Ontology-based Knowledge Management System with Verbal Interaction and Concept Learning for Home Service Robots

(ホームサービスロボットのための
言語対話機能と概念学習機能を備えた
オントロジに基づく知識管理システム)

July, 2021

Doctor of Philosophy (Engineering)

Liliana Villamar Gómez

リリアナ ビヤマル ゴメス

Toyohashi University of Technology

DATE: 2021/07/09

Department of Computer Science and Engineering		ID	D165305
Name	LILIANA VILLAMAR GOMEZ		

Supervisor	Prof. JUN MIURA

ABSTRACT

Title	Ontology-based Knowledge Management System with Verbal Interaction and Concept Learning for Home Service Robots

Service robotics, which refers to assistant robots at home, has been gaining importance. Researchers are creating robots to accompany and fulfill the needs of older adults or disabled humans. Developing a service robot that can support people at home is still an open challenge that many researchers are targeting.

Service robots need skills and characteristics to be involved in the social environment of humans. Several factors, such as the interaction with humans, the response and speed, and the usefulness for their tasks, must be considered for service robots. Moreover, they need to learn and adapt to their particular house settings, make decisions, and behave accordingly.

The ultimate goal of this research is to create fully autonomous robots that help people at home in diverse ways. Therefore, it is necessary to provide the robot with different skills depending on the scenario facing, such as vision, language communication, or learning. While a service robot can be equipped with the general knowledge and skills to cope with the most common situations at home, it also might encounter new scenarios, and hence, it must know what to do in such a case. For a service robot, learning a new concept by itself is a crucial factor.

The contributions of this research are focused on the development of a system for service robots performing tasks at home and are divided into two main components. The first component is building an Ontology-based Knowledge Management System with Verbal Interaction for Command Interpretation and Execution by Home Service Robots.

We develop a system for service robots that combines ontological knowledge reasoning and human–robot interaction to interpret natural language commands and successfully perform household chores, such as finding and delivering objects. We use an ontology to represent the general information of the components in the environment and their relationships; moreover, the system links natural language commands, the ontology object representation, and the real objects' information. The robot disambiguates uncertain requests through spoken interaction with the human before completing a task. It utilizes information from the ontological knowledge to create more precise questions.

The second component is realizing Ontology Learning of New Concepts combining Textural Knowledge, Visual Analysis, and User Interaction for Service Robots Applications. On this part, the robot is provided with another essential feature to adapt inside a home environment. In this part, we focus on the learning of new ontological concepts oriented to service robot applications. We propose combining textural knowledge, visual analysis and user interaction to determine the correct placement of the new concept in the ontology structure. We aim to make the robot able to extend its ontological knowledge as needed. Moreover, the system's functionality and performance are demonstrated by experiments in a simulated environment.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Service Robots for Home Support

Nowadays, robots are taking greater importance in areas such as the hospitality, entertainment, and healthcare industry, where they must take on roles as caregivers and companions. Researchers are creating robots to accompany and fulfill the needs of older adults or disabled humans. These kinds of robots require special skills to operate in human environments in order to work, interact, and communicate with them, especially at home. Service robotics, which refers to assistant robots at home, has also been gaining interest. Developing a service robot that can support people at home is still an open challenge that many researchers are targeting.

Service robots need skills and characteristics to be involved in the social environment of humans. Several factors, such as the interaction with humans, the response and speed, and the usefulness for their tasks, must be considered for service robots. Moreover, they need to learn and adapt to their particular house settings, make decisions, and behave accordingly. A robot's behavior is crucial during human–robot contact; the person should feel safe, willing to be assisted, and rely on the robot, and the overall experience must be satisfactory [1]. An essential element needed to provide efficient service robots is knowledge and context reasoning, given their diverse and continuously changing environments.

Various researchers have studied how to provide the knowledge that robots need to complete different tasks [2], [3], [4], [5]. Some have attempted to make robots learn new concepts or assignments by themselves [6], [7], [8]. Reasoning based on acquired knowledge before taking action is a skill be pursued in robots [9], [10], [11]. Most of these published works focus on

specific and separate skills that service robots need. However, some of them lack well-ordered concepts about the environment, and the knowledge is limited to that acquired during the learning phase that is, the first human–robot interactions. In other cases, the questions' and answers' patterns are fixed and do not handle unexpected answers from users [3].

The implementation of natural language in robots has been explored at different levels of human–robot interaction. Models that can extract information from natural language instructions and their surrounding environments have been developed to improve robots' instruction understanding. NL-based probabilistic, cognitive, and logic models are used for plan generation. In contrast, theoretical knowledge grounding, knowledge gap detection, and gap-filling models have been developed for knowledge world mapping [12].

1.2 Research Goal and Contributions

The ultimate goal of this research is to create fully autonomous robots that help people at home in diverse ways. Therefore, it is necessary to provide the robot with different skills depending on the scenario facing, such as vision, language communication, or learning. While a service robot can be equipped with the general knowledge and skills to cope with the most common situations at home, it also might encounter new scenarios, and hence, it must know what to do in such a case. For a service robot, learning a new concept by itself is a crucial factor.

The contributions of this research are focused on the development of a system for service robots performing tasks at home and are divided into two main components. The first component is building an ontology-based knowledge management system with verbal interaction for command interpretation and execution by home service robots.

We develop a system for service robots that combines ontological knowledge reasoning and human–robot interaction to interpret natural language commands and successfully perform household chores, such as finding and delivering objects. We use an ontology to represent the general information of the components in the environment and their relationships; moreover, the system links natural language commands, the ontology object representation, and the information of the real objects involved. The robot disambiguates uncertain requests through spoken interaction with the human before completing a task. It utilizes information from the ontological knowledge to create more precise questions.

The second component is realizing ontology learning of new concepts combining textural knowledge, visual analysis, and user interaction for service robots applications. On this part, the

robot is provided with another essential feature to adapt inside a home environment. In this part, we focus on the learning of new ontological concepts oriented to service robot applications. We propose combining textural knowledge, visual analysis, and user interaction to determine the new concept's correct placement in the ontology structure. We aim to make the robot able to extend its ontological knowledge as needed.

The main contributions of this work are:

- The conjunction of knowledge reasoning and verbal interaction to interpret and disambiguate natural language commands.
- Knowledge management based on ontology with inference capabilities in a home environment.
- Question formulation incorporating ontological assertions.
- Textural knowledge acquisition for word meaning identification and image data collection.
- Image visual analysis and natural language interaction to concept description selection.
- Ontological knowledge update with the conceptualization of new objects.

Moreover, the system's functionality and performance are demonstrated by experiments in a simulated environment. The contributions are explained further in the following chapters.

1.3 Related Work

1.3.1 Ontology and Knowledge-based Systems

The first step in creating a service robot with the abovementioned characteristics is to provide it with knowledge not only for completing chores but also for understanding what is being asked, to manage unexpected situations or ambiguous assignments. Numerous systems have been developed to manage knowledge in different areas. For instance, in [13], the authors emphasized the importance of managing knowledge for software maintenance that is hard to track using conventional documentation methods. They proposed the use of ontologies to save such knowledge and described a methodology for developing such an ontology.

A model conception for enterprise knowledge management was proposed in [14]. The model includes a set of ontologies specializing in specific parts of the process of knowledge management, such as the knowledge acquisition, knowledge storage, and knowledge identification processes. Another system proposing a merged ontology was presented in [15], which includes medical, hotel, and city ontologies.

An application for knowledge management in mechatronics was developed in [16]; it contains retrieval information methods that humans and computers can understand. However, the knowledge is static and does not consider the dynamic management of the information, which is essential in a changing environment. The authors of [3] described an attempt to create cognitive maps through interactive dialog; new attributes and names of objects can be learned through the interaction considering the user preference toward an object. Nevertheless, the acquired knowledge is limited to the user's response and does not consider solving inconsistencies in knowledge.

In the area of robotics, different studies have proposed the use of ontologies as a standardization of knowledge representation. Ontologies have been proposed to describe robots, parts of robots, and their relationships [17]; products and their final assembled states [18]; and appliances, their moving parts, and their functionalities [19].

Some ontologies were designed to include concepts related to advanced driver-assistance systems, driving tasks, and driving distractions [20]. To eliminate the heterogeneity between devices and applications, some ontology models describe concepts on the basis of sensor-stimulus-observation design patterns [4].

However, the knowledge described in these ontologies needs to be complete, including all the parts involved and their functions, because inferences rely on the integrity and precision of the represented knowledge. Moreover, these approaches do not consider any mechanism besides the inference process for dealing with lack of information.

A more robust approach is KnowRob [21], a knowledge processing system designed to give entirely autonomous robots knowledge to accomplish manipulation tasks. This system includes knowledge representation, reasoning techniques, and methods of knowledge acquisition and exchange. Its applicability has been demonstrated through experiments of a robot making pancakes [22] and in projects such as openEASE [23].

Two important aspects are the use of WordNet to disambiguate vague natural language task descriptions [24] and the integration of robot control data into symbolic reasoning to generate information needed to execute tasks. However, the symbolic representation used in the study was relatively shallow and customized to produce functional knowledge to execute tasks, and the consistency in the knowledge base was not considered. Hence, executing symbolic inference processes with a large amount of implicitly encoded information in the control system might increase the computational cost. Moreover, the disambiguation process focused on creating a detailed robot plan of actions to be executed; by contrast, our objective is to disambiguate the objects required in the actions and complete the plan to be executed.

Recently, KnowRob was extended and partially redesigned, and the new release was introduced as KnowRob 2.0. It integrates photorealistic rendering and acquisition of low-level robot data and information concerning tasks, contexts, and goals, among others [25]. KnowRob 2.0 implements hybrid reasoning, unifying inner world knowledge, virtual and logical knowledge, and knowledge acquired from perceptual data during task execution. It also includes a question-answering feature enabled by Prolog in the interface shell.

However, potential queries are meant to request manipulation information or motion parameters needed for a task. It does not consider actual human interaction to cooperate or support task decisions in dynamic plan generation. Furthermore, as stated in [25], the knowledge originated from data collected from different sources might be redundant and inconsistent, leading to the computation of multiple hypotheses to determine the correct answer. Moreover, the paper did not mention how to deal with contradictory information.

Our purpose is to reason on the natural language command against the knowledge base to identify vague commands in terms of the objects needed for a task rather than the motions. Moreover, we include information derived from human interaction to support task completion when doubts arise about the involved entities and the reasoning cannot resolve them.

1.3.2 Human-robot Interactive Systems

Apart from knowledge, service robots need the ability to interact with humans. Tutoring systems are an example of human–robot interaction. The use of robots in one-on-one tutoring promotes learning gains and strengthens engagement [26]. The combination of the use of a robot's gestures and adaptive training leads to better learning outcomes in children [27].

Systems relying on learning-by-teaching methods using robots that act as learners and receive correction from children increase children's abilities and overall performance [6]. Methods of finding the best strategies for teaching robots have been studied in several works. In particular, physical interactions and reductions in unintended learning enhance robot's learning efficiency [7].

Means of interaction is a key factor in human–robot interactive systems. In [8], a human–robot architecture for interactive learning and conceptual reasoning using ontologies was described. The interaction is based on natural language communication, and it can learn concepts such as actions and targets of actions, although low-level mechanisms for linking the concepts to physical actions were not considered.

Teaching and learning strategies are also prominent in human–robot interaction. Correct strategies or combinations of strategies can be used to interpret the meaning of human actions well, since every user interacts differently [9]. Other tools can be used to reduce the physical and mental demands of humans when they collaborate with robots. These include an interactive combination of projection and touch-enabled table and kinesthetic teaching with a high level of abstraction [28]. However, users might take some time to learn the correct method of using such tools.

1.3.3 Dialog Systems

The usage of spoken language to interact with systems is a natural way of communication in humans. Many works have attempted to use natural language as the primary means of communication. In [29], scene generation was achieved by using natural language, and it can be dramatically improved by online image cloud retrieval, and synthesized scene refinement through natural language.

Some widespread usages of dialog systems are chatbots, recommendation systems, and feedback collection. Sufficient and convincing information improves user satisfaction with virtual dialog [30]. Letting on the user's initiative creates a more appealing and entertaining experience [31], and gathering guest feedback with robots is a convenient and useful method for the hospitality industry [32]. Dialog systems also emphasize adaptation to new domains and changes in the knowledge content with new information [33].

Conversational robots are a central topic regarding the future of human–robot communication. Multiagent conversations and interruptions must be managed appropriately while considering the importance of different factors, such as the conversation topics and emotional behaviors [10]. Moreover, a user's experience in establishing dialogs with robots can be enhanced by switching comment-speaking between more than one robot and the human to increase the impression of dialog continuity and avoid breakdowns [34]. The robot can increase the engagement, attention, and understanding of the user by adapting its dialog through considering verbal and nonverbal feedback [35].

The use of dialog-based interaction is an important skill for robots socializing with humans. However, rather than creating dialogs for entertainment purposes, a home service robot needs to create context and environment-specific conversations to solve requests accurately. The dialog creation needs to incorporate grounded knowledge, including general and situation-dependent concepts. The information received through the conversation is essential for task planning and execution.

1.3.4 Service Robots

Systems and service robots that are in contact with humans require several features, such as general knowledge, interaction skills, and language skills. When a robot deals with a verbal request, it sometimes needs to clarify the elements involved in the human request. Research has been performed to evaluate different visualization mechanisms that will allow a user to determine objects that match a request, and to develop more features to help robots select correct ones, such as head-mounted displays, monitors, and projectors [36].

A robot requires techniques to connect verbs to their associated tasks intuitively and the tools or sensors needed to execute such tasks [37]. A study expanded a robot's language understanding and grounded physical objects through conversations with humans [38]. The system creates dialogs to confirm the understood command or clarify unknown words to later ground the concept. It incorporates the description of objects, including visual, audio, and haptic properties. Although the system can learn new concepts referring to object properties successfully, it still requires a considerable number of clarification questions, even after training perception and parsing modules, to understand before executing a command.

Language understanding is essential for service robots. Some probabilistic graphical models are trained by associating language to scene semantics and perceptual classifiers to map

natural language instructions correctly [11]. However, these models do not consider techniques of solving ambiguous instructions, such as verbal interaction.

The authors of [39] proposed an approach that combines existing inference technologies to identify a command's semantic information and improve the design of human–robot interaction architectures. This work was extended in [40] with the incorporation of discriminative learning and distributional semantics. Spoken language understanding is achieved using linguistic data and perceptual knowledge. The resulting semantic frames are mapped into plans whose respective arguments are associated with their corresponding actors. Although this approach deals with ambiguous sentence structures, it does not consider situations where knowledge is missing in the semantic map, and neither creates any spoken interaction other than upon receiving the initial command.

Another approach to language understanding was developed in [41]; in this approach, the interpretation of utterances and context information are represented as logical forms. Then, given a first-order formula, the framework uses theorem provers along with model builders as inference engines to find either a proof or a model that satisfies it. Despite the robustness of this approach, the search for a suitable model or proof might require several processes; thus, it will need a long time to solve a task but still not guarantee that the identified model is the most efficient. Although this approach implements clarification dialog when the recognition confidence is low, it does not consider the number of dialogs made before being able to obtain the information needed for the model or the proof.

In [42], a log-linear reward function model was presented to find the possible sequence of instructions considering environment context information to perform tasks, such as cooking (ramen). This method handles generalization to new environments and can deal with ambiguities, such as incomplete instructions. However, the instruction disambiguation focuses on finding the subtasks' sequence to perform when the main instruction does not explicitly describe all the required subtasks. In addition, this method does not consider disambiguation methods, such as spoken interaction, when the entities involved in performing the instruction are not precise.

1.3.5 Concept Learning

Ontology learning techniques have been explored in different research domains such as tourism [43], [44], medicine [45], [46], and computer science [47], [48]. Their main goals vary from enriching concepts in the ontology, extending it with new concepts, building taxonomies, and discovering new relations between concepts. Some examples of potential input data are text,

existing ontologies, databases, dictionaries, lexical resources, and corpus from the web. Moreover, the main techniques for ontology learning include term and relation extraction, syntactic patterns, statistical methods, and machine learning [49].

Researchers have developed tools for ontology development tasks, such as the ROBOT tool [50]. This tool helps to check logical errors and standard quality control; it eases the creation, maintenance, and release of ontologies, currently working only with Open Biomedical Ontologies [51].

In the field of robotics, an attempt to create an ontology standard for autonomous robotics was presented in [52]. The authors demonstrated some of the benefits that using ontologies provide to a system, such as semantic interoperability. The knowledge representation focuses on notions like behavior, function, goal, and task. Although the ontology includes object manipulation behaviors, it is not clear how the objects are described. Moreover, it is not mentioned if the robot can extend its knowledge autonomously.

An approach to expand the knowledge of a conversational agent during run-time was presented in [53]. It presents a technique for automatic knowledge extraction, and it was tested with a social humanoid robot in a residential care home setup. The insertion of concepts is accomplished using verbal interaction, and it includes cultural knowledge about the user. Before the robot inserts a new concept, it asks whether the new concept belongs to a previously mapped class. However, it will continue asking in the same way for each class of the ontology in a descending form which might lead to a rather prolonged confirmation process if the mapped class is too general. Furthermore, the correct insertion of concepts depends on the ability of the user to describe the new concept so that the system can find a matching or related class; this can end up in a limited meaning conceptualization.

A different approach for learning concepts is found in [54], where an integrated cognitive architecture was realized to learn concepts, actions, and language. The authors integrated multiple probabilistic generative models to accomplish this. Experiments proved that a robot was able to learn based on its own experiences while interacting with the objects and receiving feedback from the user. However, they do not consider any hierarchy or relation between the concepts related to objects learned by the robot. Also, the model does not consider reasoning on the newly acquired concepts.

Learning word meanings has been achieved using statistical models such as probabilistic latent semantic analysis and latent Dirichlet allocation [55]. The approach, further extended in [56], used multimodal information such as visual, auditory, and haptic information, and it can

learn object concepts incrementally. However, the categorization of objects is based on their visual features; hence the semantic relationships between the object concepts are not present.

Another approach for learning visual concepts is proposed in [57]. It collects training videos and uses human annotation for data collection. The vision system learns concepts that are used to extract related concepts with ontology knowledge. The system searches for more videos using those extracted concepts and repeats the process. It shows the importance of having a concept hierarchy. However, it is not a fully autonomous learning process since it relies on human annotation data.

In [58], an incremental object learning system is described. The system uses a few sample images to learn new objects and their categories visually. However, the category hierarchy of concepts is not considered in the learning process.

A model for automatic ontology learning from unstructured text data was proposed in [59]. A classifier is trained with domain-specific data and consists of a two-stage process. The stages consist of classifying the data into concepts and classify them into more specific types. They showed that polysemy is one of the most important features to consider for the classification and learning model. The evaluation of the ontology learning system showed good performance; however, it considerably decreased when the test data contained different distribution from the training data.

1.4 Thesis Organization

The thesis is organized as follows. Chapter 1 briefly describes the background, explains the research goal and contributions. It also describes the related work of this research. Chapter 2 describes the ontology-based knowledge management system which can interpret commands and interact verbally. Chapter 3 describes the implementation of ontology learning of new concepts that combine textural knowledge and visual analysis with user interaction. Finally, chapter 4 provides the conclusion and future work.

Chapter 2

Ontology-based Knowledge Management System with Verbal Interaction for Command Interpretation and Execution

In the endeavor of creating a system for service robots performing tasks at home, we focused on equipping the robot with structured knowledge of concepts and language communication skills that it can use throughout the execution of a task. As we showed in section 1.3.1, the representation of knowledge using ontologies offers an advantage in managing information containing descriptions of various concepts and describing their relationships. Therefore, we decided to use an ontology to provide the robot with the knowledge necessary to function effectively in a home environment.

Furthermore, sections 1.3.2 and 1.3.3 showed that human–robot interaction presents a more enriching experience when working with robots and a natural way of interacting for non-expert users. Hence, the selected communication method between the robot and the user is through verbal interaction. An important point to bear in mind is that most of the approaches developed for service robots focus on techniques for task planning (section 1.3.4); hence, the command disambiguation process covers handling missing subtasks descriptions needed to complete the main task. By contrast, our focus is on interpreting and disambiguating commands in terms of the objects involved to complete the main task.

A service robot can perform a wide range of tasks, such as finding objects, cleaning up an area, and cooking. However, we focus on a limited number of tasks that challenge the robot's knowledge and reasoning capabilities to handle objects categories. The tasks included are bringing, delivering, finding, getting, placing, taking an object, and going to a place. This set of tasks makes it possible to create variations of commands to test different aspects of the ontological knowledge representation, reasoning, and interaction process. To satisfy all the mentioned aspects of a service robot that we are targetting, we developed an ontology-based knowledge management system that combines reasoning and human-robot interaction to interpret natural language commands. The system contains inference capabilities as it is based on ontological knowledge, and the formulation of questions incorporates parts of this knowledge. The overview of the system is explained as follows.

2.1 System Overview

The system is composed of four main modules with different functions. These modules manage the robot's knowledge and reasoning, command analysis, decision-making, and talking interaction. These modules are interconnected and share information between themselves in different steps of the processes. One of the modules has a link to the robot's perception and control modules, which enables the robot to perform the actions physically.

The system can perform some generic tasks, such as bringing an object, delivering an object to a place, finding an object, going to a place, taking an object, and placing an object. An overview of the system is shown in Fig. 2.1. General explanations of the modules and their functions are presented in the following subsections.

2.1.1 Knowledge Management

The first part of this module is the main ontology, which contains knowledge about the objects in the environment and their relationships. For instance, we can instantiate an object named *mug* and create a statement indicating that it belongs to the class *Mug*. With this connection, the ontology can immediately relate the *mug* to similar superclasses, such as *Cup* and *DrinkingVessel*.

Using this ontology, we can assign properties to the instances of the objects it has, and define classes connected by property restrictions. Considering the object class *Mug*, we can

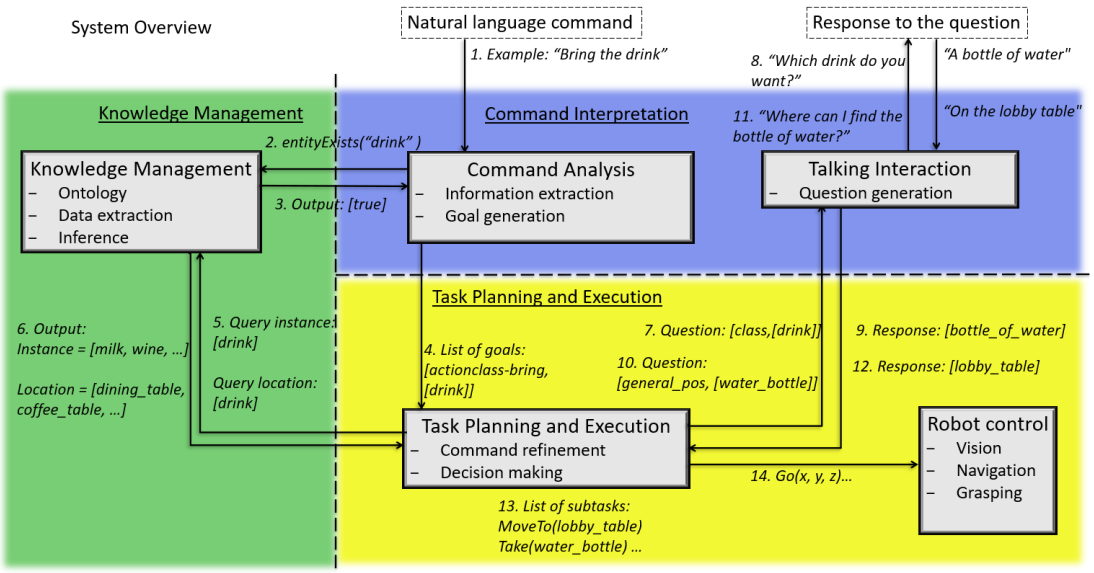


FIGURE 2.1: System overview.

assign the property *hasDefaultLocation*, which refers to a place where individuals of the class *Mug* can usually be found. The property *hasDefaultLocation* is linked to the class *Cabinet*, which is a *Furniture*. As a result, it is possible to deduce which furniture can be linked to a mug by the relation "*Mug hasDefaultLocation* some *Cabinet*".

This module contains the definition of the ontology and implements the formation of Description Logics (DL) queries to access its knowledge. The DL queries are class expressions based on the Manchester OWL syntax [60], a user-friendly syntax for handling ontologies designed using the Web Ontology Language (OWL). These DL queries are constructed as needed when a natural language command is received and during the information disambiguation process. Direct and indirect property values can be assigned to instances and classes using these queries. This module also makes complex inferences by querying links between the instances, properties, and classes, as explained in the subsequent sections.

2.1.2 Command Interpretation

2.1.2.1 Command Analysis

For a human, instructing a robot by speech is effortless and natural. One of the objectives of this module is to enable robots to receive commands through natural language and execute them.

This module processes a spoken command given by the human and creates goals based on the command. For instance, the human can tell the robot, “Go to the living room, take the soda, and place it on the coffee table". This module will generate the following set of goals for the command:

1. Go to the living room.
2. Find and grasp the soda.
3. Place the soda on the coffee table.

With this set of goals, the robot can begin the task execution process. However, the robot needs to verify whether some information is missing before accomplishing these goals. The missing information often depends on the actual environment setup, which could differ between houses. Some examples of possibly missing information from the previous set of goals are as follows:

- the location of the soda in the living room
- the type or name of soda, in case there is more than one
- the location of the coffee table
- which coffee table the human is referring to, in case there is more than one

The missing information differs according to the explicitness of the command. When the robot receives a command such as “Bring me the soda," it still needs to determine the location of the soda and the type of soda (if relevant). For this, the robot first creates DL queries based on the natural language command. The information obtained from that query is included in the generated goals. The clarification and disambiguation of this information are performed in the command refinement process that is part of the Task Planning and Execution module.

2.1.2.2 Talking Interaction

This module is in charge of the interaction between the robot and the human. It generates questions according to the missing information to complete a task. It also validates whether the answer fulfills the robot’s question. If necessary, the module restructures the question or asks a new one.

For the command “Bring me the soap," the robot can either think or interact to find the possible location of the soap. If the robot interacts, it may need to know the kind of soap using this module so that it can infer its location, or it can ask for the location directly. The robot might ask the following questions:

- Where is the soap?
- What kind of soap? Dishwashing? Laundry?

For instance, if the human response is “It is on the shelf," the robot might be confused as to which shelf in which room the human is referring to. By contrast, if the answer were “It is on the kitchen cabinet," the robot would directly go to the kitchen and approach the cabinet.

The next step for the robot is to determine the best question to ask by knowing its environment. The robot might encounter different situations. On the one hand, soaps could be kept inside a container in the laundry room. In that case, the robot needs to ask for the kind of soap the human wants, not its location. On the other hand, dishwashing soap could be in the kitchen cabinet, and laundry soap next to the washing machine. In this case, both questions will be helpful. In case there is no usual place for the soap, the best option would be to ask where the soap is. In summary, several situations arise depending on the environment setup. This module generates questions that will likely obtain the information the robot needs to fulfill a task.

2.1.3 Task Planning and Execution

This module generates a robot-specific subset of tasks based on the set of goals generated by the Command Analysis module. This subset describes each goal in a specific way such that the robot can execute them directly. For example, for the first goal, “Go to the living room," the subtask generated would be sending the coordinates of the living room to the navigation function.

Other goals may require more subtasks to be completed. For instance, when there is only one object called *soda*, the goal “Take the soda" would result in the following subtasks:

1. Receive the coordinates of the soda (from the previous goal, “Find the soda").
2. Check whether nothing is obstructing the path.
3. Grasp the *soda* located at the given coordinates.

These are examples of the possible subtasks created. However, the number of generated subtasks depends on the information given from the subset of goals and thus might vary.

The information included in the set of goals received from the Command Analysis module might not be enough to execute the subtasks. This module queries the Knowledge Management module to acquire the missing information; if it stills need added information afterward, it interacts with the user. The information found during the inferring process is used to create questions for the user. The robot is not expected to face a new concept; instead, it will receive a request of a known object in an unclear manner.

2.2 Knowledge Management

The organization and management of knowledge are an essential part of the proposed system. The robot needs to have a clear concept of the entities with which it interacts in its surrounding environment. We use an ontology to represent this knowledge. In this section, we explain the base ontology and then the inference processes along with their outputs, both corresponding to the Knowledge Management module.

2.2.1 Ontology Model Adaptation

We use the standardized description language Web Ontology Language (OWL). OWL has a high degree of expression, thereby enabling the inference of complex implicit knowledge [61]. Different ontology models can be found for robots, depending on their application. We adopt an open-source ontology for robots included in the KnowRob framework that best match our purpose. The KnowRob framework is designed to provide knowledge to totally autonomous robots [21].

However, our system aims to use language-based human interaction to supplement the robot's knowledge and facilitate natural human–robot communication. KnowRob requires translation of predicates to query the ontology, which sometimes results in unnecessarily long and inefficient queries [62], and the interpretation does not consider human interaction to solve ambiguities. In our system, natural language commands are automatically translated into queries, and they are interpreted using ontological knowledge combined with spoken interaction with humans.

For these reasons, we utilize only the base ontology instead of the entire framework. The base ontology contains the conceptualization of household domains, which aims to provide vocabulary that describes events, objects, actions, states, and parameters. Although the base ontology contains well-described concepts, we add certain types of classes and object properties. We also instantiate the individuals representing the real objects in the environment and their respective associations.

We perform this adaptation to complete missing class names according to the description of our environment, and to facilitate querying classes that are frequently used. Due to these changes, we have to modify some object property names, change their data types, and adjust some of the relationships between them.

We use statements to describe the concepts (axioms) of classes and properties in the ontology. By using axioms, we can create and associate, for instance, classes, properties, restrictions, individuals, and intersections. An axiom can describe the connection between classes with a *SubClassOf* property by assigning a value to a data property, such as *integer*, *Boolean*, *float*, and other *data types*. It also describes the *domains* and *ranges* of properties and indicates whether an entity is equivalent to another entity, and other varieties of restrictions. The ontology management module reads this information and creates OWL axioms that are based on this information. These axioms are added to the ontology, whose consistency is then checked.

Figure 2.2 shows the part of the ontology containing concepts from the *HumanScaleObject* class and the *SpatialThing* class with some relationships. The descriptions of four relations are as follows:

1. The first relation (1R) connects the *EdibleFruit* class (1a) with the *EatingTable* class (1b) using the property *hasDefaultLocation*. This connection means that the default location of entities from the *EdibleFruit* category is some entity falling into the *EatingTable* category (Fig. 2.2, items framed in blue).

2. The second relation (2R) sets up the domain and range for the *storedAtPlace* property. It assigns the domain as part of the *HumanScaleObject* class (2a) and the range as *Place* class (2b). This statement means that entities that can use this property must belong to the *HumanScaleObject* class, and the entities that will be linked need to be from the *Place* class; hence, an object can be stored at some place (Fig. 2.2, items framed in green).

3. The third relation (3R), similarly to the first relation, the *Cupboard* class (3a) is linked to the *FoodVessel* class (3b) using the *StoragePlaceFor* property, meaning that entities

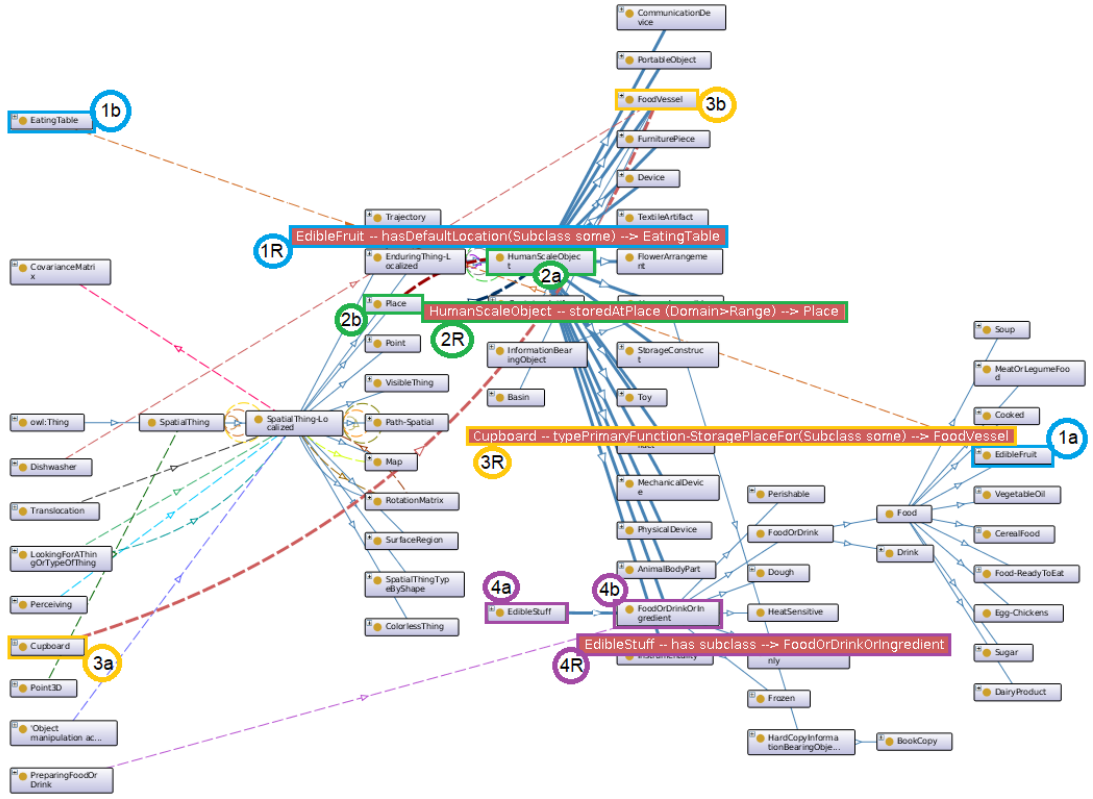


FIGURE 2.2: Visualization of a part of the ontology containing concepts from the *HumanScaleObject* class and the *SpatialThing* class with some relationships.

belonging to the *Cupboard* category will be a storage place for *FoodVessel* entities (Fig. 2.2, items framed in yellow).

4. The fourth relation (4R) connects the *EdibleStuff* class (4a) with the *FoodOrDrinkOrIngredient* class (4b) by the *subclassOf* property (Fig. 2.2, items framed in purple).

We supplement the ontology with instances of objects possibly found at home, as explained in the experimental scenario shown in Section 2.4. The Common sense regarding the objects' default location is established according to the house's layout in the experiments. This knowledge can be easily extended if necessary, as long as the concepts are appropriately described and associated. Moreover, it can be changed to describe new environments, such as hospitals and nursing homes.

2.2.2 Inference Process

In OWL, inferences are used to access to concepts in the ontology. We rely on a so-called reasoner to access the information in the ontology using DL queries. A semantic reasoner is a piece of software that can infer logical consequences from a set of asserted facts or axioms. It helps reduce the redundancy of information and finds conflicts in knowledge content [63]. If the ontology is not consistent, some conflicts or misconceptions might arise. Using the reasoner, we perform the query answering to find asserted statements, infer knowledge, check consistency, and identify hierarchical relationships.

We use the OWL API [64] for querying the information in the ontology. This library contains functions for working with OWL ontologies using the Java programming language and tools for DL parsing. We use this library to design methods of accessing the information stated in the ontology, perform query reasoning through DL queries, and extract the output information as needed.

Some of the reasoners' methods, the queries constructed during the command interpretation and disambiguation, are explained below.

Algorithm 1 receives the name of an individual and an indicator to return direct or potentially direct classes. It first brings the instance of the individual by using the input name on line 1. Then, it obtains the class types of the individual on line 2. Finally, it extracts the name of the first immediate class and return it (line 6).

Algorithm 1 getFirstTypeOfIndiv. It gets the name of the first type of class of an Individual.
Input: The name of the individual *indivName* and an indicator for considering just direct classes *isDirect*
Output: The name of the class $firstClassName$

1: $indivInstance = \text{GETONEINDIVIDUAL}(indivName)$
2: $nodesetClasses = rsn.\text{GETTYPES}(indivInstance, isDirect)$
3: $entitiesList = nodesetClasses.\text{ENTITIES}()$
4: $firstType = entitiesList.\text{FINDFIRST}()$
5: $firstClass = firstType.\text{GET}()$
6: $firstClassName = \text{GETNAMEOF}(firstClass)$
return $firstClassName$

Algorithm 2 confirms whether an *individual* exists inside the ontology by accessing the method included in the library to bring the list of individuals (line 1). It iterates (line 3), checks whether the given name matches the name of an *individual*, and returns whether it exists. This method is useful, for example, for checking whether the object that the user is asking for is instantiated in the robot's knowledge.

Algorithm 2 checkIndividualExists. It confirms if an Individual exists.

Input: The name of the individual *indivName*

Output: Whether the individual exist or not, $true$ or $false$

1: $listOfIndividuals = ontology.\text{INDIVIDUALSINSIGNATURE}()$
2: $entityExists = false$
3: **for each** $i \in listOfIndividuals$ **do**
4: **if** $\text{GETNAMEOF}(i) == indivName$ **then**
5: $entityExists = true$
6: **end if**
7: **end for**
8: **return** $entityExists$

With Algorithm 3, we can access the value assigned to a *data property* of an *individual*. Since it receives the name of the individual and the data property to access, it first looks for the instance of those entities in the ontology (line 1). Then, it applies one of the reasoner methods to extract the values by sending the instances on line 4.

Algorithm 3 getDataPropValuesOfIndiv. It gets the value from a Data property of an Individual.

Input: The name of the individual *indivName* and the name of the data property *dtPropName*

Output: Value of the property *value*

1: $indInstance = \text{GETONEINDIVIDUAL}(indivName)$
2: $dtPropInstance = \text{GETONEDATAPROPERTY}(dtPropName)$
3: $value = empty$
4: $aLiteral = rsn.\text{DATAPROPVALUES}(indInstance, dtPropInstance)$
5: $value = aLiteral.\text{GETVALUE}()$
return $value$

Not all the values of a property can be obtained immediately with the names and instances of the entities provided. Thus, we need to look through their connection with other entities as well. We created additional general methods to access inferred knowledge through *ClassExpressions* with a DL query parser. These methods receive a DL query, which is parsed to an *OWLClassExpression*. Then, they extract the required entities using the reasoner. For instance, Algorithm 4 retrieves the list of *SuperClasses* that satisfies the given expression.

Algorithm 4 getSuperClassesOfClassExpression. It gets the list of Super classes of a Class-Expression.

Input: The DL-query *classExpDL* and an indicator to clarify if the output should contain only directly stated classes or with potentially direct classes *isDirect*

Output: A list of super classes *allSuperClassesList*

1: $allSuperClassesList = empty$
2: $classExpParsed = \text{PARSEDLEXPRESSION}(classExpDL)$
3: $nodesetClasses = rsn.\text{GETSUPERCLASSES}(classExpParsed, isDirect)$
4: $entitiesList = nodesetClasses.\text{ENTITIES}()$
5: **for each** $cl \in entitiesList$ **do**
6: $\quad tempClassName = \text{GETNAMEOF}(cl)$
7: $\quad allSuperClassesList.\text{ADD}(tempClassName)$
8: **end for**
9: **return** *allSuperClassesList*

These algorithms are used at different steps of the command interpretation process to extract data and infer knowledge. The received linguistic command is preprocessed, as explained in Section 2.3, to extract the essential words from it and create query statements.

Ideally, the linguistic representation of the objects extracted from the command should be mapped to the individuals; this would require less inference process to know with certainty the objects needed from the real world. However, we consider cases where the user can ask for a specific target object (*individual*) or a general target object (*class*) in the command, e.g., “Bring the bottle of water" and “Bring a bottle".

In the first command, the target object can be mapped directly to an *individual bottle_of_water*. The second command refers to the class *Bottle*, in which case the system needs to

find the specific individual to complete this action. For the second case, the linguistic representation can be mapped to a class first, which would require more inference process and a possible speech interaction with the commander.

The following shows an input and output for the abovementioned algorithms. When the robot receives a command, such as “Bring the milk," it sends the input = [milk] to Algorithm 2 to verify whether an object called *milk* exists inside the current environment, and it receives back *true* or *false*. Moreover, by sending the input = [milk, graspable] to Algorithm 3, it can verify whether the object *milk* can be grasped by the robot; for this property, it sends back *true* or *false*. For the other properties, it can return other kinds of values, such as *black* and *white* when color is queried.

Some of the processes that utilize these algorithms are shown in Section 2.3. The entire list of methods for accessing the ontology can be found in Table 2.1 for data reasoning and Table 2.2 for data access. The data reasoning methods help obtain information that is not necessarily stated in the ontology and thus needs to be deduced from the assertions. On the other hand, the data access methods retrieve information that is directly stated in the ontology and can access any property of it.

2.3 Command Interpretation and Command Refinement

The proposed system aims to interact with humans by receiving spoken commands and executing them. The robot must understand the meaning of the words and what these words refer to in each command. To accomplish this, the system transforms the command given in natural language into DL statements to query the ontological knowledge. The obtained information is used to generate a set of goals that are based on it. When uncertainties are found, the information is validated and refined first through alternative assertions in the ontological knowledge and inferring processes, and then through interaction with the human. In the following subsections, we explain how these goals are created.

2.3.1 Information Extraction and Grammar Rules

In this subsection, the processes performed by the Command Analysis module are explained. The first step in analyzing the command is part-of-speech (POS) categorization, and dependency parses identification. With POS categorization or POS tagging, we can divide the

TABLE 2.1: List of data reasoning methods for general domain.

Name	**Description**
getAllClassesofIndividual	Brings a list of the classes that an individual belongs to and their superclasses
getDataPropValuesOfIndiv	Brings the value from a data property of an individual
getEquivalentClassesOfClassExpression	Brings a list of the equivalent classes that describe a given class expression; it can be specified to return directly stated or potentially direct
getFirstTypeOfIndividual	Brings the name of the first type of class of the given individual
getIndividualsOfClass	Brings a list of individuals that belong to the specified class
getInstancesOfClassExpression	Brings a list of individuals that satisfy the class expression
getObjectPropertyofIndividual	Brings the value from an object property of an individual
getPropValuesOfIndiv	Brings the value from a property of an individual; it does not need to specify the type of property
getSuperClassesOfClassExpression	Brings a list of the classes that describe a given class expression; it can be specified to return directly stated or potentially direct
individualBelongsToClass	Verifies if the given individual belongs to the specified class

words that share common grammatical properties and assign them to a class [65]. With the dependency parser, we obtain the type of relationship between the “head" words and their dependent words, which modify them. We implement Stanford Log-linear POS Tagger [66] and Stanford Parser [67], which are parts of the Stanford CoreNLP toolkit [68], for this purpose.

After the tagging and dependency parsing are completed, the system extracts relevant information to create a DL query which will help find the objects' instances satisfying the given concept description required for each task. The results of the query are passed as arguments to the next step. Finally, the system identifies the *action classes* and their respective arguments, which are called *keywords*. We define an *action class* as a group of verbs sharing similar semantic properties.

Patterns of a particular group of verbs can be defined considering the lexico-syntactic structure of the verb phrase. For instance, the verb “go" is commonly followed by a noun that represents a place. Some verbs require more complex validation, such as double-object verbs [69]. Therefore, we can identify *keywords* for each *action class*. The *action classes* and their

TABLE 2.2: List of data access methods for general domain.

Name	Description
checkExistsClass	Verifies if the given class exists
checkIndividualExists	Verifies if the given individual exists
containsADataProperty	Verifies if the ontology has the specified data property
containsADataType	Verifies if the ontology has the specified data type
containsAnIndividual	Verifies if the ontology has the specified individual
containsAnObjectProperty	Verifies if the ontology has the specified object property
entityExists	Verifies if the given entity exists
getClassesName	Brings a list of the names of all the classes
getFirstInstanceOfClass	Brings the first instance found of the given class
getImmediateSuperClassOf	Brings the first superclass that the given class belongs
getIndividualsName	Brings a list of the names of all the individuals
getNameofClassasOntology	Brings the name of the given class, as defined in the ontology
getOneClass	Brings the requested class
getOneDataProperty	Brings the requested data property
getOneDataType	Brings the requested data type
getOneIndividual	Brings the instance of the requested individual
getOneObjectProperty	Brings the requested object property
getOwlDatatype	Brings the requested instance of OWLDatatype
getOwlLiteralofDatatype	Brings an OWLLiteral with the specified value and data type
getRangeOfDataProperty	Brings the range of the given data property
getSuperClassesOfClass	Brings the list of superclasses that the given class belongs

respective *keywords*, which represent objects or locations that can be identified, are defined in Listing 2.1. The inferred information is used to generate the corresponding *action class*.

LISTING 2.1: Definition of Commands received by each Action Class.

⟨*actionclass-bring*⟩ ::= ⟨*verbs-bring*⟩ ‘`the`’ ⟨*object*⟩ [‘`from the`’ ⟨*location*⟩]

⟨*actionclass-deliver*⟩ ::= (⟨*verbs-deliver*⟩ | ⟨*verb-take*⟩) ‘`the`’ ⟨*object*⟩ ‘`to the`’ ⟨*furniture*⟩

⟨*actionclass-find*⟩ ::= ⟨*verbs-find*⟩ ‘`the`’ ⟨*object*⟩ [‘`in the`’ ⟨*location*⟩]

⟨*actionclass-go*⟩ ::= ⟨*verbs-go*⟩ ‘`to the`’ ⟨*location*⟩

⟨*actionclass-get*⟩ ::= ⟨*verbs-get*⟩ ‘`the`’ ⟨*object*⟩ [‘`from the`’ ⟨*location*⟩] | ⟨*verb-take*⟩ ‘`the`’ ⟨*object*⟩ [‘`from the`’ ⟨*location*⟩] [‘`to the`’ ⟨*furniture*⟩]

⟨*actionclass-place*⟩ ::= ⟨*verbs-place*⟩ ‘`the`’ ⟨*object*⟩ ‘`on the`’ ⟨*furniture*⟩

⟨*location*⟩ ::= ⟨*furniture*⟩ | ⟨*room*⟩

⟨*verbs-bring*⟩ ::= ‘`bring`’ | ‘`give me`’

⟨*verbs-deliver*⟩ ::= ‘`deliver`’

⟨*verbs-find*⟩ ::= ‘`find`’ | ‘`locate`’ | ‘`look for`’

⟨*verbs-go*⟩ ::= ‘`go`’ | ‘`navigate`’ | ‘`enter`’

⟨*verbs-get*⟩ ::= ‘`get`’ | ‘`grasp`’ | ‘`pick up`’

⟨*verbs-place*⟩ ::= ‘`place`’ | ‘`put`’

⟨*verb-take*⟩ ::= ‘`take`’

⟨*furniture*⟩ ::= instance or class name of type furniture

⟨*room*⟩ ::= instance or class name of type room

⟨*object*⟩ ::= instance or class name of a graspable object

2.3.2 Goal and final Subtasks Generation

After the *action class* identification and *keyword* extraction, it is verified whether no more information is needed or whether the information is not precise, such as when the same concept description describes several objects. Then, with the information completely disambiguated, the set of goals can be generated. The disambiguation process is explained in Section 2.3.3. Next, after forming the set of goals and *keywords* in the Command Analysis module, the system needs to create a collection of final tasks that are based on each goal; the robot will subsequently execute these tasks. The generation of the final subtasks is performed by the Task Planning and Execution module.

As shown in Listing 2.1, an *action class* can receive different numbers of arguments, depending on how much detail the human provides in the command. Therefore, we establish an algorithm for each *action class* that receives different combinations of *keyword* patterns to specify the series of subtasks needed. For instance, we create a method of selecting the valid entities required for the *action class Bring*; it uses the inferred keywords and disambiguates unclear data by interacting or querying related property statements, as shown in Algorithm 5, by which subtasks are generated (lines 25–28).

This algorithm describes the process of identifying the correct object and the specific source location to find it and bring it back to the commander. It starts receiving the *keywords* inferred from the natural language command, which contain the set of possible objects and

Algorithm 5 Break down the goal for the Action Class Bring, which takes the list of keywords inferred and the list of initial keywords.

Input: The list of inferred keywords *inferredKeywords* and initial keywords from the linguistic command *initialKeywords*

```
1:  objsOptions = inferredKeywords.GETOBJECTS
2:  sourceLocs = inferredKeywords.GETSOURCELOCATIONS
3:  if objsOptions is 1 and sourceLocs is 1 then
4:      validObjN = objsOptions.GETOBJECT
5:      validSrcLocN = sourceLocs.GETSOURCELOC
6:  end if
7:  if objsOptions more than 1 then
8:      typeObj = objsOptions.GETCLASS
9:      validObjN = CALLINTERACTION('class', typeObj)
10:     if sourceLocs is 1 then
11:         validSrcLocN = sourceLocs.GETSOURCELOC
12:     else
13:         validSrcLocN = CALLINTERACTION('general_pos', validObjN)
14:     end if
15: end if
16: if objsOptions not found then
17:     validObjN = initialKeywords.GETOBJECT
18:     alternativeLocs = MAKEDLQUERY(iObj, 'hasDefaultLocation')
19:     if alternativeLocs is 1 then
20:         validSrcLocN = alternativeLocs.GETSOURCELOC
21:     else
22:         validSrcLocN = CALLINTERACTION('specific_pos', iObj)
23:     end if
24: end if
25: MOVETOLOCATION(validSrcLocN)
26: TAKEOBJECT(validObjN)
27: MOVETOLOCATION(initial_position)
28: HANDOVEROBJECT()
```

source locations. The output of the given inference can be empty, which means the reasoning process did not find a suitable object and location according to the command description. This algorithm checks the cases where

- the object and the location are both found and unique (line 3),
- several objects correspond to the entire description (line 7),
- several objects and one source location correspond to the entire description (line 10),

- several objects and several source locations correspond to the entire description (line 13),
- no object corresponds to the entire description (line 16),
- there is an alternative location for the initial object when no other object matches the description (line 19), and
- there are several alternative source locations for the initial object when no other source location matches the description (line 22).

Once the concrete object and location are identified, then the robot can execute with certainty the rest of the commands included in this *action class* (lines 25–28), which are as follows:

1. Go to the place where the object is.
2. Take the object (including finding and grasping).
3. Go back to the place where the person is.
4. Hand over the object.

2.3.3 Command Refinement

The Task Planning and Execution module performs the command refinement process explained in this subsection, which needs information from both the ontological knowledge and the user interaction to disambiguate the commands. We explain some of the queries submitted to the Knowledge Management module (Section 2.3.3.1) and the generation of questions of the Talking Interaction module (Section 2.3.3.2), which help refine the user's command.

Communication in natural language can be complicated and ambiguous. When humans receive commands, they use different skills to understand them and be confident before doing them. Humans inadvertently access knowledge acquired from learned experiences and immediately know whether they have enough information. In this part of the system, we attempt to enable the robot to mimic this human behavior by using ontological knowledge as a first attempt to understand the command and then interacting with humans to solve ambiguities in the command.

2.3.3.1 Ontology Access

The robot can access the knowledge kept in the ontology at any time as it attempts to understand the command. However, there are cases where the inference process might not find an instance with the natural language command's initial description.

For instance, when the command requires a *book from the kitchen*, the knowledge will state that *books are usually in bookshelves*, and *bookshelves are not found in kitchens*. In this case, the initial query, which is based solely on *book from the kitchen*, will not retrieve any instance of a book. Consequently, an alternative *furniture name* where to find the *book* is needed. The robot first queries the ontological knowledge to find an alternative location. Then, if no output is retrieved, the robot proceeds to interact with the user, as explained in the following subsection.

For the example of Algorithm 5, on line 18, a query for finding an alternative source location of the initial object is generated when no inferred possible objects are received. The number of queries generated to find alternative instances of a concept description depends on the *action class* definition. For example, the *action class Deliver* needs an object and a destination, according to the definition shown in Section 2.3.1. This *action class* method will need to find an alternative source location and determine the correct destination if needed.

2.3.3.2 User Interaction

In regular conversations, humans omit details when they think they are unnecessary or to communicate faster. The same happens when humans give commands. However, the robot needs a way to know the omitted information. The robot can find such information by using its ontological knowledge to infer those statements. Another way is to interact with the person who is giving the command. The robot interacts with the user for assistance when the general inference rules cannot find the appropriate answer.

In this part of the system, the robot can create questions based on the information it needs to complete its understanding of the tasks to be executed. In a question generation system, the use of domain knowledge helps create significant questions and vary specificity [70]. We apply the notion of generating questions from concepts where we can formulate flexible questions instead of having completely fixed ones.

TABLE 2.3: Labels with semi-static questions and types of answers.

Label	**Question**	**Answer type**
aspect	What [property] is it?	Property description
class	Which [object_class] do you want?	Object name
general_pos	Where can I find the [object]?	Room or furniture
options_pos	Is it on the [furniture] in the [room_1] or in the [room_2]?	Room
specific_pos	In which piece of furniture can I find the [object]?	Furniture

We define the set of semi-static “Wh" questions shown in Table 2.3. The set includes labels identifying the types of questions and the strings describing them. Each question requires a word to be completed depending on the previously given task, and it expects specific information about the answers. The missing word and the expected answer can be the name of a graspable object, the object class, a property of the object, a room, or furniture.

Due to the broad information stored in the ontological knowledge base, we have the convenience of creating questions that add details, thus narrowing the answer expected from the user. The input for creating a question consists of two components: a label that describes the type of question to ask (e.g., position or object class) and a set of concepts that should be included in the question.

Consider the command “Bring me the salad bowl". Following the generation of the set of subtasks, the robot knows that it needs the furniture and the room. Then, the robot can use the input [general_pos, [salad bowl]], which will return the question “Where can I find the salad bowl?" After receiving the answer from the human, the robot needs to confirm the location of the given position (e.g., the *kitchen cabinet* or the *high table* in the *kitchen*) or only the name of the room (e.g., the *kitchen*). Although the *kitchen* is received as the answer, the robot still needs to find the furniture inside the *kitchen* where the *salad bowl is*.

The robot can encounter more complex situations where the questions must include more concepts to narrow the answer. If the user asks, “Bring me the book," the robot can take advantage of its ontological knowledge to create a more precise question.

Let us consider a scenario with two shelves — one in the living room and one in the bedroom. By querying the ontology, the robot finds that books are usually on shelves. The robot can use this information as an input, [options_pos, shelf, [living room, bedroom]], and receive a question, such as “Is it on the shelf in the living room or in the bedroom?" The response is narrowed to two options, unlike the possible responses to the question in the first example.

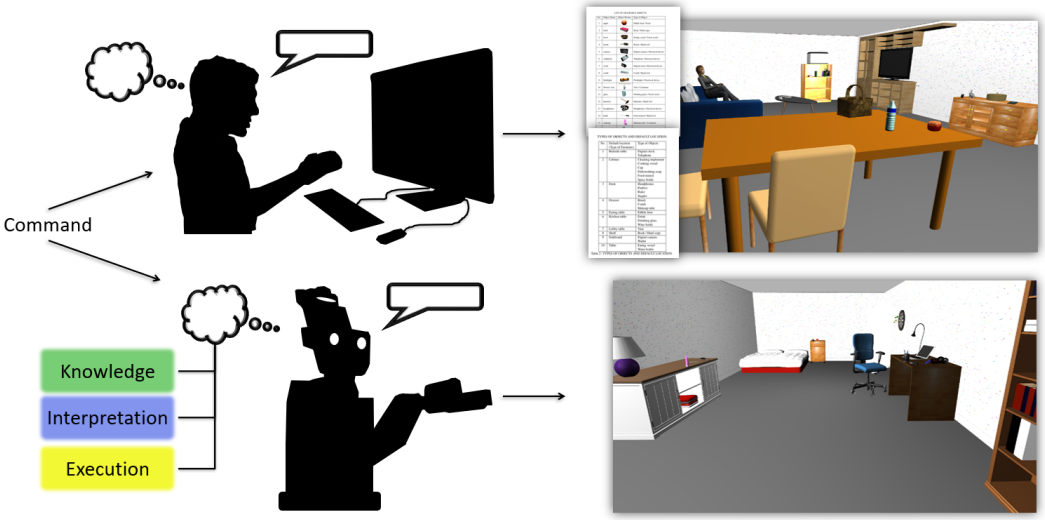


FIGURE 2.3: Outline of the experiments. Top: Human subject receiving a command. The subject's view includes information about the objects in the simulated environment, and the camera view perspective. Bottom: Robot receiving a command. The objects' information is in the ontology, and the camera view has the same perspective as the human subject. The figure shows the camera view of the living room (top) and the bedroom (bottom).

Interacting with humans enables natural communication and allows the robot to clarify the information before taking action. It also reduces the time needed by the robot to find the object.

By generating questions from concepts, we look for balance between the inference process and the questions made. The inferred knowledge gives us the advantage of selecting the pertinent question and including useful data about it. Moreover, we aim to avoid overusing the dialog to solve a task, which might lead to an unnecessarily large number of questions [38], [41].

2.4 Experiments in Simulated Environment

We conducted two separate experiments inside a simulated environment to analyze human behavior and compare it with the behavior of a robot that used our system. Fig. 2.3 shows the outline of the experiments.

In the first experiment, the human subjects received commands that they had to fulfill. They could interact with the commander with no restrictions, as the communication must be as natural as possible. In the second experiment, the same commands were given to the robot. It could use its knowledge, interact, or perform a combination of both to complete the tasks. In

these experiments, we can observe the different sequences of decisions followed by the subjects. The average solution for each command helped us discern what the subjects preferred to do to fulfill the same commands.

We investigated the combinations that led to possibly faster solutions. Furthermore, we used it as guidance to specify the range of an admissible number of decisions for each command. We analyzed and compared the results of both experiments, as shown in the following subsections. We aim to show the feasibility of using inferred knowledge along with verbal interaction to solve tasks such as searching and finding objects, rather than looking for a system that will outperform humans.

2.4.1 Environment Layout and Contents

We designed a four-room house using the simulator Gazebo. The house is composed of a kitchen, bedroom, living room with dining room, and lobby (Fig. 2.4). Each room has static furniture, such as tables, cabinets, a sofa, and a bed. It also has graspable objects, such as food, dishes, drinks, toys, electronic devices, and clutter (Fig. 2.5).

Most of the objects in the environment are organized intuitively, according to the house's layout. Food containers and kitchen utensils are in the kitchen, stationery on the office desk, and books on the bookshelf. However, there are some objects whose location might not always be the same, such as toys, tools, and personal items (e.g., wallet and cellphone). These objects are randomly located around the house and do not have a default location assigned. We used a virtually modeled Toyota Human Support Robot (HSR) [71] as the service robot for these experiments. It has the same features and configurations as the real HSR.

We composed eight commands, as shown in Table 2.4. They included going and finding commands. Some did not contain explicit details about the place to go to, the place to start looking for the object, or the specific object to search. The purpose of having commands with missing information and objects with no default location was to motivate reasoning about the context and interaction with the instructor.

2.4.1.1 Experiment with Humans in Simulation

In this experimental scenario, the human subjects had to follow the commands shown in Table 2.4 inside the simulation. We provided the subjects with a full list of the graspable

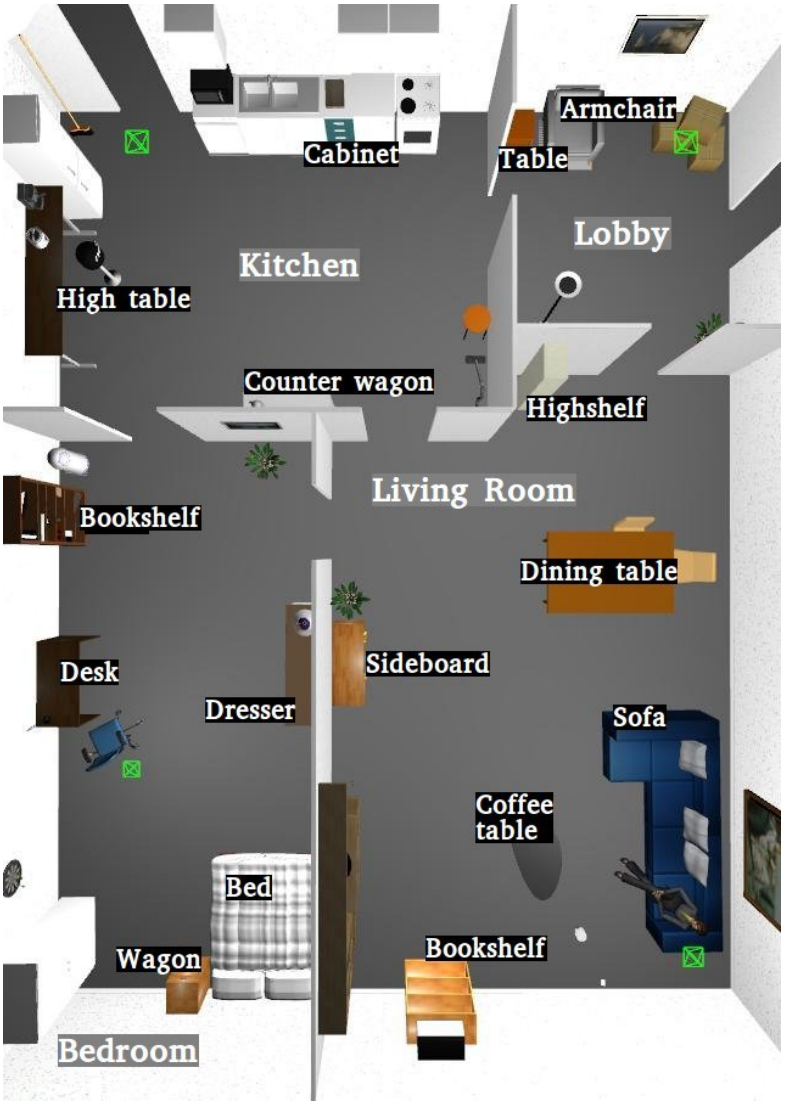


FIGURE 2.4: Layout of the house in Gazebo simulator.

TABLE 2.4: Commands used in the experiments with the ontology-based knowledge management system.

No.	Command
1	Go to the kitchen.
2	Find the bottle of wine in the kitchen.
3	Go to the living room and find a book.
4	Find a toy in the living room.
5	Find the makeup.
6	Find the Rubik's cube.
7	Find the hammer on the table.
8	Find a drink.

FIGURE 2.5: Examples of graspable objects.

objects, a list of the furniture, and a house map. The list of graspable objects included the name, picture, and type of the object. The list of furniture contained the type of furniture and the type of objects they usually store. The house map is shown in Fig. 2.4. During the experiment, the subjects were able to look through all three items at any time. Each subject's camera view provided a first-person perspective; the controls for the subject's robot included moving forward, moving backward, and turning; they were allowed to go around freely through the house until they finished the task.

We selected eight people who could speak English (basic to fluent) to fulfill each of the eight commands, resulting in a total of 64 natural language commands completed. Some of the subjects had previous experience with service robots and simulations. This diversity of the subjects provided contrasting conditions; the robot-experienced subjects were accustomed to similar simulations, but their English-speaking skills might lead to varied responses.

Measuring the performance of a service robot in a real environment is difficult. Some robotics competitions use independent test sets to benchmark robots' performance and abilities [72]. Such benchmarking is divided into system benchmarking and component benchmarking, where the system is evaluated as a whole and by a single functionality, respectively. The design of these benchmarks considers functional abilities, such as navigation, person recognition, mapping, speech recognition, and object manipulation. It also takes into account system properties, such as ease of use, calibration speed, ergonomics, and adaptivity [72], [73]. As service robots evolve and the environment changes, benchmarks need to be continuously improved, and new ones might appear [74].

However, these benchmarks mostly evaluate an entire system's or task's success or failure rather than assessing the way the task is completed. Our target is to observe the different

variations of decisions made to complete the commands in the experiments. For these reasons, we decided to estimate each subject's performance by the number of times they interacted with the instructor by asking questions, checked information about the environment, and thought and acted inside the simulation. To measure these data, we asked the subjects to verbally describe what they were doing or thinking during the experiment.

We counted every action (step) performed by the subjects that is, every time

- (ask) the subject interacted with the instructor, e.g., asked or confirmed information;
- (check) the subject checked information about the environment, e.g., location of furniture, class of an object;
- (infer) the subject made some inference, e.g., the food must be in the kitchen; and
- (act) the subject performed an action inside the simulation, e.g., moving from one room to another.

Examples of the steps taken by a human subject for the command "Find a drink" are listed in Table 2.5.

Figure 2.6 shows the average number of steps taken for each command. The graph shows that the predominant step is acting, whereas the others differ depending on the command. For instance, for the fifth command ("Find the makeup"), most of the subjects preferred to take action and attempted to find the *makeup* instead of accelerating the process by asking. However, most of the subjects were unsure about the appearance or location of the *makeup*, resulting in a large number of actions. The total number of steps could have been reduced by combining asking and checking information.

For the third command ("Go to the living room and find a book"), the combination of information checking, inferring, and asking resulted in fewer steps. Moreover, a *book*'s location is more intuitive to assume compared with other objects, such as a *Rubik's cube* or a *hammer*.

Regarding the seventh and eighth commands ("Find the hammer on the table" and "Find a drink"), the subjects needed all their skills (e.g., checking information, making inferences) to find the *hammer* and the specific *drink* correctly. In this environment, we did not assign a specific location for the *hammer*; thus, it could be found anywhere. Moreover, this scenario contained several drinks, such as *milk*, *wine*, and *water*. Hence, the last command required skills other than acting to complete it.

TABLE 2.5: Sample list of steps taken by a human subject for the command “Find a drink". The numbers, types, and descriptions of the steps are shown.

Step No.	Step Type	Step Description
1	Check	The subject accesses the environment information and checks which objects are drinks. Result: The subject finds milk, water, and wine.
2	Ask	The subject asks, “Any drink?" The commander replies: “A bottle of water please".
3	Infer	The subject thinks and decides that the first place to look at is the table in the lobby. Note: The subject was near the lobby when the command was received.
4	Act	The subject approaches the lobby table. Result: There is no water bottle on the table.
5	Ask	The subject asks: “Do you know where I can find it?" The commander replies, “in the living room".
6	Check	The subject checks the map information to find a table. Result: The subject finds the dining table.
7	Act	The subject goes to the living room.
8	Act	The subject approaches the dining table. Result: The subject finds the bottle of water.

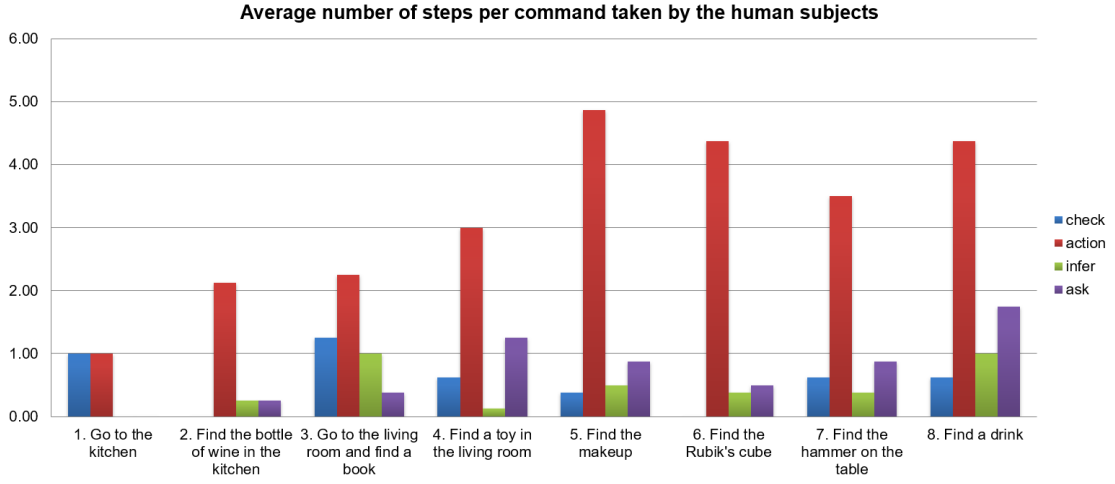


FIGURE 2.6: Average number of steps per command taken by the human subjects.

An interesting observation arose in these experiments. We expected the fluent English speakers to interact more with the instructor or create better questions, resulting in better performance. However, we noticed that the subject’s English-speaking skills did not interfere significantly. Some subjects who were not highly fluent performed better in certain commands. Their decisions may have been affected by their personalities rather than language skills.

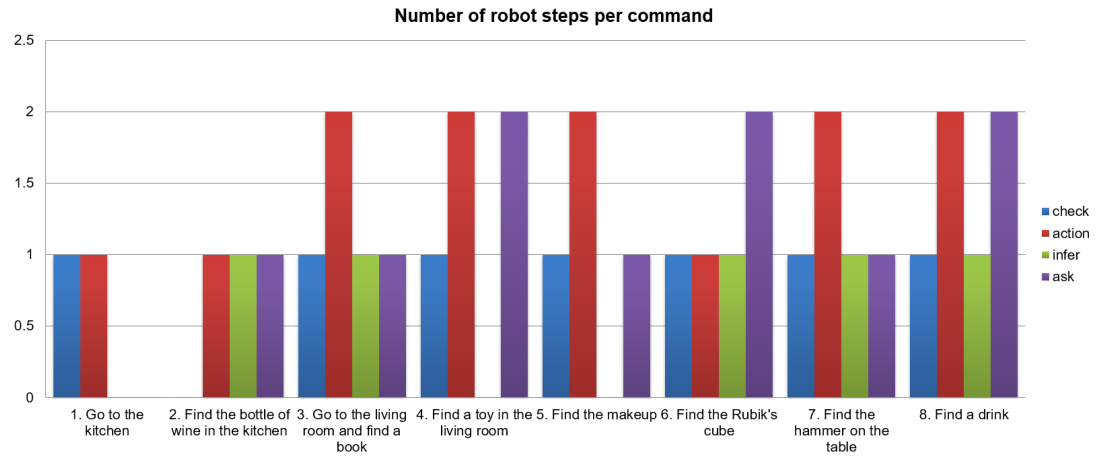


FIGURE 2.7: Number of steps per command of the robot.

2.4.1.2 Experiment with Robot in Simulation

To test our system, we performed the experiment using a virtual HSR with our proposed system as the robot's mind. The robot received the same set of commands as the humans did. It had the option of thinking (accessing the ontological knowledge), interacting, or performing a combination of them before executing the tasks. We determined its performance with the same level of definition as that in the experiments with the humans. A step was recorded every time the robot interacted, checked information, made an inference, or executed an action inside the simulation.

Figure 2.7 shows the number of steps that the robot made per command. The different combinations of steps and the total combinations per command are detailed. For most of the selected cases, accessing the information in the ontological knowledge base was essential. However, for the second command ("Find the bottle of wine in the kitchen"), the combination of acting, inferring, and asking was optimal for understanding and execution. Some commands could be completed in a few steps, while others required a larger number of steps due to different reasons. For instance, the second command already included certain information; thus, the robot needed only a few steps to complete the missing data.

The third and fourth commands ("Go to the living room and find a book" and "Find a toy in the living room") had a similar structure; both of them included the object class and location. The main difference was in the actual scenario; by making an inference, the robot could find the furniture where the *books* were. By contrast, to know the name and location of the *toy*, the robot needed to ask the instructor more questions.

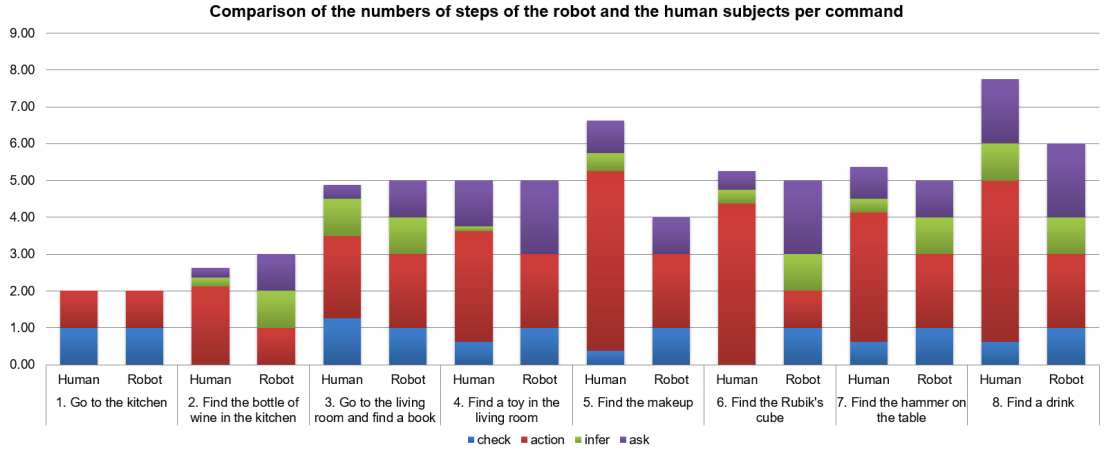


FIGURE 2.8: Comparison of the numbers of steps of the robot and the humans subjects per command.

The eighth command (“Find a drink") needed a larger number of steps. The robot needed access to ontological knowledge to check for objects that matched *drink*. It needed inference processes to know the possible location for the *drink* according to this environment. The robot also had to interact with the commander to know which *drink* they needed to find.

The robot attempted to balance its skills to acquire all the information needed to accomplish the tasks. Similarly, in the fourth and eighth commands (“Find a toy in the living room" and “Find a drink"), the robot benefited from its skill of interacting when its knowledge was insufficient.

2.4.1.3 Comparison and Analysis of Experiments

To determine whether our proposed system is adequate for such service robot tasks, we compared the results obtained in the abovementioned experiments. Figure 2.8 compares the average numbers of human and robot steps per command. Although the numbers of steps for some commands are similar, their combinations are different for almost all cases.

For the second command (“Find the bottle of wine in the kitchen"), the humans made fewer inferences but more actions than the robot, thus completing the task faster. However, for the fifth command (“Find the makeup"), the robot made fewer steps by checking for information in the ontological knowledge for this scenario. Regarding the sixth command (“Find the Rubik’s cube"); the total numbers of steps did not considerably differ. The humans preferred to do more actions and less reasoning, whereas the robot combined all of its skills. Measuring the results

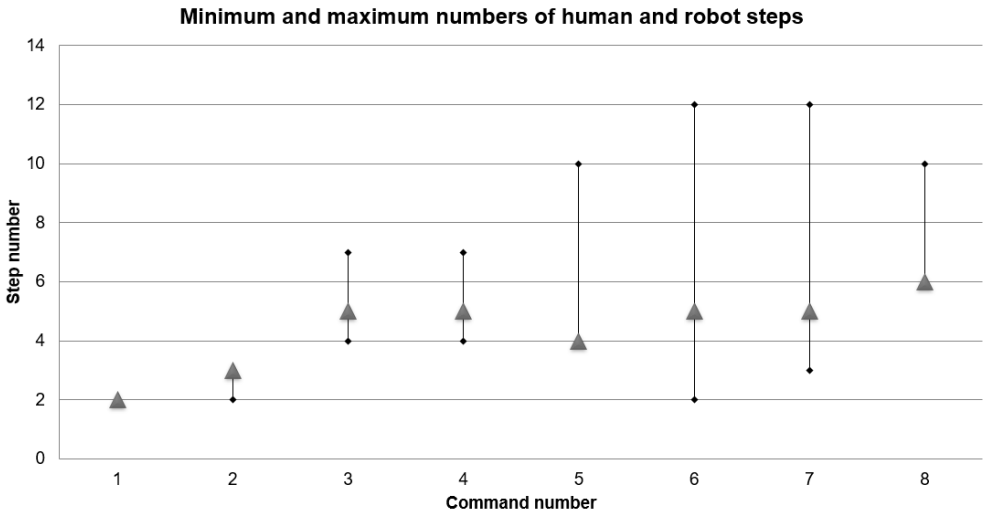


FIGURE 2.9: Minimum and maximum numbers of human and robot steps.

for this command through the number of steps might not yield a significant difference. However, in a real situation, acting requires more time than reasoning; thus, the humans took longer than the robot to complete this task.

Figure 2.9 compares the maximum and minimum numbers of steps taken by the humans and the robot for each command. The minimum number of steps may be deemed the best combination, since the more steps the humans took, the longer the time they needed to complete a task. The numbers of steps by the robot in all cases were within the minimum and maximum. For the fifth and eighth tasks, the robot's scores were close to the minimum, which meant the robot's performance was excellent.

The system showed competitive performance compared to the human subjects in the experimental setup. However, this might change if the experiments are performed in a real environment. The human subject's behavior could differ if they interact in an environment that is natural for them; hence, their performance could improve.

Knowledge management is an essential component of the proposed system; this was evaluated through the times the robot accessed stated and inferred knowledge. The human subjects' knowledge was similarly assessed; we observed during the experiments every time they used the environment's information given beforehand to check the data and infer facts. However, the subjects' knowledge would be more challenging to evaluate in a real-world experiment when the environment is utterly familiar to them; it is not known with certainty what knowledge they already have or what kind of inferences they would make.

2.5 Discussion and Summary

This chapter described an ontology-based knowledge management system with verbal interaction for command interpretation and execution by home service robots. We proposed the combination of ontological knowledge reasoning and human–robot interaction to interpret natural language commands. The system is composed of four main modules that (a) manage the robot's knowledge and reasoning, (b) analyze the command to generate goals, (c) refine the information and execute tasks, and (d) interact with humans by speech to disambiguate information. The system relies on inference methods and verbal interaction to understand commands and clarify uncertain information.

We tested the proposed system in a scenario where the robot received commands, such as going to rooms and finding objects, with missing or unclear information. It had to understand them by using reasoning and interaction to be able to execute them. We performed another experiment where human subjects solved the same set of commands. We then compared the performance of the system and the human subjects.

Some improvements could be achieved in future work, such as the interpretation of natural language statements with entirely new entities. To accomplish the mentioned skill, we need to implement a new sequence of actions to acquire a new concept and new dialog-based clarification methods to handle inconsistencies in ontological knowledge.

The concepts represented in the ontology can be easily extended with more home-related concepts or a new environment definition. The linguistic representation of verbs associated with the current action class description can also be extended. However, increasing the keywords involves modifying the goal generation's disambiguation process; adding new action classes requires the definition of new methods describing their main sequence. These bring an opportunity to adapt the system to create the subtasks sequence dynamically.

Chapter 3

Ontology Learning of New Concepts combining Textural Knowledge, Visual Analysis, and User Interaction

3.1 Problem Definition and Our Approach

Concept learning has been achieved using techniques for automatic knowledge extraction from ontologies, databases, dictionaries, and other resources from the web, as well as with human–robot interaction (section 1.3.5). The need for concept categorization and hierarchy has been emphasized when using significant amounts of concept descriptions. Moreover, using ontologies to store this knowledge opens the opportunity for semantic interoperability.

The ontology used in our proposed system for service robots includes the description of concepts of the objects that the robot operates within in its environment. These concepts have their corresponding characteristics, relationships, and instances of objects belonging to those concepts. Expanding this ontological knowledge is essential as the robot's environment can change over time.

Some challenges must be addressed to ensure the correct conceptualization during the ontology learning process, such as deciding where the new concepts will connect with existing ones in the current knowledge and preventing or dealing with inconsistencies. The scenario we face is as follows, a robot with general ontological knowledge is required to learn a new object,

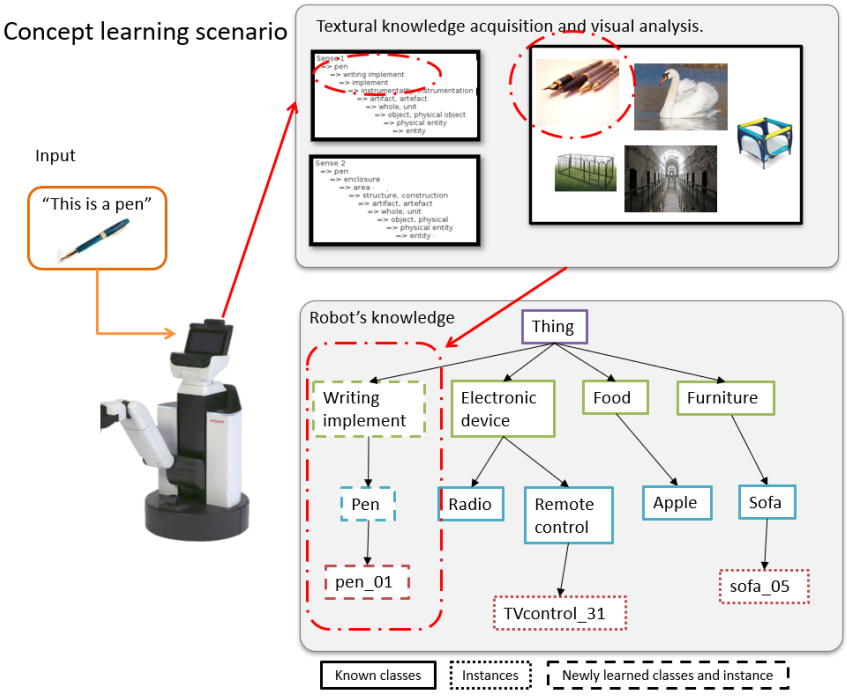


FIGURE 3.1: Concept learning scenario [75]. The robot receives the name of a new object, along with its visual image. On the right side, textural knowledge information and visual analysis are used to select the new hierarchy of classes to include in the current robot's knowledge.

which entails learning the new word concept to correctly create the new instance and the corresponding classes inside its predefined knowledge. Although the robot could use methods, such as using online resources or actively asking a user about the new object to extend its knowledge, we want the robot to learn new concepts as needed with the little burden on the user as possible.

For this purpose, we want to achieve the learning of a new concept using three methods. First, (1) using textural knowledge to identify the possible meaning of a new word; second, (2) performing meaning selection by analyzing the visual characteristics of related concepts against the new object; third, (3) creating user interactions to support meaning selection.

The proposed scenario to demonstrate the usability and feasibility of learning new concepts through textural knowledge and visual analysis is shown in Fig. 3.1. A service robot with vision capabilities and ontological knowledge is requested to learn a new object. First, the robot receives the name and the image of the new object. Then, the new object must be conceptualized to include it in the current knowledge.

3.2 Ontology Learning Components

Our approach for ontology learning of new concepts consists of four modules. These modules are responsible for (1) textural knowledge acquisition and semantic relation extraction, (2) image collection and analysis, (3) concept description selection, and (4) ontology updating. The modules are connected consecutively until finalizing with the object conceptualization process. These components enable the robot to learn new objects by conceptualizing them. The robot only needs the name of the new object and its image to start this process. It can also verbally interact with a user to obtain confirmation before conceptualizing a new object.

An overview of the ontology learning components is shown in Fig. 3.2. The concept learning process starts when a user shows the robot an image of the new object and names it; this information is sent to the Word Meaning Identification module. It acquires textural knowledge about the new object to extract its semantic relations as a concept. Then, all senses, which refer to the possible meanings of the new word concept, are sent, along with their semantic relations, to the Visual Analysis module.

The Visual Analysis module performs online image queries using the hypernyms contained in the semantic relations received. Next, it analyzes the images downloaded to find features similar to the image of the new object, and a similarity score is assigned to each sense. Senses with their respective similarity scores are sent to the Concept Description Selection module. In this module, the similarity scores are examined to choose a sense that best describes the new object concept to learn. When the robot finds more than one high similarity score, it resorts to interacting with the user to confirm the correct concept description. Once the robot knows which concept description best represents the new object, it sends it to the Knowledge Management module to update the ontology.

In the following sections, we explain the word meaning identification process, the visual analysis performed, the concept description selection, and the ontology update with new concepts.

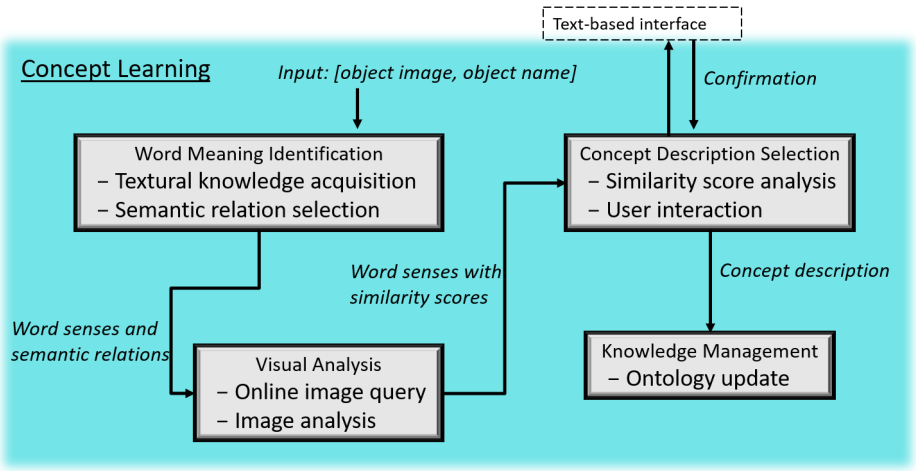


FIGURE 3.2: Concept learning components.

3.3 Word Meaning Identification

3.3.1 Textural Knowledge Acquisition

The ontological knowledge contains a collection of concepts of objects associated with each other by properties such as the “subclass of" property. It also contains instances of real objects belonging to some of those class concepts with their respective characteristics. An important factor in extending this knowledge is determining the correct correlated concepts and the corresponding categories that will connect to the whole hierarchy of concepts. According to this, the semantic relations associated with the new concept are needed. Therefore, we decided to employ online processing resources to acquire this information.

Textural knowledge containing semantic relations is based on a well-known lexical database WordNet [76]. To access the WordNet database, we use the general architecture for text engineering GATE [77], which can add WordNet as a processing resource. WordNet classifies the meanings of the new word into four types of POS tags: noun, verb, adjective, and adverb. We are only interested in the noun POS tag type since the robot will learn exclusively objects as concepts. This type of tag contains the collection of semantic relations (or synset) with associated concepts for each possible meaning (or sense). The types of semantic relations included in each collection are hyponym, meronym, and hypernym. However, only the hypernym type of relation will give us the categories into which the new word falls that can be connected to more general concepts in the ontology hierarchy.

Querying WordNet using GATE and extracting the semantic relations we need is shown in Algorithm 6. First, the POS tag and semantic relation are defined as noun and hypernym types, respectively (lines 1-2). Then, the robot queries GATE with the new word concept to retrieve the corresponding collection of noun POS tag senses only (line 4). The list of senses is looped (line 5) to get the synset (line 6) and extract the hypernyms (line 7). Since the list of hypernyms comes nested, they are extracted in two steps. First, the synset members are acquired (line 13), containing the list of words defining the current hypernym and its synonyms if applicable (line 25). This situation can be illustrated in Fig. 3.3 bottom, where the fourth hypernym of the Sense 1 contains two words “instrumentality" and “instrumentation." Next, the semantic relations of the hypernym type for that synset are pulled (line 15), and the same process of acquiring the synset members is performed recursively (line 18). The resulting list for the current sense is added to the main list of hypernyms (line 8). Last, the list of all hypernyms per sense is returned (line 10). This list represents the superclasses of the new word concept according to each sense that will be used to create the new concept in the ontology.

An example of a query for the word “pen" is shown in Fig. 3.3. The robot receives five different senses for the same word: writing implement, enclosed area, portable enclosure, correctional institution, and female swan. In Fig. 3.3 (bottom), the retrieved hypernyms for the first two senses are shown. After the robot obtains the list of senses and their semantic relations, it must identify which sense best describes the new word concept to add the corresponding semantic relation in the ontology.

3.3.2 Considerations in the Semantic Relation Selection

The selection of the correct semantic relation is a crucial step for ontology learning since this will establish the connection of a new concept with existing ones. In addition, new concepts that are correctly associated will allow the ontology to make inferences over them, such as deducing the possibly inherited characteristics from potential superclasses.

The new semantic relation to be added to the ontological knowledge must comply with the following guidelines:

- The sense of the chosen semantic relation must correspond to the correct meaning of the new word concept.
- The new semantic relation will be linked to the closest or an equivalent concept in the ontology hierarchy.

Algorithm 6 getHypernymsOfWord. It consults the GATE tool and extracts the list of hypernyms per sense of a given word.

Input: The new word concept *newWord*
Output: The list of hypernyms per sense *hypernymsPerSenseList*

```
1: posTag = POS_NOUN
2: relationType = REL_HYPERNYM
3: hypernymsPerSenseList = empty
4: sensesList = gateServer.LOOKUPWORD(newWord, posTag)
5: for each sense ∈ sensesList do
6:     synset = sense.GETSYNSET()
7:     hypernymList = GETHYPERNYMSOFSYNSET(synset)
8:     hypernymsPerSenseList.ADD(hypernymList)
9: end for
10: return hypernymsPerSenseList
11: function GETHYPERNYMSOFSYNSET(synset)
12:     synonymsList = empty
13:     synMembers = GETSYNSETMEMBERS(synset)
14:     synonymsList.ADD(synMembers)
15:     semRelationsList = synset.GETSEMANTICRELATIONS(relationType)
16:     for each semRel ∈ semRelationsList do
17:         currentSynset = semRel.GETTARGET()
18:         hypernymList = GETHYPERNYMSOFSYNSET(currentSynset)
19:         synonymsList.ADDALL(hypernymList)
20:     end for
21:     return synonymsList
22: end function
23: function GETSYNSETMEMBERS(synset)
24:     synsetMembersList = empty
25:     synonyms = synset.GETWORDSENSES()
26:     for each syn ∈ synonyms do
27:         lemma = syn.GETWORD().GETLEMMA()
28:         synsetMembersList.ADD(lemma)
29:     end for
30:     return synsetMembersList
31: end function
```

- The concept classes of the new word concept will be added sequentially following the semantic relation.
- The insertion of a new class will stop when the new class already exists in the ontology. Otherwise, a maximum of four new classes will be created as the higher the concept in the hierarchy is, the more general it becomes.

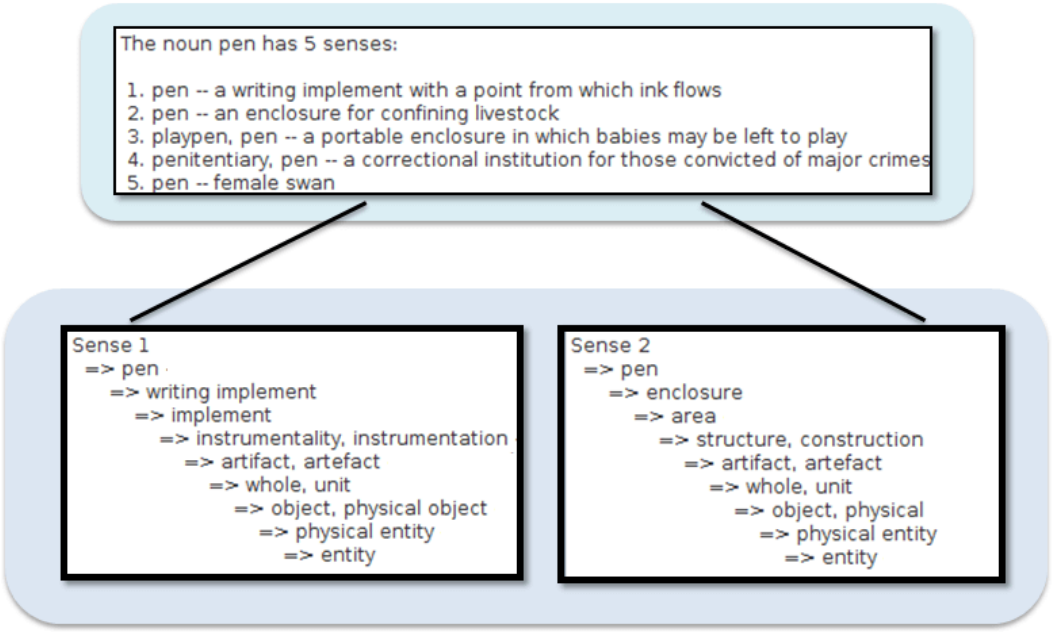


FIGURE 3.3: (top) List of senses from querying the word “pen" using GATE. (bottom) Hypernyms lists of the first two senses [75].

Semantic relations acquired from WordNet might contain general concepts that can be found in the ontological knowledge, such as an *artifact* or *physical entity* (Fig. 3.3 bottom). However, this does not ensure that those concepts accurately describe the new concept to be learned. Analyzing the visual characteristics of the new object and user interaction aid in determining the correct semantic relation.

3.4 Visual Analysis

We assume that the robot cannot recognize the new object with the current object identification module in the concept learning scenario. Hence, it is impossible to know the name or category of the object that could assert the position for the new semantic relation in the ontological knowledge. Therefore, we propose supporting the selection of semantic relations by analyzing the visual characteristics of the new object and comparing it with objects belonging to the possible semantic fields of the new object based on its name. This visual analysis process aims to find the correct semantic field of the new object by studying the similarities with other objects from a set of potential semantic relations. The process starts with an online image query

to collect images of hypernyms included in each potential semantic relation, followed by feature analysis, as explained in the following subsections.

3.4.1 Online Image Query

In the first part of the visual analysis process, the robot makes an online image query to collect image samples of object categories associated semantically. Potential categories are chosen from the list of potential semantic relations obtained previously in the word meaning identification process.

$$PSemRel_w = [S_1, S_2, \ldots, S_i]$$

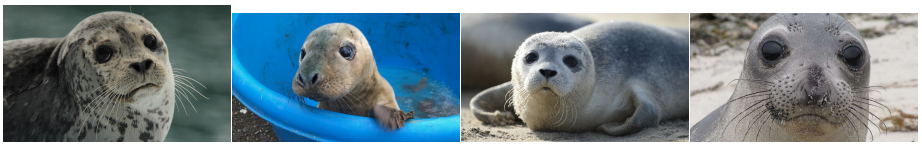
$$S_i = [H_1, H_2, \ldots, H_j]$$

$$H_j = [hypSyn_1, hypSyn_2, \ldots, hypSyn_k]$$

where:

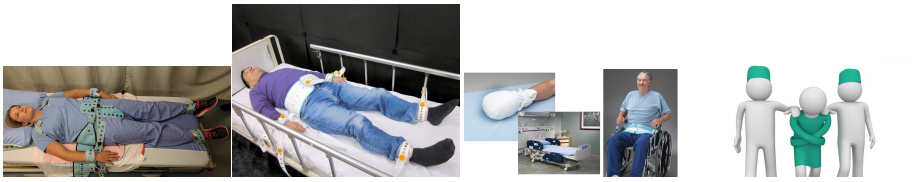
$PSemRel_w$ = list of potential semantic relations of the new word w per sense
S_i = list of sets of hypernyms belonging to the sense i
H_j = j-th set of hypernyms
$hypSyn_k$ = a hypernym or its synonym k

A query is formed using the new word w and a hypernym $hypSyn_k$, different from w, e.g., for the first sense of the concept *washer*, whose first three hypernyms are *worker*, *person*, and *organism*, the first query generated would be *"washer worker."* This method of formulating queries helps the online image query return good results according to the expected meaning. To illustrate this situation, Fig. 3.4 shows image results for querying the hypernyms of the second sense of the concept *washer* in different ways. According to WordNet, the second sense of the word washer refers to a "seal consisting of a flat disk placed to prevent leakage." The difference in image results depends on the query. In the case of (a), (b), and (c), when the query contains only a hypernym, image results do not fully represent the meaning required. This situation can confuse the robot's understanding since objects in the resulting images can be rather general and variable, not necessarily representing the expected meaning.



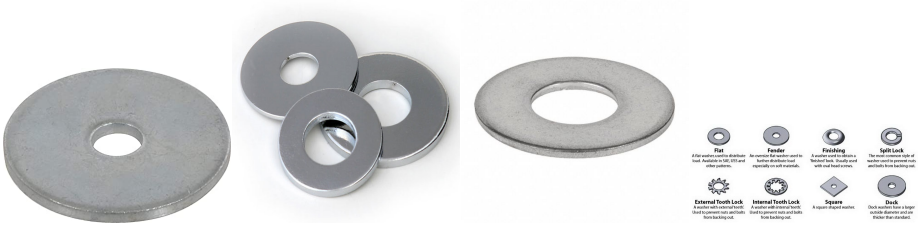
(a) Query: “seal"

(b) Query: “fastener"



(c) Query: “restraint"

(d) Query: “washer seal"



(e) Query: “washer fastener"

(f) Query: “washer restraint"

FIGURE 3.4: Examples of online image query for the first three hypernyms of the second sense of the word “washer."

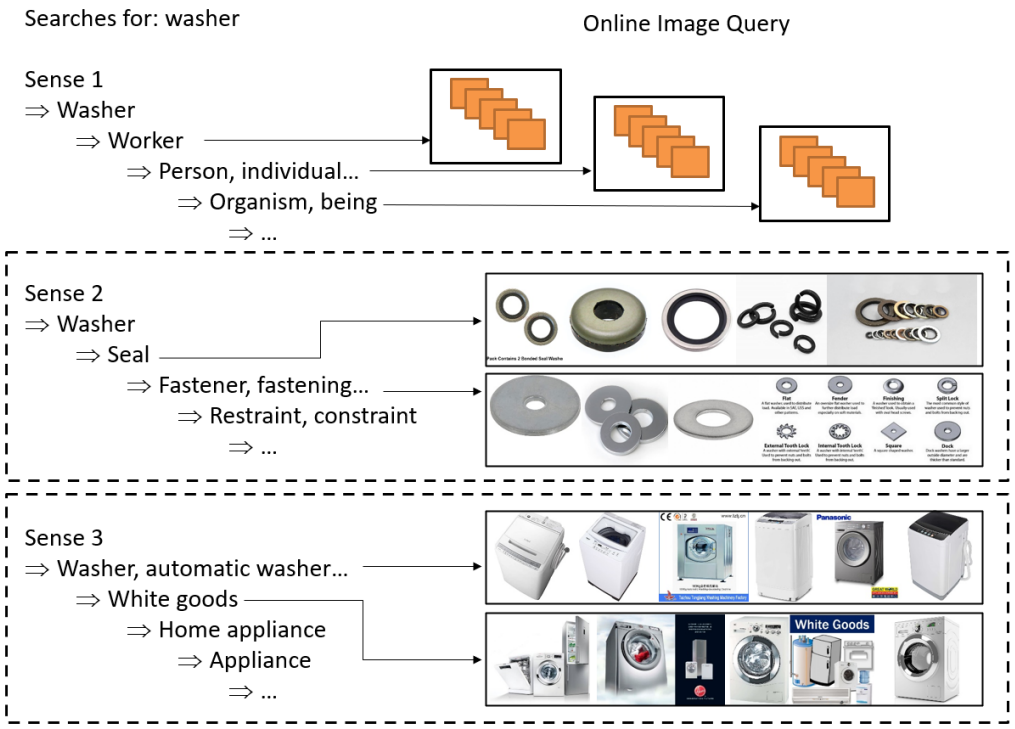


FIGURE 3.5: Online image query results for the concept *Washer*. The query generation uses the combination of the word concept "washer" following a hypernym, e.g., "washer seal," "washer fastener."

In contrast, adding the word *washer* to the query makes it more specific, and resulting images show objects closer to the meaning required (Figs. 3.4 (d-f)). Therefore, the robot can have a better approximation of objects belonging to the required sense.

Following the query generation method, the system creates online image queries for the first three sets of hypernyms H_j per sense S_i to download sample images (Fig. 3.5). Thus, these images symbolize examples of objects from each semantic relation. Fig. 3.5 depicts the hypernyms of three senses corresponding to the concept *washer* and examples of the online image query results for the second and third senses.

The importance of correctly identifying the sense that best describes the new object to learn cannot be overstated. Choosing the wrong sense could result in a completely different meaning being assigned to the new object, the incorrect concepts, and an incorrect hierarchy being created in the ontological knowledge. Once the online image query collects sample images of the potential semantic relations for the new object, the next step is to analyze the image features to find similarities between them to help determine the correct meaning for the new object.

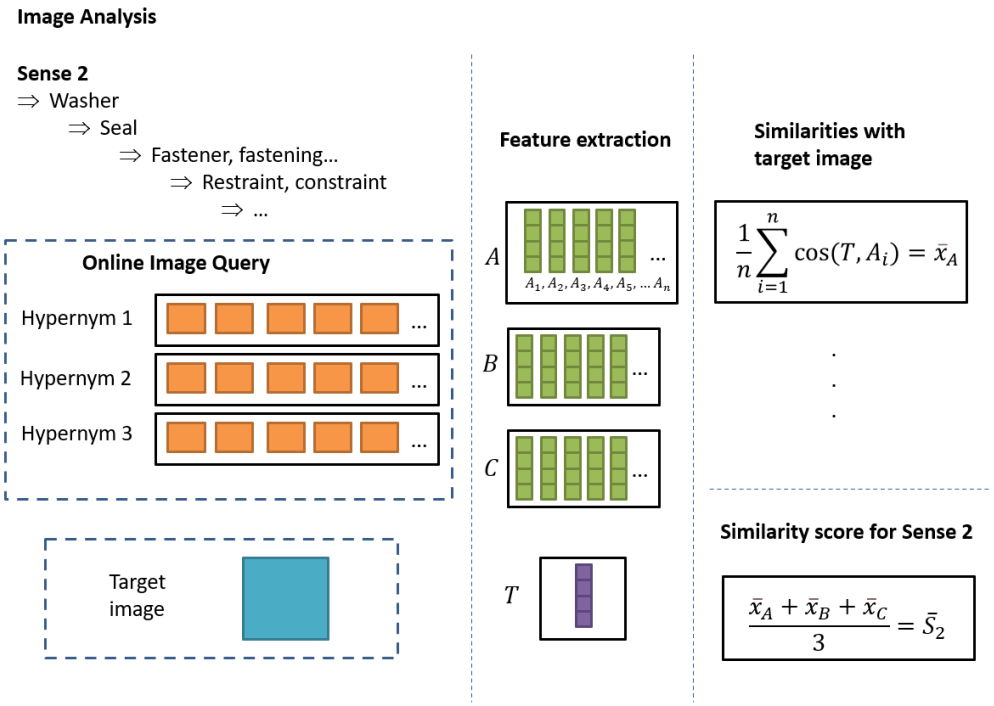


FIGURE 3.6: Image analysis process for similarity score assignment.

3.4.2 Image Analysis

The second part of the visual analysis process consists of comparing the new object's image with the potential semantic relation data collection. Up to this point, the robot has collected n number of images for the set of hypernyms H_j for each sense S_i in the online image query process. Next, the robot is expected to find a group of images belonging to only one sense similar to the new object image. We propose using a pretrained artificial neural network to extract image features of the new object and the downloaded collection of images. Then, it is possible to calculate a similarity value between them using the extracted features.

This study uses a deep convolutional neural network (CNN), namely, ResNet-152 version 2 architecture [78]. The CNN is pretrained on the ImageNet dataset [79]. We removed the network's last layer to extract a 2048-dimensional feature per image from the last fully connected layer. Next, the features were computed for the new object image and the downloaded images of all the potential semantic relations. Subsequently, the robot computes the cosine similarity between the features of the new object image and those of each image from the semantic relations. Then, the average similarity for each semantic relation, representing a sense, is calculated. Hence, each sense is assigned a similarity value. Fig. 3.6 depicts this process.

3.5 Concept Description Selection

Before the robot can add the new concept to its ontological knowledge, the final step is deciding which concept description best represents the new object shown. Thus, to recap, the first part of the concept learning process is the word meaning identification, which acquires the meanings and the semantic fields of the new word concept. The second part is image analysis, which collects image samples for each semantic field and finds similarities between them and the new object to learn. Finally, the robot needs to choose the best concept description for the new object.

3.5.1 Issues in Concept Description Selection

The selection of the concept description utilizes two resources, the similarity values computed during the image analysis process and user interaction which will add confidence to the selection. Ideally, the sense with the highest similarity score would be the best to describe the new object concept. However, the similarity is significantly affected by the variation of results during the online image query, as shown in Fig. 3.5. There are some situations when image results may contain unrelated images, even with the descriptive query. That is the case of Fig. 3.7, where querying *"pitcher containerful"* retrieves mixed results. Therefore, it is necessary to emphasize that having additional images for each semantic relation contributes to a more accurate differentiation.

The senses of a word can refer to multiple different meanings for the same word concept. Hence, the robot needs to be sure that the correct concepts will be added to its ontological knowledge. There are three main cases to consider regarding the results of the similarities of images:

1. The sense with the highest similarity score is the correct one, and no other sense has a close similarity score.
2. The sense with the highest similarity score is the correct one, and a second sense has a close similarity score.
3. The sense with the highest similarity score is the incorrect one.

As previously mentioned, the first is the ideal case since the new object would be adequately conceptualized. However, for the second and third cases, an additional method is

(a) Query: “pitcher containerful"

FIGURE 3.7: Example of online image query using the query “pitcher containerful".

required to confirm that the correct sense is being selected. Therefore, we propose assisting in selecting the concept description using human–robot interaction, as explained in the following subsection.

3.5.2 User Interaction for Concept Description Selection

In the concept learning process, a situation may occur where the robot requires the user’s approval for the series of concepts that it is about to learn. Also, as discussed in the previous chapter, human–robot interaction can be considerably helpful in a service robot environment.

The robot may encounter three main scenarios, previously discussed, after obtaining image similarities: when the highest score is either for the correct or incorrect sense, and when the second-highest score is close to the first. We propose using human–robot interaction to provide final assistance in the concept description selection depending on which scenario the robot faces, having the following benefits:

- it helps the robot confirm that it is conceptualizing the new object concept correctly,
- it helps the robot decide which definition of the new object is accurate when the image analysis results are not confident,
- it gives assurance to the user that the robot is learning the new object correctly, and
- teaching a new object to the robot would be more interactive for the user.

The robot creates three types of dialog sentences for (1) requesting the image and the name of the new object, (2) reporting that it saved the new object and a one-word description

of it, and (3) allowing the user to choose between two options of concepts that possibly define the new object. While the first and second dialog sentences are always used in the concept learning to start and finish the process, the third dialog sentence is used when the first two highest similarity scores are very close. Hence, the robot can ask which one it should save. The user interaction helps the robot with the final selection of the concept description. It is worth mentioning that limited interaction is preferable since spending much time teaching one object could be exhausting for the user.

3.6 Ontology Update

The final step in the concept learning process is to update the ontological knowledge with information about the new object. At this point, the correct semantic relation has been chosen according to the guidelines explained in subsection 3.3.2.

The ontology update process starts by creating a new instance of the object. Then, the first concept class created corresponds to the exact name of the object that is being taught; this means that if the new object is a “pen," the first concept class would be *Pen*. Next, the following three hypernyms of the semantic relation will be added sequentially based on the hierarchy. Finally, the creation of new classes stops when (1) the concept already exists in the ontological knowledge or when (2) four class concepts have been created. Figs. 3.8 and 3.9 show an example of these two cases, respectively.

One problem arises in the second case when four classes are created without finding an existing class. In this case, the four classes will not have any precedent class and would be forcefully linked to the root ontology concept. A new concept class linked to the root ontology will not connect with any other concept unless explicitly specified. The robot has to conceptualize new tangible objects available in the current environment in the proposed concept learning scenario. Therefore, the previously mentioned problem is overcome by linking the last class created to the *HumanScaleObject* class, which best describes the possible objects the robot can learn (Fig. 3.9).

3.7 Experiments in Learning a New Concept

We conducted a set of experiments to show the applicability of the presented method for ontology learning of new concepts. Some essential factors are considered in these experiments:

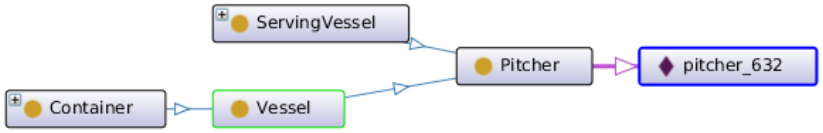


FIGURE 3.8: Concept *pitcher* created in the ontology. The new class *Pitcher* is connected to the existing class *Vessel*.

FIGURE 3.9: Concept *drill* created in the ontology. The new classes *Drill*, *Tool*, *Implement*, and *Instrumentality* are connected to the existing general class *HumanScaleObject*.

- Ontological knowledge enables the robot to progressively and accurately expand its knowledge. It also allows the utilization of more linguistic variations for the same referent.
- Textural knowledge contributes to the understanding of the meanings of a new concept.
- Visual analysis significantly supports the robot in the selection of the meaning of the new object concept.
- Human–robot interaction creates a more natural learning process and assists the robot in the correct conceptualization of objects.

3.7.1 Experiment Setup

The experiments consisted of two parts: concept learning experiments and experiments with an integrated robot system. In the concept learning experiments (Section 3.7.2), the robot was asked to learn new objects as follows (Fig. 3.10):

1. The user showed an image of the new object and named it.
2. The robot conceptualized the new object using textural knowledge and visual analysis.
3. In case more interaction with the user was required, the robot created it.

The concept learning experiments were divided into two tasks. In the first task, the robot learned one object, and the results were compared with a baseline method. In the second task, the robot learned eight different objects with specific characteristics to challenge the robot.

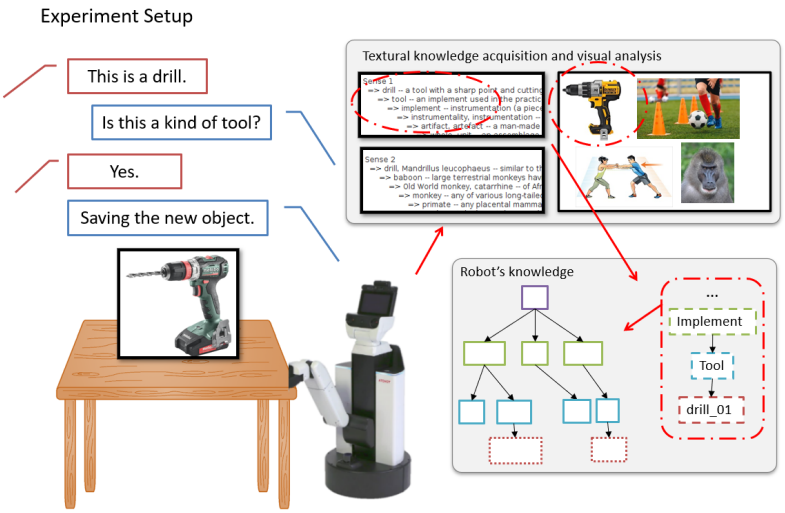


FIGURE 3.10: Layout of experiments in concept learning.

After the robot had finished learning the new objects of the concept learning experiment, we conducted the integrated system experiments (Section 3.7.3), where the robot was asked to search and find the new objects learned.

3.7.2 Concept Learning Experiments

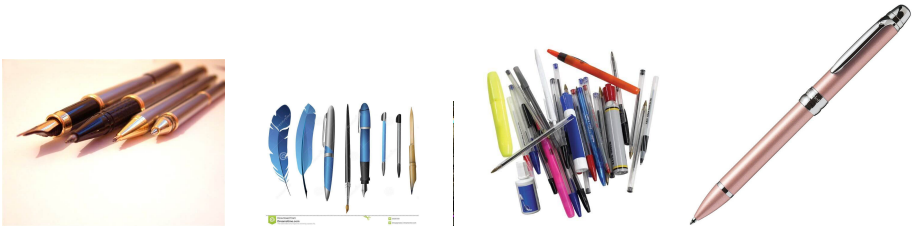
3.7.2.1 Learning an Object with no Interaction

The first experiment consisted of the robot learning only one object concept, and it did not include interaction with the user. This experiment is a continuation of experiments performed in [75]. This experiment aimed to show the importance of visual information to assist in selecting the semantic relation corresponding to the new object concept.

In this experiment, we taught the robot the concept of a “pen." The image and name of the new object were inputted to the robot. The robot started the learning process by acquiring the senses of the word “pen." Then, it made an online image query of the first three hypernyms of each sense. We set the online image query process for this experiment to download the first 10 images found in Google for each search. With this, the robot created sets of images for each hypernym of the semantic relation. Table 3.1 shows 4 sample images for each sense of the concept *Pen* taken from the online image query results.

As a baseline, we used color histograms in the visual analysis process for feature extraction. Hence, color histograms for all sets of images were created. Subsequently, the robot

TABLE 3.1: Sample images of querying the hypernyms of the concept *Pen* per sense used in the concept learning experiments [75].

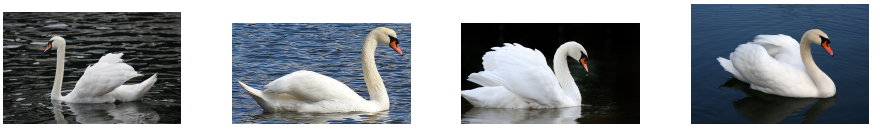


Writing implement

Enclosure

Playpen

Penitentiary



Swan

TABLE 3.2: Similarity scores assigned to each sense of the object “pen" using the baseline and proposed methods.

Word “pen"	**Baseline**	**Our method**
Sense 1. Writing implement with a point from which ink flows.	**0.82**	**0.15**
Sense 2. Enclosure for confining livestock.	0.54	0.11
Sense 3. Portable enclosure in which babies may be left to play.	0.70	0.13
Sense 4. Correctional institution for those convicted of major crimes	0.22	0.09
Sense 5. Female swan.	0.12	0.09

computed the cosine similarity between the histograms of each image searched online and the new object image. Finally, the average similarity was calculated for the set of images of each semantic relation.

The similarity score was assigned to each of the senses. The sense with the highest similarity score was selected to describe the new concept in the ontology. Table 3.2 shows the similarity scores calculated for each sense of the object “pen" using the baseline and our method. Sense 1 has the highest similarity score in both methods; therefore, the robot conceptualized it in the ontology. The robot added the new semantic relations starting from the first concept listed in the hypernyms list of the chosen sense. The new concepts created for the object “pen" according to its Sense 1 (Fig. 3.3 bottom left) are *Pen*, *Writing implement*, *Implement*, and *Instrumentality*. The semantic relations were created on the basis of only the sense description to keep consistency in the terminology used to describe the concepts, and no manual modifications were made. The new concepts are linked to a general concept in the ontology when the ontological knowledge and the sense have no common concepts. Consequently, the robot can conceptually know about the new object.

The similarity scores computed using the baseline and our method shown in Table 3.2 are significantly distant due to the difference in the methods employed in the visual analysis process. While the baseline utilized a color histogram as the feature extraction method, our method used the pre-trained ResNet-152 network. We believe that analyzing the images considering more features identifies them into a category more accurately. Based on these experiments, we

confirmed the importance of selecting the best sense describing the new object in the concept learning process. In addition, the correct placement of the new concepts in the ontology hierarchy is crucial for ensuring that the overall ontological knowledge is consistent.

3.7.2.2 Learning Objects with the Proposed Method

The second experiment consisted of showing the robot eight different objects, which it must conceptualize. In this experiment, the robot used the proposed method for image analysis and user interaction. We demonstrated the challenges in a concept learning scenario involving objects and concepts. In the visual analysis process, the robot used the pretrained network to extract image features. This time, the set of objects included the following types:

- The resulting images between each sense are visually similar.
- Images of the correct sense are significantly different.
- Images of the correct sense appear in the results of another sense.
- Very few images of the correct sense appear in the results.
- Images of the correct sense appear in the results of all senses.
- The image results are visually different for all senses.
- The correct semantic relation contains common concepts with the ontology.
- The correct semantic relation does not contain any common concepts with the ontology.

Fig. 3.11 shows the set of objects used for this second concept learning experiment. It includes a drill, durian, nail, pitcher, sponge, trunk, washer, and wrench. Each of these objects corresponds to at least one of the types of objects listed above.

During the experiment, the robot received the image of each object and its name sequentially. Next, it performed the word meaning identification process, acquiring the corresponding senses and semantic relations, as explained in Section 3.3. Then, it continued with the visual analysis process. In the online image query method, the robot created two datasets: a small dataset of 10 images and a large dataset of 50 images. Next, it extracted features using the pretrained network and computed the similarity scores for each semantic relation, as explained in Section 3.4.

FIGURE 3.11: Objects used in the concept learning experiment. The names of the objects are: *drill*, *durian*, *nail*, *pitcher*, *sponge*, *trunk*, *washer*, and *wrench*.

Table 3.3 illustrates the results of the similarity scores assigned to each sense of the set of objects for the small and large datasets. The senses marked in blue represent the true meaning of the new object concepts to learn. In addition, the highest similarity score for each sense per new object is marked in bold; this corresponds to the sense chosen by the robot.

Based on these results, we can see that the robot correctly chose the desired sense for all objects in both datasets. However, the similarity score is not as high as expected, and this is due to the variety of objects used in the previous experiments. The chosen senses with a low similarity score belong to the object *drill* and the *nail*. The similarity score was significantly affected by the results of the online image query, where the images did not fully represent the expected meaning. In both cases, images similar to the new object appear only in the results of one hypernym of the chosen senses, as shown in Fig. 3.12.

In the case of the objects *durian*, *sponge*, and *wrench*, the highest two similarity scores are slightly close for the small dataset. These results were caused by images of the correct sense appearing in other senses' results, increasing the score of the incorrect sense. However, we believe that having several samples for each sense according to its hypernyms adds more variations of objects helping in the semantic selection. This is evident in the results of *sponge* and *wrench* in the large dataset, where the two highest similarity scores are more distant than the scores for the same objects in the small dataset.

A challenging case was observed in the object *wrench* for the small dataset. In these results, images similar to the target object were found in all senses (Fig. 3.13). This case corresponds to the third situation mentioned in Section 3.5, where a second similarity score is

TABLE 3.3: Similarity scores computed per sense of all objects. The correct sense is marked in blue and the highest similarity score is marked in bold.

Object name	Sense number	Small dataset		Large dataset	
		Baseline	Our method	Baseline	Our method
drill	**Sense 1**	**0.58**	**0.19**	0.53	**0.15**
	Sense 2	0.10	0.09	0.16	0.08
	Sense 3	0.43	0.11	0.50	0.10
	Sense 4	0.46	0.12	**0.54**	0.12
durian	Sense 1	0.09	0.29	0.01	0.26
	Sense 2	**0.14**	**0.35**	**0.06**	**0.29**
nail	Sense 1	**0.73**	0.10	0.56	0.10
	Sense 2	0.68	**0.17**	**0.62**	**0.15**
	Sense 3	0.62	0.10	0.43	0.10
pitcher	Sense 1	0.27	0.09	0.17	0.07
	Sense 2	**0.66**	**0.30**	**0.58**	**0.27**
	Sense 3	0.51	0.18	0.32	0.14
	Sense 4	0.44	0.11	0.22	0.09
	Sense 5	0.08	0.07	0.11	0.07
sponge	**Sense 1**	**0.56**	**0.27**	**0.58**	**0.24**
	Sense 2	0.41	0.21	0.38	0.17
	Sense 3	0.45	0.22	0.34	0.17
	Sense 4	0.03	0.15	0.13	0.14
trunk	Sense 1	0.00	0.09	0.00	0.09
	Sense 2	0.00	**0.23**	0.00	**0.22**
	Sense 3	0.00	0.12	0.00	0.12
	Sense 4	0.00	0.16	0.00	0.14
	Sense 5	**0.01**	0.10	**0.01**	0.09
washer	Sense 1	0.31	0.13	0.51	0.12
	Sense 2	**0.63**	**0.27**	0.54	**0.25**
	Sense 3	0.60	0.14	**0.58**	0.13
wrench	Sense 1	0.52	0.17	0.63	0.16
	Sense 2	0.77	0.24	0.72	0.19
	Sense 3	**0.85**	**0.26**	**0.84**	**0.23**

very close to the highest score. Therefore, the robot proceeded to use the text-based interaction to confirm the meaning of this new object, *wrench*. Thus, it created an alternative question using the first hypernyms of the two optional senses: “Is it a spanner or a twist?" With this question, the user helped the robot to confirm the correct meaning. Then, the robot conceptualized the new object successfully.

Sponge is another object in the small dataset with two close similarity values. However, the robot did not prompt a question with this concept due to the threshold value. Setting up a

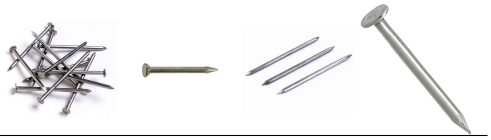
Concept	Hypernym	Sample images
drill	tool	
	implement	
	instrumentality	
nail	device	
	fastener	
	restraint	

FIGURE 3.12: Sample images obtained for the hypernyms of the *drill* and *nail* concepts.

higher threshold might cause unnecessary interaction with the user. Therefore, an appropriate threshold calculation must be investigated in the future.

3.7.3 Integrated System Experiment

The last experiment consisted of the robot searching and finding the new objects in the current environment once it completed the concept learning process. The user challenged the robot skills to deduce the requested object and infer its location according to its knowledge. In this last experiment, we demonstrated the advantage of having ontological knowledge when conceptualizing new objects, such as referring to the new object differently according to its newly connected classes and making inferences about the new objects using possibly inherited attributes.

Wrench	**Sample images**
Sense 1. A sharp strain on muscles or ligaments.	
Sense 2. A jerky pulling movement.	
Sense 3. A hand tool that is used to hold or twist a nut or bolt.	

FIGURE 3.13: Sample images obtained for the senses of the *wrench* concept.

The robot test included the following specifications to examine the integrated system after the concept learning effects:

- The objects are requested by their names or the names of their class.
- When the object class's names are used, they are as follows: a directly stated class name, an inherited class name, and a possibly linked class.
- Object concepts that are expected to inherit attributes such as possible locations are requested.
- The integrated ontology-based knowledge management system with verbal interaction and concept learning was evaluated by its success in solving the object search experiment.

To provide the robot skills to complete the mentioned challenge, we joined the concept learning components (see Fig. 3.2) and the ontology-based knowledge management system for home service robots presented in Chapter 2. The system integration consists of command analysis, talking interaction, task planning, execution modules, and knowledge management connected to the concept learning components.

We conducted the experiments in the simulated environment used in Section 2.4, which includes a four-room house with static furniture and graspable objects commonly found in home settings. We placed boxes at different locations inside the living room, bedroom, lobby, and kitchen, as shown in Fig. 3.14. Each box had one of the images of the new objects that were

FIGURE 3.14: Map showing the locations of the new objects. The pictures highlighted in red represent the new objects.

used during the learning process. The new objects were taught to the robot sequentially during the concept learning. Then, the robot was given different kinds of commands to find the newly acquired objects. The conceptualization of the new concepts was tested using the following natural language commands, which evaluated the correct association of the concepts and the inheritance of features in the ontology:

1. Find the wrench.
2. Find the durian.
3. Find the sponge on the kitchen cabinet.
4. Find the nail in the bedroom.
5. Find the seal (washer).
6. Find the vessel (pitcher).
7. Find the luggage on the bed (trunk).
8. Find the tool in the lobby (drill).

The first four commands request for the new objects by their names. The first two (“Find the wrench" and “Find the durian") do not include a location to search for the object. The third command (“Find the sponge on the kitchen cabinet") includes the name of the furniture. The fourth command (“Find the nail in the bedroom") includes the room’s name to find the object.

TABLE 3.4: Types of commands used for the integrated system experiments. They are divided by commands referring to the object by its name or class. The locations of the objects are divided into non-inherited and inherited if it is not given, and furniture and room if it is given. The number of instances available in the environment for those objects is listed.

Command	**Location not given**		**Location given**		**Number of**
with an object called by	**Non-inherited**	**Inherited**	**Furniture**	**Room**	**instances**
Name:					
1. Find the wrench.	O				1
2. Find the durian.		O			1
3. Find the sponge on the kitchen cabinet.			O		1
4. Find the nail in the bedroom.				O	1
Class:					
5. Find the seal (washer).	O				1
6. Find the vessel (pitcher).		O			1
7. Find the luggage on the bed (trunk).			O		1
8. Find the tool in the lobby (drill).				O	more than 1

The last four commands request for the new objects using the name of an upper class and two do not include a location to search for the objects. In the fifth command (“Find the seal"), the class name used refers to the object “washer," and the sixth command (“Find the vessel") refers to the object “pitcher." The seventh command (“Find the luggage on the bed") includes the furniture on which the object is located, and the class name refers to the object “trunk." Finally, the eighth command (“Find the tool in the lobby") includes the room’s name, and the class name refers to the “drill."

In addition, the second and sixth commands (“Find the durian" and “Find the vessel") include objects that are expected to inherit a location where are commonly found from its ontological knowledge. A particular case is considered in the eighth command (“Find the tool in the lobby"), where more than one object belonging to the required class exists in the environment. A summary of the characteristics of each command can be found in Table 3.4.

The robot completed all the commands using the integrated system successfully during the experiment. The robot used its ontological knowledge and interacted with the user when

TABLE 3.5: Experimental results using the integrated system. The robot found the instances of the requested objects using its knowledge or by asking the user. The locations were acquired using its knowledge or from the command, or by asking the user.

Command	**Object**		**Location**		**Succeeded**
	In knowledge	**Asked user**	**In knowledge /command**	**Asked user**	
1. Find the wrench.	O			O	Yes
2. Find the durian.	O		O		Yes
3. Find the sponge on the kitchen cabinet.	O		O		Yes
4. Find the nail in the bedroom.	O			O	Yes
5. Find the seal (washer).	O			O	Yes
6. Find the vessel (pitcher).	O		O		Yes
7. Find the luggage on the bed (trunk).	O		O		Yes
8. Find the tool in the lobby (drill).		O		O	Yes

necessary. We analyzed the robot's process in each command to evaluate the use of its ontological knowledge after the learning process. The results can be found in Table 3.5. The robot found object instances applicable to the description when the objects were requested either by their names or classes. Furthermore, the robot found a suitable instance of a location to search for the object when its ontological knowledge had an inherited location or when the command provided it.

A particular case was observed when the commands included any object belonging to the *Tool* class, which are "wrench," "nail," "washer," and the "drill." This type of class does not include an inherited location that could be inferred from the robot's knowledge; therefore, the robot could not find a suitable location. However, since the integrated system includes human–robot interaction, the robot could ask the user for an object's location.

Fig. 3.15 shows the dialogs that occurred during this interaction. The robot asked the object's location for commands 1 and 4 with the questions: "Where could I find the wrench?" and "In which piece of furniture can I find the nail?", respectively. However, the question formulated differs because, in command 4, the name of a room is given *bedroom*, causing the

Dialog command 1:

Human: “Find the wrench."
Robot: “Where could I find the wrench?"
Human: “It is on the sideboard."

Dialog command 4:

Human: “Find the nail in the bedroom."
Robot: “In which piece of furniture can I find the nail?"
Human: “It is on the desk."

Dialog command 5:

Human: “Find the seal."
Robot: “Where could I find the washer?"
Human: “It is on the chest."

Dialog command 8:

Human: “Find the tool in the lobby."
Robot: “Which tool do you want?"
Human: “The drill."
Robot: “In which piece of furniture can I find the drill?"
Human: “On the armchair."

FIGURE 3.15: Illustration of the dialogs between the robot and the user when receiving a command to find a newly learned object. The commands that are shown required objects corresponding to the *Tool* category.

robot to require a more specific place inside the room to search for the object since no inherited location fulfilling these characteristics could be found in the knowledge.

As for the dialog generated for command 5, the user required a seal which is a category for the newly learned object washer, and no other instance of objects of the seal category exists. Consequently, the robot proceeded to ask the question: “Where could I find the washer?". Another case occurred in the last command (“Find the tool in the lobby"), where the robot found more than one object that could belong to the *Tool* class, and it asked the user for the specific object with the question: “Which tool do you want?". Next, similarly to command 4, the robot could not find a location for the object *drill* in the room *lobby*, resulting in asking for the specific location: “In which piece of furniture can I find the drill?".

We can conclude that the new objects learned by the robot were accurately conceptualized in the ontological knowledge, considering that the robot could find the possible instances of the

requested objects. Furthermore, the new concepts learned for each new object were linked to the proper hierarchy since the robot found inherited properties such as the location. In cases, where the ontological knowledge was insufficient, the system used human–robot interaction to acquire information, which is the expected behavior of a service robot.

3.8 Discussion and Summary

This chapter described ontology learning of new concepts combining textural knowledge, visual analysis, and user interaction for service robot applications. We proposed analyzing the semantic relations and visual features of a new object concept to determine its correct conceptualization in an ontology. In addition, we employed human–robot interaction to assist the robot when necessary.

We tested the proposed concept learning method in a scenario, where the robot has to conceptualize new objects with only the image and the name of the object. In addition, we demonstrated some challenges to consider in concept learning scenarios. Future work could make some improvements, such as improving the input query formulation for the online image query process to obtain more accurate image results according to each sense. This improvement could be achieved by further analyzing concept meanings and the semantic relations of a concept since more information defining each sense can be obtained.

Furthermore, the interaction with users during the word meaning selection process could be improved by including more variations of questions formulated by the robot, such as inquiring about the new object and mentioning its possible applications. Improving the verbal interaction would be helpful too during the ontology update process, as there is a possibility of concept inconsistency. Ontology inconsistencies occurs when a concept is not adequately defined, including missing equivalent classes, disjoint classes, and other properties. Hence, a more extensive verbal interaction would be necessary.

Chapter 4

Conclusions and Future Work

4.1 Conclusions

Service robots at home convey the possibility of having assistance and companionship to older adults or disabled humans living independently. However, dealing with people requires specific skills for a robot as human contact and changing environment are inevitable. Functionalities as finding objects, interacting with humans, or learning new concepts are desired features in a service robot. An essential factor in enabling such functionalities is providing the robot with knowledge about the environment. Knowledge represented using ontologies describes the conceptualization of entities of the real world in a structured form. Using ontologies offers the robot the option of making inferences in its knowledge and gradually incrementing its concepts. However, it is important to handle or prevent inconsistencies.

This research aims to develop a fully autonomous system for service robots to help people at home, giving the robot knowledge of the environment, reasoning, natural language interaction, and incremental concept learning. To achieve this, we have successfully built an ontology-knowledge management system for service robots. It combines ontological knowledge reasoning and human–robot interaction to interpret natural language commands. The system allows the robot to disambiguate uncertain requests through spoken interaction. It also utilizes information stated in the ontology to create more precise questions. The system contains modules to manage the robot's knowledge and reasoning, command analysis, decision-making, and talking interaction. These modules are interconnected and share information between themselves in different steps of the processes.

We introduced a second essential component of this system that is concept learning. This component uses textural knowledge to identify the possible meaning of a new word that will be added to the ontological knowledge. Furthermore, it supports the meaning's selection by analyzing the visual characteristics of related concepts against the new object. Finally, it updates the ontological knowledge incrementally as needed.

We conducted separate experiments inside a simulated environment to analyze the behavior of a robot using our system. The experiments included completing tasks given a natural language command and the conceptualization of new objects given an image and an object name.

4.2 Future Work

Some improvements could be achieved in future work mentioned in previous chapters, such as the interpretation of natural language statements with entirely new entities. The concepts represented in the ontology can be easily extended with more home-related concepts or a new environment definition. The linguistic representation of verbs associated with the current action class description can also be extended.

The concept learning process could be improved by changing the input query formulation for the online image query process to obtain more accurate image results according to each sense. Furthermore, the interaction with the user in the word meaning selection could be enhanced by including more variations of questions formulated by the robot, such as asking about the new object mentioning its possible usage.

Although the current system tries to prevent inconsistencies in ontological knowledge before adding a new concept, it does not include dealing with them if found in the ontology. We believe that a more extensive verbal interaction would be necessary to deal with inconsistencies appearing in the ontology.

We did not approach some challenges in this research as we focused on the ontological knowledge-based system development and functionalities, such as the vision and navigation features. The vision module needs to be upgraded with a method that can either be retrained fast or learn visual objects incrementally. Likewise, the navigation module needs to be extended with functionalities to explore the environment and learn locations.

Multifunctionality in a service robot is greatly desired as one robot must help humans with numerous tasks at home. However, this is highly restricted by the hardware components of

the robot. This study used the HSR robot to test the proposed system, which limits the number of objects that can be manipulated simultaneously since the robot has only one arm. Also, if using a real robot, the weight of the objects affects the grasping motion. By employing a different robot, e.g., a two-handed robot, more functionalities can be implemented to carry out multiple or more complex tasks.

The proposed approach for service robots is currently meant to be applied in home settings scenarios where a human needs help. However, it can be easily transferred to a different type of scenario as long as it is required to handle objects and move them to different places, such as a convenience store where the human–robot interaction could simplify the collaboration process. One final application, although ambitious, is the deployment of this kind of system in service robots helping children, for instance, in a childcare facility where a wide variety of objects must be organized every day, such as toys, books, stationery, handicrafts supplies, and more.

Bibliography

[1] S. Nielsen, E. Bonnerup, A. Hansen, J. Nilsson, L. Nellemann, K. Hansen, and D. Hammershøi, "Subjective experience of interacting with a social robot at a danish airport," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, ser. IEEE RO-MAN proceedings. United States: IEEE, 11 2018, pp. 1163 – 1170.

[2] B. Bruno, C. T. Recchiuto, I. Papadopoulos, A. Saffiotti, C. Koulouglioti, R. Menicatti, F. Mastrogiovanni, R. Zaccaria, and A. Sgorbissa, "Knowledge representation for culturally competent personal robots: Requirements, design principles, implementation, and assessment," *International Journal of Social Robotics*, vol. 11, no. 3, pp. 515–538, Jun 2019.

[3] H. M. R. T. Bandara, M. A. V. J. Muthugala, A. G. B. P. Jayasekara, and D. P. Chandima, "Grounding object attributes through interactive discussion for building cognitive maps in service robots," in *IEEE International Conference on Systems, Man, and Cybernetics, SMC 2018, Miyazaki, Japan, October 7-10, 2018*, 2018, pp. 3775–3780.

[4] C. Li, G. Tian, and H. Chen, "The introduction of ontology model based on sso design pattern to the intelligent space for home service robots," in *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec 2016, pp. 1673–1678.

[5] H. Jeon, K. Yang, S. Park, J. Choi, and Y. Lim, "An ontology-based home care service robot for persons with dementia," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2018, pp. 540–545.

[6] S. Chandra, R. Paradeda, H. Yin, P. Dillenbourg, R. Prada, and A. Paiva, "Do children perceive whether a robotic peer is learning or not?" in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '18. New York, NY, USA: ACM, 2018, pp. 41–49.

[7] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan, "Learning from physical human corrections, one feature at a time," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '18. New York, NY, USA: ACM, 2018, pp. 141–149.

[8] A. Angleraud, Q. Houbre, V. Kyrki, and R. Pieters, "Human-robot interactive learning architecture using ontologies and symbol manipulation," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2018, pp. 384–389.

[9] D. P. Losey and M. K. O'Malley, "Enabling robots to infer how end-users teach and learn through human-robot interaction," *IEEE Robotics and Automation Letters*, vol. 4, pp. 1956–1963, 2019.

[10] T. Horie and K. Takashio, "Handling conversation interruption in many-to-many hr interaction considering emotional behaviors and human relationships," in *RO-MAN 2018 - 27th IEEE International Symposium on Robot and Human Interactive Communication*. Institute of Electrical and Electronics Engineers Inc., 11 2018, pp. 528–533.

[11] S. Patki, A. F. Daniele, M. R. Walter, and T. M. Howard, "Inferring compact representations for efficient natural language understanding of robot instructions," 2019.

[12] R. Liu and X. Zhang, "A review of methodologies for natural-language-facilitated human–robot cooperation," *International Journal of Advanced Robotic Systems*, vol. 16, no. 3, p. 1729881419851402, 2019. [Online]. Available: https://doi.org/10.1177/1729881419851402

[13] M. Serna and A. Serna, "Ontology for knowledge management in software maintenance," *International Journal of Information Management*, vol. 34, pp. 704–710, 2014.

[14] A. Eckhard, I. Navas Delgado, and J. Aldana Montes, "Towards an ontology for enterprise knowledge management," in *SEMAPRO International Conference on Advances in Semantic Processing*, Nov 2011, pp. 75–80.

[15] F. Ali, D. Kwak, P. Khan, S. H. A. Ei-Sappagh, S. M. R. Islam, D. Park, and K. Kwak, "Merged ontology and svm-based information extraction and recommendation system for social robots," *IEEE Access*, vol. 5, pp. 12 364–12 379, 2017.

[16] J. Zhang, W. Zhao, G. Xie, and H. Chen, "Ontology- based knowledge management system and application," *Procedia Engineering*, vol. 15, pp. 1021 – 1029, 2011, cEIS 2011.

[17] J. I. Olszewska, M. Barreto, J. Bermejo-Alonso, J. Carbonera, A. Chibani, S. Fiorini, P. Goncalves, M. Habib, A. Khamis, A. Olivares, E. P. de Freitas, E. Prestes, S. V. Ragavan, S. Redfield, R. Sanz, B. Spencer, and H. Li, "Ontology for autonomous robotics," in *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2017, pp. 189–194.

[18] D. Beßler, M. Pomarlan, and M. Beetz, "Owl-enabled assembly planning for robotic agents," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '18. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 1684–1692.

[19] G. A. G. Ricardez, Y. Osaki, M. Ding, J. Takamatsu, and T. Ogasawara, "Estimating the operation of unknown appliances for service robots using cnn and ontology," *2018 Second IEEE International Conference on Robotic Computing (IRC)*, pp. 181–182, 2018.

[20] B. Fan, J. Ma, N. Jiang, H. Dogan, and R. Ali, "A rule based reasoning system for initiating passive adas warnings without driving distraction through an ontological approach," *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 3511–3517, 2018.

[21] M. Tenorth and M. Beetz, "Knowrob: A knowledge processing infrastructure for cognition-enabled robots," *The International Journal of Robotics Research*, vol. 32, no. 5, pp. 566–590, 2013. [Online]. Available: https://doi.org/10.1177/0278364913481635

[22] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mösenlechner, D. Pangercic, T. Rühr, and M. Tenorth, "Robotic roommates making pancakes," in *2011 11th IEEE-RAS International Conference on Humanoid Robots*, 2011, pp. 529–536.

[23] M. Beetz, M. Tenorth, and J. Winkler, "Open-ease," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 1983–1990.

[24] M. Tenorth, D. Nyga, and M. Beetz, "Understanding and executing instructions for everyday manipulation tasks from the world wide web," in *2010 IEEE International Conference on Robotics and Automation*, 2010, pp. 1486–1491.

[25] M. Beetz, D. Beßler, A. Haidu, M. Pomarlan, A. Bozcuoglu, and G. Bartels, "Know rob 2.0 — a 2nd generation knowledge processing framework for cognition-enabled robotic agents," 05 2018, pp. 512–519.

[26] A. Ramachandran, C.-M. Huang, E. Gartland, and B. Scassellati, "Thinking aloud with a tutoring robot to enhance learning," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '18. New York, NY, USA: ACM, 2018, pp. 59–68.

[27] J. de Wit, T. Schodde, B. Willemsen, K. Bergmann, M. de Haas, S. Kopp, E. Krahmer, and P. Vogt, "The effect of a robot's gestures and adaptive tutoring on children's acquisition of second language vocabularies," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI '18. New York, NY, USA: ACM, 2018, pp. 50–58.

[28] Z. Materna, M. Kapinus, V. Beran, P. Smrž, and P. Zemčík, "Interactive spatial augmented reality in collaborative robot programming: User experience evaluation," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2018, pp. 80–87.

[29] Y. Cheng, Y. Shi, Z. Sun, D. Feng, and L. Dong, "An interactive scene generation using natural language," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 6957–6963.

[30] B. Galitsky, D. Ilvovsky, and E. Goncharova, "On a chatbot conducting dialogue-in-dialogue," in *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*. Stockholm, Sweden: Association for Computational Linguistics, Sep. 2019, pp. 118–121.

[31] M. Abrams, L. Gessler, and M. Marge, "B. rex: a dialogue agent for book recommendations," in *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*. Stockholm, Sweden: Association for Computational Linguistics, Sep. 2019, pp. 418–421.

[32] M. J. Chung and M. Cakmak, ""how was your stay?": Exploring the use of robots for gathering customer feedback in the hospitality industry," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2018, pp. 947–954.

[33] L. Shu, P. Molino, M. Namazifar, H. Xu, B. Liu, H. Zheng, and G. Tür, "Flexibly-structured model for task-oriented dialogues," *ArXiv*, vol. abs/1908.02402, 2019.

[34] H. Sugiyama, T. Meguro, Y. Yoshikawa, and J. Yamato, "Improving dialogue continuity using inter-robot interaction," in *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2018, pp. 105–112.

[35] N. Axelsson and G. Skantze, "Modelling adaptive presentations in human-robot interaction using behaviour trees," in *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*. Stockholm, Sweden: Association for Computational Linguistics, Sep. 2019, pp. 345–352.

[36] E. Sibirtseva, D. Kontogiorgos, O. Nykvist, H. Karaoguz, I. Leite, J. Gustafson, and D. Kragic, "A comparison of visualisation methods for disambiguating verbal requests in human-robot interaction," *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, Aug 2018.

[37] K. Wolfel and D. Henrich, "Grounding verbs for tool-dependent, sensor-based robot tasks," *2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pp. 378–383, 2018.

[38] J. Thomason, A. Padmakumar, J. Sinapov, N. Walker, Y. Jiang, H. Yedidsion, J. Hart, P. Stone, and R. J. Mooney, "Improving grounded natural language understanding through human-robot dialog," *2019 International Conference on Robotics and Automation (ICRA)*, May 2019.

[39] E. Bastianelli, G. Castellucci, D. Croce, and R. Basili, "Textual inference and meaning representation in human robot interaction," in *Proceedings of the Joint Symposium on Semantic Processing. Textual Inference and Structures in Corpora*, Trento, Italy, Nov. 2013, pp. 65–69. [Online]. Available: https://www.aclweb.org/anthology/W13-3820

[40] E. Bastianelli, D. Croce, A. Vanzo, R. Basili, and D. Nardi, "A discriminative approach to grounded spoken language understanding in interactive robotics," in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, ser. IJCAI'16. AAAI Press, 2016, p. 2747–2753.

[41] J. Bos and T. Oka, "A spoken language interface with a mobile robot," *Artificial Life and Robotics*, vol. 11, pp. 42–47, 01 2007.

[42] D. K. Misra, J. Sung, K. Lee, and A. Saxena, "Tell me dave: Context-sensitive grounding of natural language to manipulation instructions," *The International Journal of Robotics Research*, vol. 35, no. 1-3, pp. 281–300, 2016. [Online]. Available: https://doi.org/10.1177/0278364915602060

[43] X. Jiang and A.-H. Tan, "Crctol: A semantic-based domain ontology learning system," *Journal of the American Society for Information Science and Technology*, vol. 61, no. 1,

pp. 150–168, 2010. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/asi.21231

[44] L. Karoui, M.-A. Aufaure, and N. Bennacer, "Contextual concept discovery algorithm," in *FLAIRS Conference*, 2007.

[45] E. Drymonas, K. Zervanou, and E. G. M. Petrakis, "Unsupervised ontology acquisition from plain texts: The ontogain system," in *Natural Language Processing and Information Systems*, C. J. Hopfe, Y. Rezgui, E. Métais, A. Preece, and H. Li, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 277–287.

[46] B. Frikh, A. Djaanfar, and B. Ouhbi, "A hybrid method for domain ontology construction from the web," pp. 285–292, 01 2011.

[47] S. Chuprina, V. Alexandrov, and N. Alexandrov, "Using ontology engineering methods to improve computer science and data science skills," *Procedia Computer Science*, vol. 80, pp. 1780–1790, 2016, international Conference on Computational Science 2016, ICCS 2016, 6-8 June 2016, San Diego, California, USA. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050916309310

[48] A. Salatino, T. Thanapalasingam, A. Mannocci, F. Osborne, and E. Motta, *The Computer Science Ontology: A Large-Scale Taxonomy of Research Areas*, 01 2018, pp. 187–205.

[49] Y. Xu, D. Rajpathak, I. Gibbs, and D. Klabjan, "Automatic ontology learning from domain-specific short unstructured text data," 2019.

[50] R. Jackson, J. Balhoff, E. Douglass, N. Harris, C. Mungall, and J. Overton, "Robot: A tool for automating ontology workflows," *BMC Bioinformatics*, vol. 20, 07 2019.

[51] B. Smith, M. Ashburner, C. Rosse, J. Bard, W. Bug, W. Ceusters, L. Goldberg, K. Eilbeck, A. Ireland, C. Mungall, N. Leontis, P. Rocca-Serra, A. Ruttenberg, S.-A. Sansone, R. Scheuermann, N. Shah, P. Whetzel, and S. Lewis, "The obo foundry: coordinated evolution of ontologies to support biomedical data integration," *Nature Biotechnology*, vol. 25, pp. 1251–1255, 2007.

[52] J. I. Olszewska, M. Barreto, J. Bermejo-Alonso, J. Carbonera, A. Chibani, S. Fiorini, P. Goncalves, M. Habib, A. Khamis, A. Olivares, E. P. de Freitas, E. Prestes, S. V. Ragavan, S. Redfield, R. Sanz, B. Spencer, and H. Li, "Ontology for autonomous robotics," in *2017 26th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, 2017, pp. 189–194.

[53] L. Grassi, C. T. Recchiuto, and A. Sgorbissa, "Knowledge triggering, extraction and storage via human-robot verbal interaction," *CoRR*, vol. abs/2104.11170, 2021. [Online]. Available: https://arxiv.org/abs/2104.11170

[54] K. Miyazawa, T. Horii, T. Aoki, and T. Nagai, "Integrated cognitive architecture for robot learning of action and language," *Frontiers in Robotics and AI*, vol. 6, p. 131, 2019. [Online]. Available: https://www.frontiersin.org/article/10.3389/frobt.2019.00131

[55] T. Nakamura, T. Araki, T. Nagai, and N. Iwahashi, "Grounding of word meanings in latent dirichlet allocation-based multimodal concepts," *Advanced Robotics*, vol. 25, pp. 2189 – 2206, 2011.

[56] T. Araki, T. Nakamura, T. Nagai, K. Funakoshi, M. Nakano, and N. Iwahashi, "Online object categorization using multimodal information autonomously acquired by a mobile robot," *Advanced Robotics*, vol. 26, no. 17, pp. 1995–2020, 2012. [Online]. Available: https://doi.org/10.1080/01691864.2012.728693

[57] S. Arora, C. Cho, P. Fitzpatrick, and F. Scharffe, "Towards ontology driven learning of visual concept detectors," *CoRR*, vol. abs/1605.09757, 2016. [Online]. Available: http://arxiv.org/abs/1605.09757

[58] A. Ayub and A. R. Wagner, "Tell me what this is: Few-shot incremental object learning by a robot," *CoRR*, vol. abs/2008.00819, 2020. [Online]. Available: https://arxiv.org/abs/2008.00819

[59] J. Wątróbski, "Ontology learning methods from text - an extensive knowledge-based approach," *Procedia Computer Science*, vol. 176, pp. 3356–3368, 2020, knowledge-Based and Intelligent Information I& Engineering Systems: Proceedings of the 24th International Conference KES2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050920319566

[60] M. Horridge, N. Drummond, J. Goodwin, A. L. Rector, R. Stevens, and H. H. Wang, "The manchester owl syntax," in *OWLED*, 2006.

[61] *Knowledge Representation in Description Logic*. London: Springer London, 2007, pp. 35–55. [Online]. Available: https://doi.org/10.1007/978-1-84628-710-7{_}3

[62] S. Lee and I. Kim, "A robotic context query-processing framework based on spatio-temporal context ontology," *Sensors*, vol. 18, no. 10, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/10/3336

[63] S. Abburu, "Article: A survey on ontology reasoners and comparison," *International Journal of Computer Applications*, vol. 57, no. 17, pp. 33–39, November 2012, full text available.

[64] "The owl api," http://owlcs.github.io/owlapi/, accessed: 2019-12-16.

[65] *Words, Parts of Speech, and Morphology*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 113–145.

[66] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, ser. NAACL '03. Stroudsburg, PA, USA: Association for Computational Linguistics, 2003, pp. 173–180. [Online]. Available: https://doi.org/10.3115/1073445.1073478

[67] D. Chen and C. D. Manning, "A fast and accurate dependency parser using neural networks," in *EMNLP*, 2014.

[68] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Association for Computational Linguistics (ACL) System Demonstrations*, 2014, pp. 55–60. [Online]. Available: http://www.aclweb.org/anthology/P/P14/P14-5010

[69] R. K. Larson, "On the double object construction," *Linguistic Inquiry*, vol. 19, no. 3, pp. 335–391, 1988. [Online]. Available: http://www.jstor.org/stable/25164901

[70] M. M. Berg, A. Isard, and J. D. Moore, "An openccg-based approach to question generation from concepts," in *Natural Language Processing and Information Systems*, E. Métais, F. Meziane, M. Saraee, V. Sugumaran, and S. Vadera, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 38–52.

[71] T. Yamamoto, K. Terada, A. Ochiai, F. Saito, Y. Asahara, and K. Murase, "Development of the research platform of a domestic mobile manipulator utilized for international competition and field test," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7675–7682.

[72] T. Wisspeintner, T. Zant, L. Iocchi, and S. Schiffer, "Robocup@home: Scientific competition and benchmarking for domestic service robots," *Interaction Studies*, vol. 10, pp. 392–426, 12 2009.

[73] D. Holz, L. Iocchi, and T. Zant, "Benchmarking intelligent service robots through scientific competitions: the robocup@home approach." 01 2013.

[74] M. Basiri, E. Piazza, M. Matteucci, and P. Lima, "Benchmarking functionalities of domestic service robots through scientific competitions," *KI - Künstliche Intelligenz*, vol. 33, 09 2019.

[75] L. V. Gómez and J. Miura, "Ontology learning of new concepts combining textural knowledge and visual analysis for service robot applications," in *2021 IEEE International Conference on Intelligence and Safety for Robotics (ISR)*, 2021, pp. 98–100.

[76] G. A. Miller, "Wordnet: A lexical database for english," *Commun. ACM*, vol. 38, no. 11, p. 39–41, Nov. 1995. [Online]. Available: https://doi.org/10.1145/219717.219748

[77] H. Cunningham, V. Tablan, A. Roberts, and K. Bontcheva, "Getting more out of biomedical documents with gate's full lifecycle open source text analytics," *PLOS Computational Biology*, vol. 9, no. 2, pp. 1–16, 02 2013. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1002854

[78] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," 2016.

[79] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

List of Publications

Journals

1. L. Villamar Gómez and J. Miura, Ontology-based knowledge management with verbal interaction for command interpretation and execution by home service robots, Robotics and Autonomous Systems, Volume 140, June 2021, doi: 10.1016/j.robot.2021.103763.

2. L. Villamar Gómez and J. Miura, Ontology learning of new concepts combining textural knowledge, visual analysis, and user interaction, IEEE Access, Volume 9, pp. 146023-146037, 2021, doi: 10.1109/ACCESS.2021.3122295.

Conferences

1. L. Villamar Gómez and J. Miura, Ontology learning of new concepts combining textural knowledge and visual analysis for service robot applications, Proceedings of 2021 IEEE International Conference on Intelligence and Safety for Robotics (ISR), pp. 98–100, Nagoya, Japan, March 2021, doi: 10.1109/ISR50024.2021.9419551.