From this node matching, the system now has knowledge about objects such as how a dinner knife looks like etc. The system was able to correctly identify all the objects from the image using *script1* description (note that the image corresponds to *script1*). This was the case when the number of objects in the text description was equal to the number of objects in the image. Note that the graph matching need not be exact, it is possible that sometimes the instructions given to the robot are not accurate and might have missing objects or extra objects and in that case it will try to use a subgraph of the graph derived from text to match with the graph from the image. It will use the overall graph similarity measure to see which subgraph (i.e. combination of nodes) selected has a good

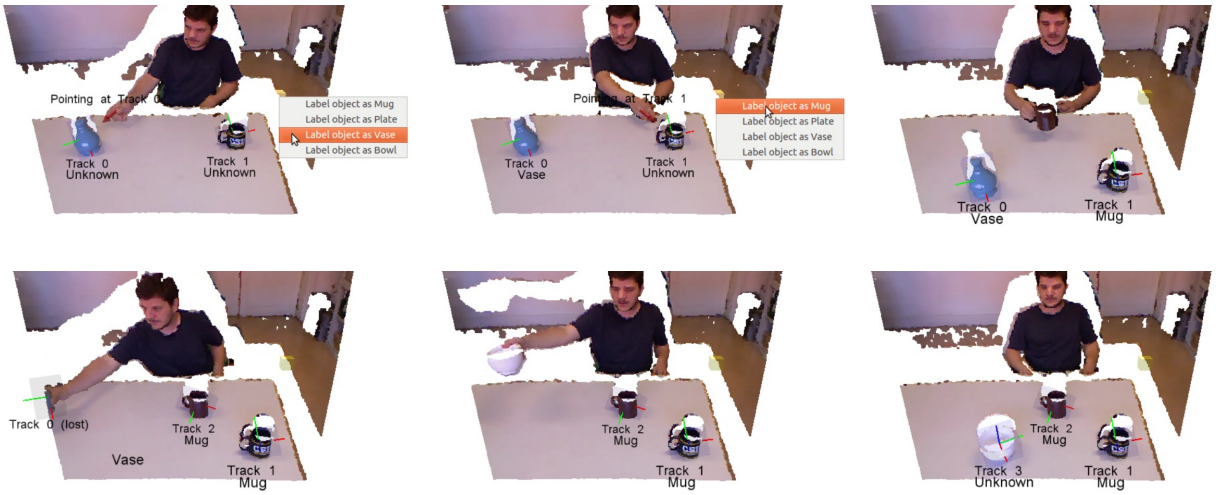


Figure 5: Key frames of the sequence presented in this case study. All images shown are automatically generated by the system.

match. We experimented to test if overall graph similarity measure is a good metric to maximize correct object matchings. Figure 6*(a)* shows different runs by choosing different subgraphs from the graph derived from text (each point in the plot corresponds to a subgraph) and the plot shows that the graph similarity measure is a good metric to maximize the object matchings i.e. the higher the graph similarity measure, the more the number of objects matched. In order to test the robustness of the approach, we also used another script (*script2*) to see if the robot can learn any objects from a script not specifically describing the image in question. In this experiment there were more objects in the graph derived from text (15) compared to 12 objects in the image, hence we did several runs by choosing different subgraphs with combinations of 12 objects from 15. The algorithm has to decide which subgraph to consider for finalizing the match and once again it is evident from the plot shown in Fig. 6*(b)* that the graph similarity measure is a good metric to decide which combination of objects to consider for best matching. Interestingly, it obtained better objects matchings and high graph similarity when it dropped objects that were not present in the image (such as wine glasses, champagne glass etc).

The results are significant in the contexts where the robot is asked to recognise place setting layouts and any mismatches. This could be done from hand coded layout knowledge or from learned models such as we acquire here.

Conclusion and Future Work

In this paper, we proposed a cognitive architecture and learning framework for robot learning through natural human supervision and using multiple input sources by grounding language in perception. The results obtained in the restaurant domain are encouraging though some of components (robust perception and natural language parser) need to be implemented for the whole system to be completely automatic. Future directions include using ego-centric videos and verbal description as input for robot learning. The spatial primitives used in the current framework are very simple (Rectangle Algebra) and there is scope to use a combination of spatial calculi for providing the description of scenes to the robot. Another aspect to investigate is mapping the grounding problem to the problem of merging and aligning ontologies (Noy and Musen 1999) where the information extracted from text description and perception can be considered as two different ontologies that need to be aligned.

right(water_glass, dinner_plate)
on(napkin, dinner_plate)
top(bread_plate, dinner_plate) _left
on(butter_knife, bread_plate)
left(salad_fork, dinner_plate)
left(dinner_fork, dinner_plate)
top(dessert_fork, dinner_plate)
top(dessert_spoon, dinner_plate)
right(dinner_knife, dinner_plate)
right(fish_knife, dinner_plate)
right(soup_spoon, fish_knife)

Figure 7: Spatial relations derived manually from *script1* that describes a place setting on a table.

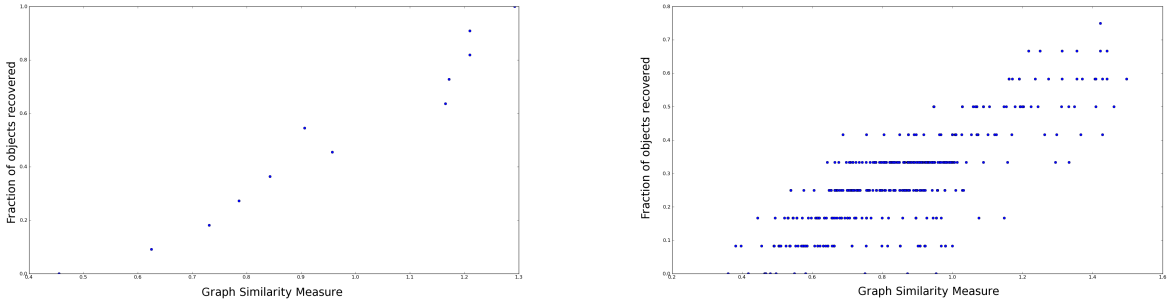
Acknowledgment

This work is supported by the EC Seventh Framework Program under Project RACE (Robustness by Autonomous Competence Enhancement, grant agreement no. 287752).

References

Balbiani, P.; Condotta, J. F.; and Del Cerro, L. F. 1998. A model for reasoning about bidimensional temporal relations. In *Principles of KRR*, 124–130.

Bentley, J. L. 1975. Multidimensional binary search trees



(*a*) (*b*)

Figure 6: Results comparing graph similarity measure and the fraction of the objects matched correctly between textual descriptions (*a - script1, b - script2*) and perception.

used for associative searching. *Communications of the ACM* 18(9):509–517.

Blondel, V. D.; Gajardo, A.; Heymans, M.; Senellart, P.; and Van Dooren, P. 2004. A measure of similarity between graph vertices: Applications to synonym extraction and web searching. *SIAM review* 46(4):647–666.

Cattell, R. 2011. Scalable SQL and NoSQL data stores. *SIGMOD Rec.* 39(4):12–27.

Chauhan, A.; Seabra Lopes, L.; Tomé, A. M.; and Pinho, A. 2013. Towards supervised acquisition of robot activity experiences: an ontology-based approach. In *16th Portuguese Conference on Artificial Intelligence - EPIA'2013*.

Cousins, S.; Gerkey, B.; Conley, K.; and Garage, W. 2010. Welcome to ROS topics [ROS Topics]. *Robotics Automation Magazine, IEEE* 17(1):12 –14.

Heymans, M., and Singh, A. K. 2003. Deriving phylogenetic trees from the similarity analysis of metabolic pathways. *Bioinformatics* 19(1):138–146.

Johnson, A., and Hebert, M. 1999. Using spin images for efficient object recognition in cluttered 3D scenes. *PAMI, IEEE Transactions on* 21(5):433–449.

Kasaei, S.; Seabra Lopes, L.; Tomé, A.; Chauhan, A.; and Mokhtari, V. 2013. An instance-based approach to 3D object recognition in open-ended robotic domains. In *Proceedings of the 4th Workshop on RGB-D: Advanced Reasoning with Depth Cameras, RSS'2013*.

Kirstein, S.; Wersing, H.; Gross, H.-M.; and Körner, E. 2012. A life-long learning vector quantization approach for interactive learning of multiple categories. *Neural Networks* 28:90–105.

Kruschke, J. K. 2005. Category learning. *The handbook of cognition* 183–201.

Kuhn, H. W. 1955. The Hungarian method for the assignment problem. *Naval research logistics quarterly* 2(1-2):83–97.

Lim, G. H.; Suh, I. H.; and Suh, H. 2011. Ontology-based unified robot knowledge for service robots in indoor environments. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 41(3):492–509.

Matuszek, C.; Fitzgerald, N.; Zettlemoyer, L.; Bo, L.; and Fox, D. 2012. A joint model of language and perception for grounded attribute learning. In *ICML*, 1671–1678.

Mavridis, N., and Roy, D. 2006. Grounded situation models for robots: Where words and percepts meet. In *Intelligent Robots and Systems, 2006 International Conference on*, 4690–4697.

Nau, D. S.; Au, T.-C.; Ilghami, O.; Kuter, U.; Murdock, J. W.; Wu, D.; and Yaman, F. 2003. SHOP2: An HTN planning system. *Journal of AI Research* 20:379–404.

Nedas, K. A., and Egenhofer, M. J. 2008. Spatial-scene similarity queries. *Transactions in GIS* 12(6):661–681.

Noy, N. F., and Musen, M. A. 1999. An algorithm for merging and aligning ontologies: Automation and tool support. In *Workshop on Ontology Management at AAAI-1999*.

Oliveira, M.; Lim, G. H.; Kasaei, H.; Chauhan, A.; Seabra Lopes, L.; and Tomé, A. M. 2014. A software architecture for open-ended object conceptualization from experiences. In *ICRA 2014*, submitted.

Potmesil, M. 1987. Generating octree models of 3D objects from their silhouettes in a sequence of images. *Computer Vision, Graphics, and Image Processing* 40(1):1–29.

Rusu, R., and Cousins, S. 2011. 3D is here: Point Cloud Library (PCL). In *ICRA, 2011*, 1–4.

Scopelliti, M.; Giuliani, M. V.; and Fornara, F. 2005. Robots in a domestic setting: a psychological approach. *Universal Access in the Information Society* 4(2):146–155.

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

Seabra Lopes, L., and Chauhan, A. 2008. Open-ended category learning for language acquisition. *Connection Science* 20(4):277–297.

Tulving, E. 1991. Concepts of human memory. In *Memory: Organization and locus of change*. Oxford Univ. Press. 3–32.