Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

A novel framework to improve motion planning of robotic systems through semantic knowledge-based reasoning

Rodrigo Bernardo [a,b,*], João M.C. Sousa [a], Paulo J.S. Gonçalves [a,b]

[a] IDMEC, Instituto Superior Técnico, Universidade de Lisboa, Portugal
[b] IDMEC, Instituto Politécnico de Castelo Branco, Castelo Branco, Portugal

ARTICLE INFO

Keywords:
Knowledge representation
Ontologies
Manipulation
Motion planning
Semantic maps

ABSTRACT

The need to improve motion planning techniques for manipulator robots, and new effective strategies to manipulate different objects to perform more complex tasks, is crucial for various real-world applications where robots cooperate with humans. This paper proposes a novel framework that aims to improve the motion planning of a robotic agent (a manipulator robot) through semantic knowledge-based reasoning. The Semantic Web Rule Language (SWRL) was used to infer new knowledge based on the known environment and the robotic system. Ontological knowledge, e.g., semantic maps, were generated through a deep neural network, trained to detect and classify objects in the environment where the robotic agent performs. Manipulation constraints were deduced, and the environment corresponding to the agent's manipulation workspace was created so the planner could interpret it to generate a collision-free path. For reasoning with the ontology, different SPARQL queries were used. The proposed framework was implemented and validated in a real experimental setup, using the planning framework ROSPlan to perform the planning tasks. The proposed framework proved to be a promising strategy to improve motion planning of robotics systems, showing the benefits of artificial intelligence, for knowledge representation and reasoning in robotics.

1. Introduction

Within the ongoing advances of Industry 5.0, robotic systems are increasingly present in highly dynamic environments, including environments shared with humans (Nahavandi, 2019). The need to find efficient paths (motion planning) for manipulator robots, and new effective strategies to manipulate different objects in order to perform more complex tasks, is crucial for various real-world applications. Motion planning for robotic manipulators consists of defining a continuous path that connects a given initial state of a robotic system to a shared goal region for that system. The path satisfies constraints (e.g. collision avoidance, bounded forces, bounded acceleration) (Latombe, 2012). Traditionally, there are two approaches to the problem: offline planning, which assumes a perfectly known and stable environment, and online planning, which focuses on dealing with environmental uncertainties. The majority of motion planning algorithms are based on random sampling algorithms (Karaman & Frazzoli, 2011), such as Probabilistic RoadMaps (PRMs) (Hsu, Latombe, & Kurniawati, 2006; Siméon, Laumond, Cortés, & Sahbani, 2004) and Rapidly-exploring Random Trees (RRTs) (LaValle et al., 1998; Rodriguez, Tang, Lien, & Amato, 2006). Recently, algorithms such as optimization-based (Salzman & Halperin, 2016), Probabilistic Movement Primitives (ProMPs)-based (Gomez-Gonzalez, Neumann, Schölkopf, & Peters, 2020; Paraschos, Daniel, Peters, & Neumann, 2018) and physics-based methods (Kitaev, Mordatch, Patil, & Abbeel, 2015; Moll, Kavraki, Rosell, et al., 2017), have been shown to be more effective (Liu & Liu, 2021).

Human–robot collaboration (HRC) has been emerging in the manufacturing domain (Lasota, Fong, Shah, et al., 2017). Collaborative robots usually have the advantages of high safety, good adaptability to the environment, and human–computer solid interaction capabilities (Gualtieri, Rauch, & Vidoni, 2022; Polverini, Zanchettin, & Rocco, 2017). To achieve better collaboration, robots must be able to perceive and analyse information holistically from the working environment to plan and act accordingly (Fan, Zheng, & Li, 2022) proactively. The emergence of collaborative robots has increased the difficulty of motion planning.

Knowledge representation and reasoning have become promising fields in artificial intelligence (Olszewska et al., 2017; Pignaton de Freitas et al., 2020). Ontology-based knowledge representation and reasoning techniques provide sophisticated knowledge about the environment for processing tasks or methods. Ontologies emerge as an

* Correspondence to: Av. Empresário, 6000-767 Castelo Branco, Portugal.
E-mail address: rodrigo.f.bernardo@tecnico.ulisboa.pt (R. Bernardo).

https://doi.org/10.1016/j.cie.2023.109345

Available online 27 May 2023

excellent way to describe manipulation actions and facilitate the reasoning process in path planning, and the representation of knowledge about the environment to make robots more autonomous (Bernardo, Sousa, & Gonçalves, 2021, 2022; Feyzabadi & Carpin, 2014; Olszewska et al., 2017). Knowledge reasoning techniques can infer new conclusions and thus help to plan dynamically in a non-deterministic environment. Semantic Web Rule Language (SWRL)-rules are increasingly becoming an important form of knowledge representation on the Semantic Web (Mun & Ramani, 2011).

The main goal of this paper is to develop a framework where motion planning is improved, allowing the possibility to reconfigure the initially defined plan (e.g. recovery from a situation where an unexpected obstacle appears on the path), for a robotics manipulative agent. Through semantic knowledge and reasoning (about manipulation actions and the objects present in the environment), i.e., the inferred knowledge should be used to fine-tune motion planning. SWRL-rules[1] are used to infer knowledge. A deep neural network was trained for the detection and classification of the objects present in the environment where the robotic agent is located was used. Based on this information, a semantic map (described in an ontology) of the environment was created. The proposed domain ontology is based on the CORA (Robotics & Society, 2015) (Core Ontology for Robotics and Automation) and AuR (Goncalves et al., 2021) (Autonomous Robotic) ontologies. It has as a fundamental layer the top-level ontology DOLCE (Descriptive Ontology for Linguistic and Cognitive Engineering), to follow the interpretation of some relevant concepts and thus define an ontology (domain) with a high level of flexibility (Borgo & Masolo, 2010). Experimental validation is performed in a simple environment of a house, based on a smart-home environment. In order to control the robotic manipulator (UR3 from Universal Robots), and the gripper (2f-140 from Robotiq), two toolboxes were developed, one for the robotic arm (toolbox_ur3) and another for the gripper (toolbox_gripper), these were developed to work on the robot operating system (ROS[2]). Finally, knowledge was inferred based on semantic knowledge, and the ROSPlan[3] framework was used to perform task planning based on the actions defined in the ontology.

This paper is organized as follows: The next section provides an overview of related work and presents the research gaps. Section 3 introduces the ontological framework proposed here. Section 4 reports the results of the implementation of the different components of the framework. Section 5 presents an example of a practical validation of the proposed framework in a real environment. Finally, Section 6 provides the work's conclusions and challenges/futures.

2. Literature review

2.1. Ontologies for robotic systems

Ontologies are a powerful solution for acquiring and sharing common knowledge (Garg et al., 2020; Wang, Wong, & Wang, 2010). Consist of the formal conceptualization of knowledge representation and provide the definitions of the concepts and relationships of a given domain of knowledge as a set of concepts and the relationships among those concepts. "A domain ontology does not aim to list all concepts in a domain exhaustively, but rather to build an abstract (yet extendable) philosophical (yet practical) conceptualization of the essence of knowledge in a domain". (El-Diraby, 2013). Ontologies can be encoded using the Web Ontology Language (OWL), the most recent development in ontology languages from the World Wide Web Consortium (W3C[4]). OWL is a set of languages for authoring ontologies for formal knowledge representation; There are three versions of OWL, each with different degrees of expressiveness, OWL Lite, OWL DL (Description Logic), and OWL Full (Horrocks & Patel-Schneider, 2003).

Different types of ontology can be defined depending on the specific application needs. Guarino proposes a classification based on levels of generality (Guarino, 1998) and defines four classes: (i) *Top-level or Upper ontologies* describe very general concepts like e.g., space, time, event or action, that are independent from a particular problem or domain; (ii) *Domain ontologies* describe general concepts related to a specific domain; (iii) *Task ontologies* describe generic tasks or activities. (iv) *Application ontologies* characterize a specific application and describe concepts whose relevance is limited to a specific domain and task.

The CORA is an ontology developed by the IEEE-RAS Ontologies for Robotics and Automation Working Group (IEEE ORA WG) in 2015 to map and represent the most general concepts and axioms in the Robotics and Automation (R&A) domain (Prestes et al., 2013). It is part of the IEEE Standard 1872–2015 (Robotics & Society, 2015). However, extra effort is needed to review and adapt the definition of some key concepts of CORA, as it is based on the upper ontology SUMO (Suggested Upper Merged Ontology) (Niles & Pease, 2001).

Although similar, the upper ontologies cannot be directly integrated without introducing contradictions. DOLCE is an ontology of particulars. It does have universal (classes and properties), but the claim is that they are only employed in the service of describing particulars. In contrast, SUMO could be described as an ontology of both particulars and universals (Mascardi, Cordì, & Rosso, 2008). Also, DOLCE uses meta-properties as a guiding methodology, while SUMO pursues a formal definition of such meta-properties directly in the ontology itself (axiomatization) (Mascardi et al., 2008; Trojahn, Vieira, Schmidt, Pease, & Guizzardi, 2021).

The IEEE1872.2 Autonomous Robotics (AuR) Ontology Working Group has recently developed the AuR ontology (Goncalves et al., 2021). This standard is a logical extension to IEEE 1872–2015 Standard Ontologies for Robotics and Automation, CORA. The standard has been developed to be widely adopted from an ontological viewpoint. The standard does not commit to a specific top-level ontology. Indeed, the ontology definitions are compatible and aligned to two distinct top-level ontologies, namely DOLCE and SUMO. In this way, systems developed adopting this standard can interoperate with foundational views (like DOLCE) and practitioner-driven views (like SUMO).

Several works have already presented strategies based on using ontologies to improve motion planning in dynamic environments (Akbari, Rosell, et al., 2015; Beetz, Mösenlechner, & Tenorth, 2010; Diab, Akbari, Rosell, et al., 2017; Feyzabadi & Carpin, 2014). Zhao, Fillatreau, Elmhadhbi, Karray, and Archimede (2022) present an ontology "ENVOn", which contains geometric information of the objects present in the environment. Path planning is generated based on the information present in the ontology, and reasoning at the "primitive" tasks level is improved. Leidner, Borst, and Hirzinger (2012) presented work in which, through centralized world representation, they propose a method for solving arbitrary manipulation tasks depending on the type of objects. Diab, Akbari, Ud Din, and Rosell (2019) propose an ontological framework to organize the knowledge needed for physics-based manipulation planning, allowing to derive manipulation regions and behaviours. Akbari, Rosell, et al. (2018) proposes ontologies to separately describe the knowledge on the manipulation objects and on the manipulation actions. However, all of these works present isolated approaches that do not use an upper ontology as a base, making them difficult to be reusable. Moreover, such approaches have not presented inference systems to increase the robot semantic knowledge of the environment and the new objects that can be arrive on the environment, to improve motion planning.

Ontologies have shown great potential in improving the motion planning of agents in symbiotic work systems (e.g. a team of robots can work together to assemble a product on the shop floor, each robot being responsible for a specific task, without them clashing) (Akbari,

[1] https://www.w3.org/Submission/SWRL/
[2] https://www.ros.org
[3] https://kcl-planning.github.io/ROSPlan/
[4] https://www.w3.org/OWL/

Muhayyuddin, & Rosell, 2019; Schou, Andersen, Chrysostomou, Bøgh, & Madsen, 2018). Ontologies have also been applied in human-centric cyber–physical production systems (Kanazawa, Kinugawa, & Kosuge, 2019; Olivares-Alarcos, Foix, Borgo, & Alenyà, 2022; Umbrico, Orlandini, & Cesta, 2020). Recently with the advances in digitalization, strategies such as Digital Twins (DT) have been emerging, changing the limits and possibilities of the current autonomous systems (Phanden, Sharma, & Dubey, 2021). Different works have linked DTs together with semantic knowledge. Knowledge-based DT can help improve motion planning by using data and analytics to optimize robot movement (Havard, Jeanne, Savatier, & Baudry, 2017; Tuli, Kohl, Chala, Manns, & Ansari, 2021). Additionally, using machine learning algorithms, a DT can optimize the robot's movement over time, improving efficiency and reducing the risk of errors or accidents (Tuli et al., 2021).

2.2. Semantic maps

Robotic mapping. In the considerable body of literature about robot maps and mapping, maps are metrical in most cases, and less frequently, topological (Nüchter & Hertzberg, 2008). Going further, semantic maps augment traditional representations of robot workspaces, typically based on their geometry and/or topology (He, Sun, Hou, Ha, & Schwertfeger, 2021; Niloy et al., 2021; Sim & Little, 2009), with meta-information about their compositional elements properties, relationships, and functionalities. These provide robots with the ability to understand beyond the spatial aspects of the environment, the meaning of each element, and how humans interact with them (e.g. features, events, relationships, etc.) (Hanheide et al., 2017; Kostavelis & Gasteratos, 2015; Toscano, Arrais, & Veiga, 2017). Semantic maps deal with meta information that models the properties and relationships of relevant concepts in the domain in question, encoded in a Knowledge Base (KB). Semantic maps enable the execution of high-level robotic tasks efficiently, and several strategies have been presented (Achour, Al-Assaad, Dupuis, & El Zaher, 2022; Bernardo et al., 2021; Fernandez-Chaves, Ruiz-Sarmiento, Petkov, & Gonzalez-Jimenez, 2021), that include map building and planning for navigation.

2.3. Semantic Web Rule Language (SWRL)

To enhance the OWL-DL's expressivity and allow rules modelling, the SWRL has been developed (Horrocks et al., 2004). SWRL is a language to express rules and logic that, combined with OWL, enhances the reasoning of knowledge. The SWRL is a standard rule language based on OWL-DL and the Rule Markup Language (Rule ML) (Horrocks et al., 2004). According to Fiorentini, Rachuri, Suh, Lee, and Sriram (2010), the OWL/SWRL is the only approach that gathers ontology and rules in product development. A SWRL-rule consists of an antecedent (body) and a consequent (head). Both antecedent and consequent have multiple atoms. Atoms in these rules can be of the form C(x), P(x,y), where C is an OWL description, P is an OWL property, and x, y are either variables, OWL individuals or OWL data values (Schmidt-Schauß, 1988). Listing Eq. (1), describes the abstract syntax of a SWRL-rule (Fiorentini et al., 2010).

$$\begin{aligned} rule &::= 'Implies\,('[\text{ URIreference }]'\{\text{ annotation }\}\text{ antecedent consequent })' \\ antecedent &::=' Antecedent\,('\{\text{ atom }\}')' \\ consequent &::=' Consequent\,('\{\text{ atom }\}')' \end{aligned} \tag{1}$$

The SWRL-rules in OWL/RDF format can perfectly merge with the OWL-based proposed ontology. Besides, SWRL-rules offer a human-readable syntax, such as the above SWRL-rule means that if an agent's (e.g. belongs to the class "*robotic_agent*") has a battery charge (dataproperty – > "*BatteryLevel*") greater than 30, then the agent is available to execute a mission. A boolean variable (dataproperty – > "*BatteriesCharged*") is set to true.

$$robotic\_agent(?r)\ \wedge\ BatteryLevel(?r,?bt)\ \wedge swrlb:greaterThan(?bt,30) \rightarrow BatteriesCharged(?r,true) \tag{2}$$

Formalism: In Eq. (2), a "'robotic_agent(?r)" atom stores in the variable ?r the value of an instance of the "*robotic_agent*" class, a "*BatteryLevel(?r,?bt)*" atom stores in the variable ?bt the value of the *dataproperty* "*BatteryLevel*" if it is related to ?r. Through the set of constructs for SWRL,[5] for comparisons, the "*swrlb:greaterThan*" was used; "*swrlb:greaterThan(?bt, 30)*" is satisfied if the first argument ?bt is greater than the second argument (value: 30). In case this statement is true the *dataproperty* "*BatteriesCharged*", if it is related to ?r, takes on the value of the second element, i.e. *"true"*.

Different authors have resorted to using SWRL rules to infer knowledge. Wang et al. (2023) proposed safety management approaches for manufacturing processes based on a digital twin that improves upon traditional safety management based on the subjective human experience. Zhai, Martínez Ortega, Lucas Martínez, and Castillejo (2018) proposed an approach with rule-based reasoning to infer and provide OWL and SWRL-based query services for underwater robots. SARbot (Sun, Zhang, & Chen, 2019) uses OWL for ontology development and adopts SWRL rules to infer tasks to be performed according to the environment and obtain the state of the victims. The experiments were conducted using TurtleBot3 with a real robotic platform established in ROS. Zheng et al. (2022) presented a proposed automatic generation of manufacturing programs based on semantic descriptions and reasoning through rules; this approach showed significant advantages in stability and output quality. In summary, several works show the potentiality of using rules to improve manufacturing processes (O'connor et al., 2005; Umbrico, Cesta, & Orlandini, 2022; Zheng & Terpenny, 2013).

2.4. Research gaps

Recently and driven by industry 5.0, the exploration of new forms of cooperation and collaboration between humans and robots, aims efficient production (Simões, Pinto, Santos, Pinheiro, & Romero, 2022). The description of a Cyber–Physical Systems, like a Human–Robot Collaborative scenario, requires a model of complex adaptive behaviours of involved agents from both a "local perspective" (i.e., the point of view of an agent) and a "global perspective" (i.e., the point of view of the production and related constraints and objectives) (Borgo, 2019).

Current classical methods (e.g., PRM, RRT, etc.) used in trajectory planning become limited if the actions required to perform the task are subject to strong geometric constraints of the environment (lack of space to place objects, occlusions) and of the robot (accessibility of objects, kinematic constraints of the manipulators) (Li & Tian, 2020). It is crucial to use semantics to provide a complete description of the environment, the set of available actions and the state variables to represent the domain, as the robots need to perform actions efficiently and effectively.

This paper presents a domain ontology based on the CORA and AuR ontologies. The proposed ontology intends to serve as a basis for future works which focus on improving motion planning because, to date, existing approaches have not presented inference systems to increase the robot semantic knowledge of the environment and the new objects that can arrive at the environment, in order to improve motion planning. Moreover, all the works presented are isolated works, i.e., are not based on an upper ontology (Upper ontologies aim to describe reality from a general perspective to define general concepts that are the same in all domains. Top-level ontologies represent an excellent design choice for building new domain ontologies.). As noted in Jansen

[5] https://www.w3.org/Submission/SWRL/#Schmidt-Schauss89

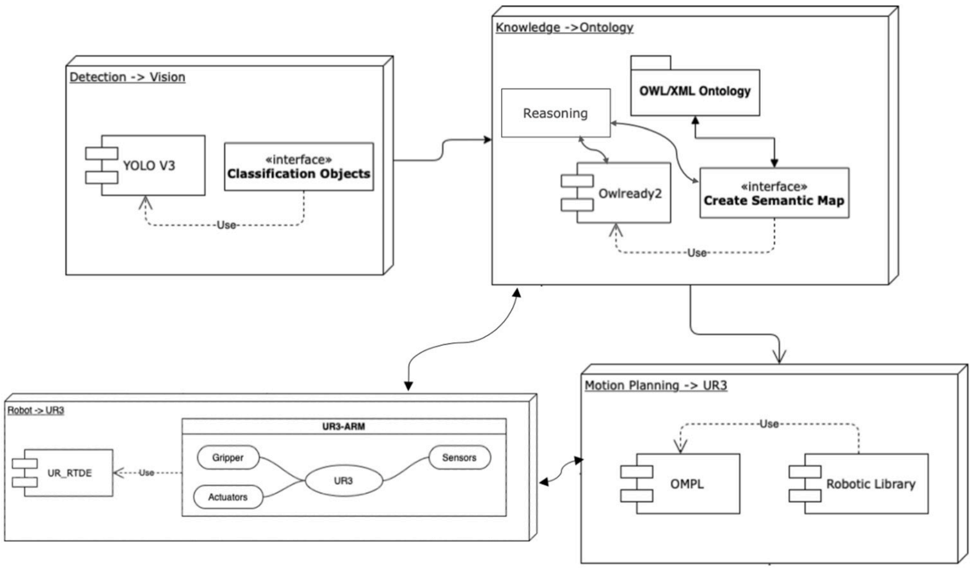


Fig. 1. Proposed framework overview.

and Schulz (2011), these concepts represent a stable theoretical base, promoting a clear structuring and disambiguation of new concepts and related relations. In summary, the strategies presented in the past are difficult to reuse by other researchers in the field.

SWRL-rules are increasingly becoming an important form of knowledge representation on the Semantic Web (Mun & Ramani, 2011). Unlike in the areas of production and manufacturing, in the area of robotics, there is still a gap on using SWRL rules to improve the reasoning on robotic agents and the representation of the environment, when they perform tasks. As such, we present the examples for rule development and application examples in this paper, along with their potential, i.e., how they can be easily applied in robotic systems to infer new knowledge.

3. Proposed ontological framework

In recent years, given the increasing interest in robots being able to coexist with humans and perform high-level automated tasks, multiple contributions have been presented to complement geometrical information of the environments with semantic knowledge. The proposed framework, depicted in Fig. 1, comprises four main modules: object detection, motion planner, hardware-level of robotic agent, and the knowledge-based reasoning engine. The implementation of each module is detailed in the following sections.

3.1. Object detection

A deep neural network was trained for the detection and classification of the objects present in the environment where the robotic agent is located was used. Based on this information, a semantic map was created. Object detection algorithm YOLO v3 (Redmon & Farhadi, 2018) by Darknet for ROS was used for object detection (Bjelonic, 2016–2018). YOLO has a few advantages over classifier-based systems. It looks at the whole image at test time so that the global context in the image informs its predictions. Moreover, it makes predictions with a single network assessment, in contrast to systems like R-CNN, which require thousands for a single prediction (Buric, Pobar, & Ivasic-Kos, 2018). This makes it several orders of magnitude faster than the previous methods.

3.2. Motion planning

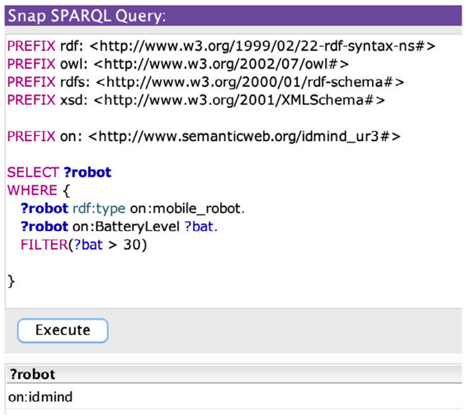
The trajectory planner comprises the Robotics Library (RL) (Rickert & Gaschler, 2017), a self-contained C++ library for robot kinematics, motion planning and control. It covers mathematics, kinematics and dynamics, hardware abstraction, motion planning, collision detection, and visualization, which uses the Open Motion Planning Library (OMPL) (Sucan, Moll, & Kavraki, 2012) as the core set of planning algorithms. OMPL allows planning under geometric constraints as well as differential constraints, including those that require dynamic simulations (OMPL uses the Open Dynamic Engine for dynamic simulation). Based on the classical libraries mentioned above, we implemented three sampling-based algorithms: (i) Probabilistic Roadmap Method (PRM) (Siméon et al., 2004); (ii) Rapidly-expanding Random Trees (RRT) (LaValle et al., 1998); and (iii) exploring/exploiting tree (EET) (Rickert, Sieverling, & Brock, 2014). However in the present work the results are presented based on the PRM algorithm, which proven better results for the robotic system used.

Semantic knowledge was used to improve the manipulator agent's motion planning. The ontology is consulted to identify all objects and their properties (e.g. dimensions, weight, position on the map, etc.) in the robot's work area. Based on the semantic information, the scenario around the manipulator agent is created to create a collision-free path and replan a new path if a change in the environment makes the initial path unfeasible. The environment surrounding the agent is created based on primitive geometric shapes (e.g. Box, Cone, Cylinder and Sphere), as they provide better performance than convex hulls or concave geometries (Kockara, Halic, Iqbal, Bayrak, & Rowe, 2007).

3.3. Hardware-level of the robotic agent

In order to control the robotic arm (UR3 from Universal Robots), and the gripper (2f-140 from Robotiq), two toolboxes were developed, one for the robotic arm (toolbox_ur3) and another for the gripper (toolbox_gripper), these were developed to work on the ROS, having been written in C++, these toolboxes have the advantage of being easily reused by other research groups. They are written based on the ur-rtde[6] library, which communicates with UR3 via the real-time data exchange protocol (RTDE). By using these libraries, the desired trajectory points

[6] https://sdurobotics.gitlab.io/ur_rtde


Snap SPARQL Query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>

PREFIX on: <http://www.semanticweb.org/idmind_ur3#>

SELECT ?robot
WHERE {
  ?robot rdf:type on:mobile_robot.
  ?robot on:BatteryLevel ?bat.
  FILTER(?bat > 30)

}
```

?robot
on:idmind

Fig. 2. Example of SPARQL query.

obtained from the methods described in the previous section, are sent to the built-in controller in the UR3 industrial robot. This controller allows a smooth control of the robot joints, with repeatability equal to 0.1 [mm]. As such, the final trajectories of the robot does not have significant disturbances.

3.4. Ontology description and knowledge-based reasoning engine

The ontology can be written using the Protégé software. Protégé version 5.5.0 was used (Protégé, 2022). The ontology was verified through version 2.2.0 of Pellet logic Reasoner to ensure that it is free of inconsistencies (Sirin, Parsia, Grau, Kalyanpur, & Katz, 2007). Protégé is a free, open-source editor for developing the ontologies produced by Stanford University. It is a java-based application (multi-platform), with plugins such as onto Viz to visualize the ontologies. The backbone of protégé is that it supports the tool builders, domain specialists, and knowledge engineers. The Owlready2 library was used to access the developed OWL ontology. Owlready2 is a Python module; it can load OWL 2.0 ontologies as Python objects, modify them, save them, and perform reasoning. Owlready2 allows transparent access to OWL ontologies (unlike the standard Java-based API). Lamy (2017).

Owlready2 was used to create and edit SWRL-rules. The class Imp ("Implies") represents a rule. The easiest way to create a rule is to define it using Protégé-like syntax, with the .set_as_rule() method.

$$\begin{aligned} rule &= Imp() \\ &\quad rule.set\_as\_rule(\text{""""}robotic\_agent(?r)\ BatteryLevel(?r, ?bt) \\ &\quad swrlb : greaterThan(?bt, 30) \rightarrow BatteriesCharged(?r, true)\text{"""}) \end{aligned} \tag{3}$$

SPARQL (SPARQL Protocol and RDF Query Language) is the standard query language and protocol for Linked Open Data and RDF databases Sirin and Parsia (2007). For reasoning with the ontology, different SPARQL queries were used (see Fig. 2).

Fig. 3 presents the hierarchical class where the main concepts are defined in the proposed domain ontology. The main classes of the ontology proposed here are based on the classes of the upper ontology DOLCE since the latter is used as the basis for the construction of our domain ontology presented here. That said the main classes of the domain ontology presented here are: *Situation*, *InformationEntity*, *Object*, *Event* and *Abstract*. The definition of these classes and their sub-classes is presented in the next, where each of the constituent parts of the domain ontology proposed is explained in detail.

In Fig. 4 the *AllObjects* class represents, any physical, social, or mental object, or a substance. Following DOLCE, objects are always participating in some event (at least their own life), and are spatially located. The class *Social_object* represent any *Object* that exists only within some communication Event, in which at least one *Object* participates in. The definitions of classes *AllObjects* and *Social_object* are expressed based on the upper ontology DOLCE. The class *Types_Devices* is a collection of properties that define different components and behaviours of a type of device (i.e., actuators, sensors, etc.).

In Fig. 5 the *Agent* class is shown, which based on DOLCE,[7] is defined as: "*Any agentive Object, either physical (e.g., a whale, a robot, or an oak tree) or social (e.g., a corporation, an institution, or a community)*". This class is further subdivided into *human_agent* and *robotic_agent*. To cover the agents that can interact in the environment (e.g. robotic agent, human, etc.).

The *robotic_agent* class is subdivided into:

- *mobile_robot:* robot that is able to move in the surrounding (locomotion) (i.e., autonomous mobile robot (AMR)).
- *robotic_arm:* is a type of mechanical arm, usually programmable, with similar functions to a human arm.

Different classes were defined in the domain ontology to improve the robotic arm agent planning task based on semantic knowledge (Figs. 6 and 8). The knowledge-based reasoning engine reads the initial state of the world and extracts abstract knowledge related to the agent's environment, such as the type of objects in the manipulation zone, the manipulation constraints (e.g. *FixedObjects*, and *ManipulatableObject*) (Fig. 6), the state of the goal, the state of the agents, etc. (Fig. 7) Based on a reasoning process, the instantiated knowledge is inferred from the abstract knowledge and fills the data structures in the *Actions* and *Motion* layer (such as the current manipulation constraints, the trajectory state (e.g. *CollidingTrajectory*, *FreePath*, etc.), which are periodically updated.

The *Regions* class defines different types of regions is subdivided into tree classes:

- *GoalRegion* is a region defined around the goal state.
- The *ObjectRegions* class is subdivided into the *ManipulationRegion* This is the region that defines the manipulator's workspace (e.g. objects contained in this region are possible manipulable objects). The *NonManipulationRegion* is a inverse of a *ManipulationRegion*.
- *Rooms* is a space that can be occupied or where something can be done (e.g. Living Room, Bedroom, etc.).

When moving in the environment, the robotic agent collects information from the environment that it stores in the ontology to update it. One of these parameters is the euclidean distance between the geometric centre of each object and the agent's referential. Based on the knowledge of how far away the object is and based on the reach of the robot's arm, the following SWRL rule can be written, using the ontology concepts, to automatically identify which objects are in the workspace (*ManipulationRegion*).

$$\begin{aligned} &AllObjects(?obj)\ \wedge\ robotic\_arm(?r)\ \wedge\ Reach(?r, ?re) \\ &\quad \wedge\ EuclideanDistance(?obj, ?dist)\ \wedge swrlb : lessThan(?dist, ?re) \\ &\quad \rightarrow located\_at(?Obj, ManipulationRegion) \end{aligned} \tag{4}$$

In Fig. 7 are depicted the relations of the *Situation* class. Based on upper ontology DOLCE. *A view, consistent with ("satisfying") a Description, on a set of entities. It can also be seen as a "relational context" created by an observer on the basis of a "frame" (i.e. a description). For example, a Plan_execution is a context including some actions executed by agents according to certain parameters and expected tasks to be achieved from a Plan*;

The *Configuration* class defines the states of the agents at each instant they operate (e.g. *ArmCollisingState*, *BaseAtPoseState*, etc.).

[7] http://www.ontologydesignpatterns.org/ont/dul/DUL.owl

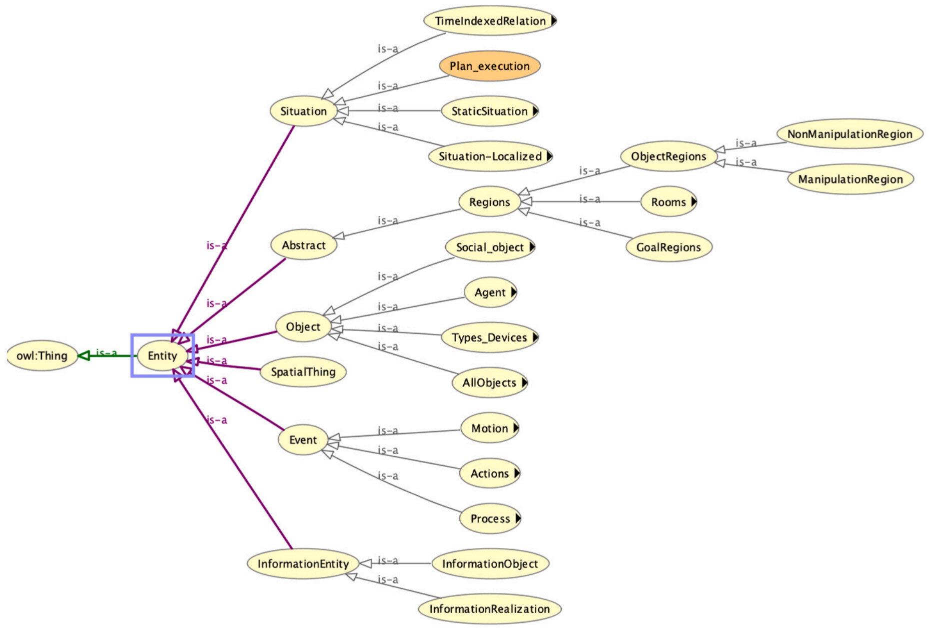


Fig. 3. Snapshot of the main ontology (relations is-a).

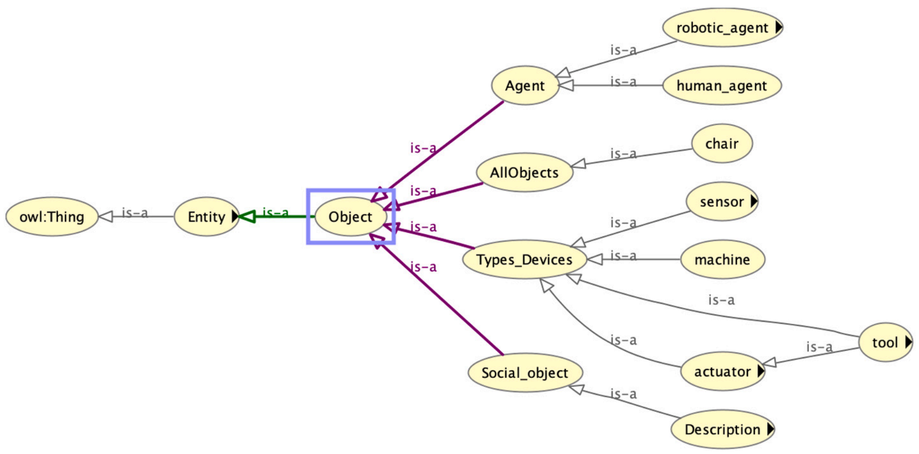


Fig. 4. Representation in the ontology of subclasses of class *Object* (relations is-a).

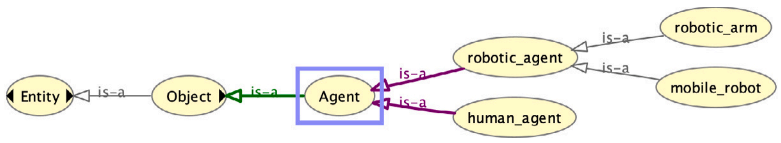


Fig. 5. Relationships of the class *Agent*.

Tree classes, derived from a general *Actions* class (i.e. contains the possible actions of the agents) have been defined (Fig. 8):

- *Mobile_actions* actions of mobile agent.
- *Gripper_actions* actions of gripper.
- *Manipulator_actions* actions of manipulator agent.

4. Real world implementation

In the real world implementation, an autonomous mobile manipulator robot (AMMR) was used, consisting of a mobile base and the Universal Robot UR3 equipped with a Robotiq gripper. These robots combine autonomous navigation with autonomous manipulation capabilities. With a wheeled mobile base and sensors that allow the robot to sense where it is in the environment, AMMRs can navigate from point A to B with the ability to “see”. These robots are also equipped with an arm and end-effector to pick up different items physically. The goal of autonomous mobile manipulation as defined by the IEEE Robotics & Automation Society[8] is to execute complex manipulation tasks in an unstructured and dynamic environment where cooperation with humans may be required. For the experimental validation of the proposed framework, only the UR3 manipulator was used.

[8] https://www.ieee-ras.org/mobile-manipulation

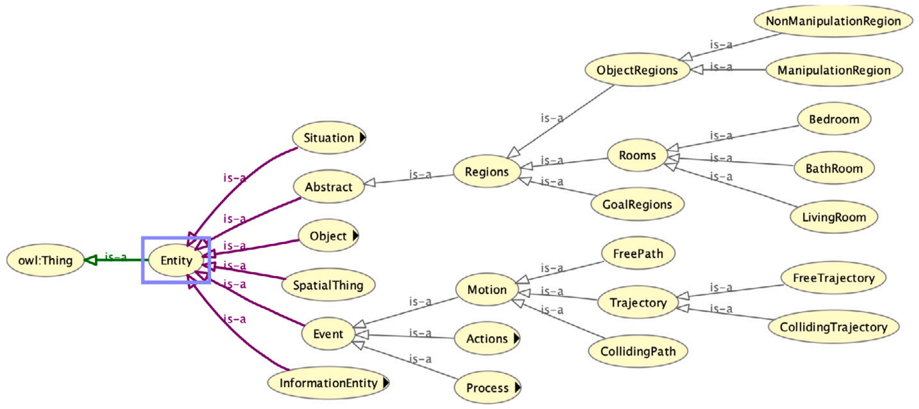


Fig. 6. Representation in the domain ontology of the environment (*Regions* class), and motion planning (*Motion* class).

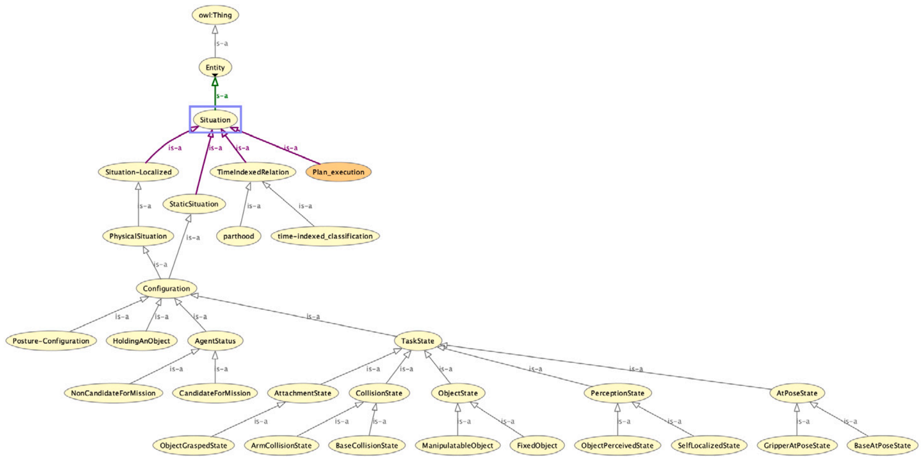


Fig. 7. Representation in the domain ontology of the actions status.

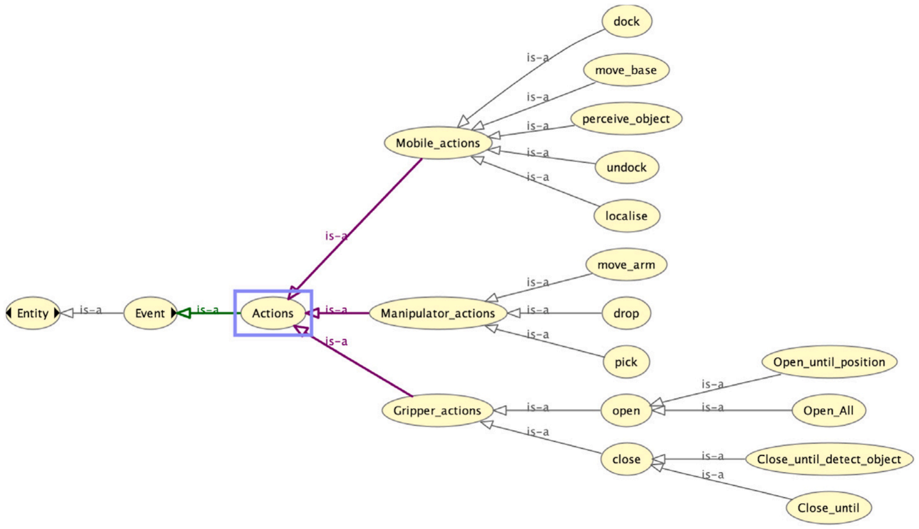
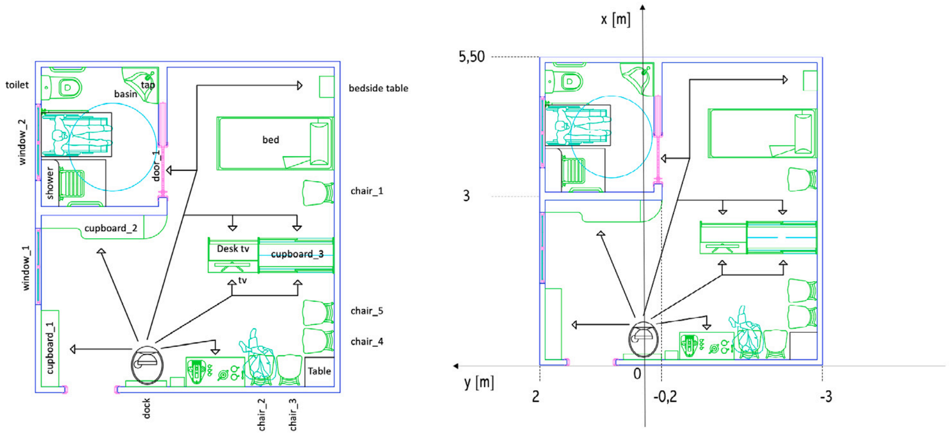


Fig. 8. Representation in the domain ontology of the *Actions* that the agent can perform (relations is-a).



(a) Position of objects in environment. (b) Metric representation of the rooms.

Fig. 9. Smart-home environment, built on the robotics laboratory of Instituto Politécnico de Castelo Branco.

The experimental validation is realized in a simple environment of a house, based on a smart-home environment built in the robotics laboratory of Instituto Politécnico de Castelo Branco (Fig. 9).

The framework has been implemented and tested within Robot Operating System (ROS) Noetic and Ubuntu 20.04.4 LTS operating system with an Intel®Core™ i7-7740X CPU @ 3.30 GHz × 8 processor, 16 GB RAM, and Quadro P2000/PCIe/SSE2 Graphics. The input video is obtained using Intel® RealSense™ D415i Depth Camera (It is represented in Fig. 10b with frame "camera_link").

4.1. Hardware and software low-level

The AMMR used in the study is composed of (Fig. 10): An omnidirectional mobile base with mecanum wheels, giving the robot the ability to move in any direction; A UR3.[9] robotic arm from Universal Robotics for object manipulation, equipped with a Robotiq 2f-140 gripper[10] A set of sensors (e.g. rear and front lasers, and cameras for autonomous navigation and obstacle recognition) are also present on the mobile base.

Fig. 10b shows the transformation (tf[11]) of the different constituent systems of the robot. The tf maintains the relationship between the coordinate structures in a time-buffered tree structure and allows the user to transform points, vectors, etc., between any two coordinate structures at any point in the desired time. Based on this, it is possible to quickly calculate the robot's coordinate on the map, or for example, the coordinate of a particular observed object concerning any of the robot's coordinate systems.

4.2. Information of robotic agent(s)

As previously mentioned, an AMMR is used. Based on our ontology, our agent (AMMR) is subdivided into two agents that work together a mobile agent (instantiated as "idmind") and a manipulator agent (instantiated as "ur3_arm") (Fig. 11). We use the *part_of* property to link both agents together as a whole.

A node was then created in ROS, which subscribes to all the information from the sensors present in the agents and updates the properties of our agents in the ontology. In Fig. 12 are depicted the different properties of the agents that are constantly updated. These properties are later used to improve reasoning. In order to generate a set of tasks that our agent has to perform (e.g. pick objects, transport an object from one point to another, etc.)

4.3. Representation of the environment - semantic map

Previously the environment was represented using a semantic map (Bernardo et al., 2021). In this paper, the previous work was enhanced because it was possible to identify the room where the agent was based on the objects around it. For example, based on picture 13, if the agent sees a chair, a bedside table and a bed, he automatically knows its location ("*BedRoom*").

Now besides being able to identify the room based on the objects observed (Figs. 9a and 13), rules were created to identify the room where the agent is present. For this, it is only necessary the knowledge of the floor plan of the environment (Fig. 9b), and the knowledge of the position of the agent in the environment (e.g. localizing the robot within it using an Adaptive Monte-Carlo Localizer (AMCL[12])). Examples of rules are as follows:

- Rule to check if the agent is in the "*LivingRoom_1*":

$$\begin{aligned}&Agent(?Ag) \wedge position\_x(?Ag,?px) \wedge position\_y(?Ag,?py) \wedge\\ &swrlb: greaterThanOrEqual(?px,0) \wedge swrlb: lessThanOrEqual(?px,3) \wedge\\ &swrlb: greaterThanOrEqual(?py,-3) \wedge swrlb: lessThanOrEqual(?py,2) \wedge\\ &\rightarrow located\_at(?Ag, LivingRoom\_1)\end{aligned} \tag{5}$$

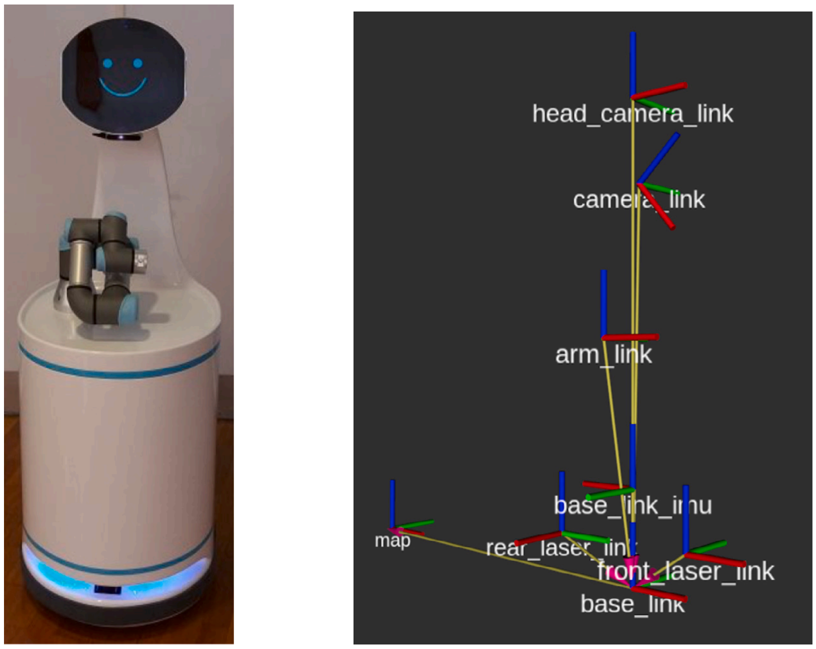
- Rule to check if the agent is in the "*BathRoom_1*":

$$\begin{aligned}&Agent(?Ag) \wedge position\_x(?Ag,?px) \wedge position\_y(?Ag,?py) \wedge\\ &sswrlb: greaterThan(?px,3) \wedge swrlb: lessThanOrEqual(?px,5.5) \wedge\\ &swrlb: greaterThan(?py,-0.2) \wedge swrlb: lessThanOrEqual(?py,2)\\ &\rightarrow located\_at(?Ag, BathRoom\_1)\end{aligned} \tag{6}$$

- Rule to check if the agent is in the "*BedRoom_1*":

$$\begin{aligned}&Agent(?Ag) \wedge position\_x(?Ag,?px) \wedge position\_y(?Ag,?py) \wedge\\ &sswrlb: greaterThan(?px,3) \wedge swrlb: lessThanOrEqual(?px,5.5) \wedge\\ &swrlb: greaterThanOrEqual(?py,-3) \wedge swrlb: lessThanOrEqual(?py,-0.2)\\ &\rightarrow located\_at(?Ag, BedRoom\_1)\end{aligned} \tag{7}$$

[9] https://www.universal-robots.com/pt/produtos/ur3-robot/
[10] https://robotiq.com/products/2f85-140-adaptive-robot-gripper
[11] https://wiki.ros.org/tf
[12] https://wiki.ros.org/amcl



(a) Robotic agent (AMMR). (b) Frames of robot.

Fig. 10. Autonomous Mobile Manipulator Robot used in experimental validation.

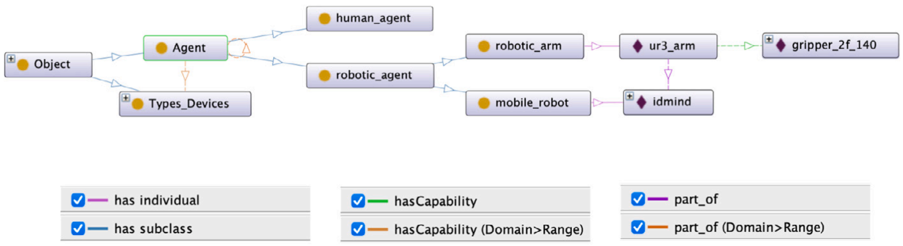
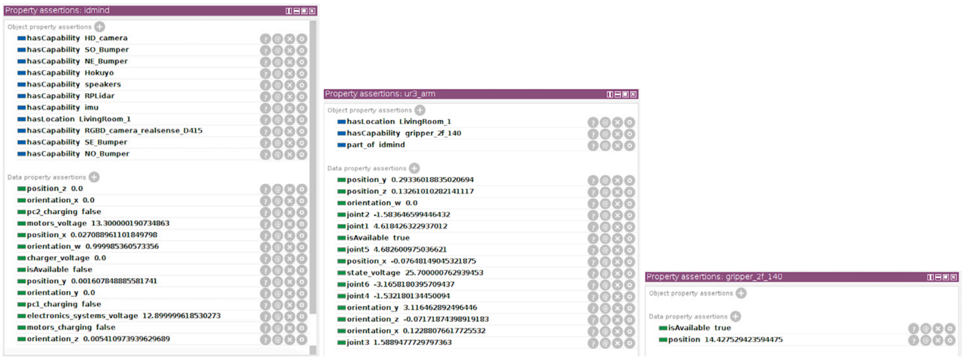


Fig. 11. Agent instances (AMMR) in the ontology.



Property assertions: idmind

Object property assertions

hasCapability HD_camera
hasCapability SO_Bumper
hasCapability NE_Bumper
hasCapability Hokuyo
hasCapability speakers
hasCapability RPLidar
hasCapability imu
hasLocation LivingRoom_1
hasCapability RGBD_camera_realsense_D415
hasCapability SE_Bumper
hasCapability NO_Bumper

Data property assertions

position_z 0.0
orientation_x 0.0
pc2_charging false
motors_voltage 13.300000190734863
position_x 0.027088961101849798
orientation_w 0.999985360573356
charger_voltage 0.0
isAvailable false
position_y 0.0016078488855581741
orientation_y 0.0
pc1_charging false
electronics_systems_voltage 12.899999618530273
motors_charging false
orientation_z 0.005410973939629689

(a) Properties of base (idmind).

Property assertions: ur3_arm

Object property assertions

hasLocation LivingRoom_1
hasCapability gripper_2f_140
part_of idmind

Data property assertions

position_y 0.29336018835020694
position_z 0.13261010282141117
orientation_w 0.0
joint2 -1.583646599446432
joint1 4.618426322937012
isAvailable true
joint5 4.682600975036621
position_x -0.07648149045321875
state_voltage 25.700000762939453
joint6 -3.1658180395709437
joint4 -1.532180134450094
orientation_y 3.116462892496446
orientation_z -0.07171874398919183
orientation_x 0.12288076617725532
joint3 1.5889477729797363

(b) Properties of robotic arm (ur3_arm).

Property assertions: gripper_2f_140

Object property assertions

Data property assertions

isAvailable true
position 14.427529423594475

(c) Properties of tool (gripper_2f_140).

Fig. 12. The three instances that make up the AMMR and their respective properties.

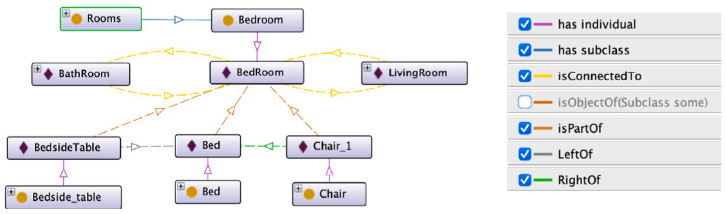
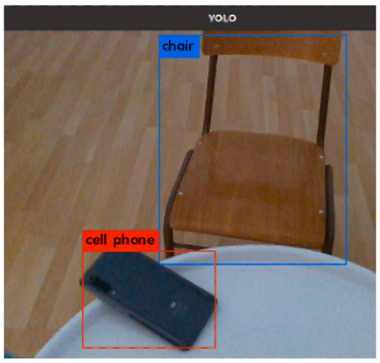
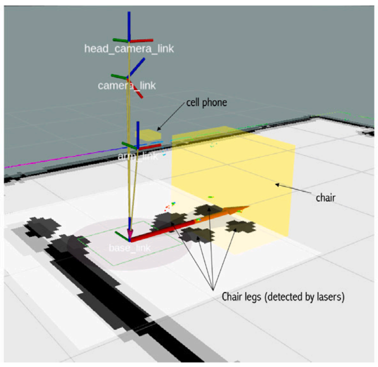


Fig. 13. Semantic map of a Bedroom (Bernardo et al., 2021).



(a) YOLO v3: Detection of the objects.



(b) Representation of the space occupied by the object.

Fig. 14. Detection of the objects based on YOLO v3, and representation of the space occupied by the object with a box.

This approach allows us to increase further the confidence level in predicting the agent's location in the environment. A new system is thus presented to solve the problems to obtain an accurate localization of agents in indoor environments (Yassin et al., 2016).

4.4. Object detection algorithm

For the obstacle detection test, two objects (a chair and a cell phone) were placed in front of the robot in a random position, as we can see in Fig. 14a. We see the detection of two objects of two different classes (class "*chair*" and class "*cell phone*"), in a practical example, based on YOLO v3. The darknet algorithm publishes a set of bounding boxes which gives information about the position and size of the bounding box in pixel coordinates of each object (Information regarding the camera frame "camera_link"). Using the darknet_ros_3d library,[13] the information from the darknet_ros topic is combined with the point cloud information from the RGBD (Red Green Blue and Depth) camera, delimiting a 3D bounding box of each observed object (Fig. 14b) already in metric units. The geometric centre of the 3D bounding box of the object is also calculated. Finally, the previously calculated coordinates are converted from the RGBD camera frame to the map frame (Fig. 10b).

All the information collected regarding the detected objects now has to be stored in the ontology (Fig. 15). For this, a node was created in ROS, which will update our knowledge base (ontology). In this node, it is verified if the class to which the observed object belongs already exists (The object classes defined in the ontology have the same name as the classes defined in the YOLO v3). If not, a new class is created that corresponds to the observed object, which will be a sub-class of the *AllObjects* class already defined in the ontology (Fig. 4). After creating the object class it is then instantiated. The name of the object instance is defined based on the class it belongs to + prefix. The prefix is "_number" where the number is incremental according to the number of already detected objects belonging to the object class, going from 1 to inf. The object is then instantiated along with all known object information.

Before instantiating a new object, the ontology is always consulted in order to check if the object observed had not already been observed at a previous moment. If it has already been observed, only the object's information is updated if appropriate.

According to Fig. 9b, knowing the metric dimensions of the rooms, together with the information collected from each object, we can thus write the following rules to identify the room in which each observed object is located:

- Objects contained in the area belong to the room instantiated as "*LivingRoom_1*":

$$\begin{aligned}&AllObjects(?obj) \wedge Pos\_x(?obj, ?px) \wedge Pos\_y(?obj, ?py) \ \wedge \\ &swrlb: greaterThanOrEqual(?px, 0) \wedge swrlb: lessThanOrEqual(?px, 3) \ \wedge \\ &swrlb: greaterThanOrEqual(?py, -3) \ \wedge swrlb: lessThanOrEqual(?py, 2) \ \wedge \\ &\rightarrow hasLocation(?obj, LivingRoom\_1)\end{aligned} \tag{8}$$

- Objects contained in the area belong to the room instantiated as "*BathRoom_1*":

$$\begin{aligned}&AllObjects(?obj) \wedge Pos\_x(?obj, ?px) \wedge Pos\_y(?obj, ?py) \wedge \\ &sswrlb: greaterThan(?px, 3) \wedge swrlb: lessThanOrEqual(?px, 5.5) \ \wedge \\ &swrlb: greaterThan(?py, -0.2) \wedge swrlb: lessThanOrEqual(?py, 2) \\ &\rightarrow hasLocation(?obj, BathRoom\_1)\end{aligned} \tag{9}$$

- Objects contained in the area belong to the room instantiated as "*BedRoom_1*":

$$\begin{aligned}&AllObjects(?obj) \wedge Pos\_x(?obj, ?px) \wedge Pos\_y(?obj, ?py) \wedge \\ &sswrlb: greaterThan(?px, 3) \wedge swrlb: lessThanOrEqual(?px, 5.5) \ \wedge \\ &swrlb: greaterThanOrEqual(?py, -3) \wedge swrlb: lessThanOrEqual(?py, -0.2) \\ &\rightarrow hasLocation(?obj, BedRoom\_1)\end{aligned} \tag{10}$$

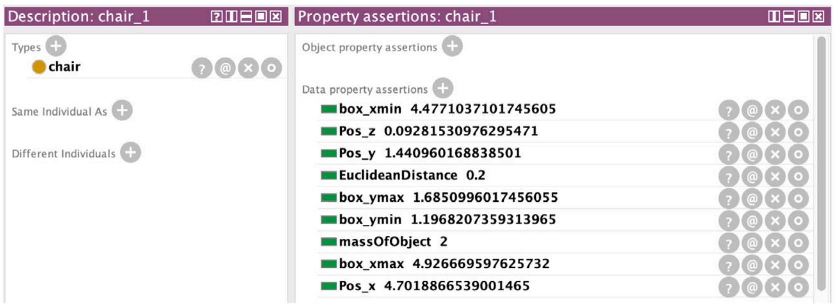
4.5. Implementation of the motion planning

The agent (AMMR) was modelled in 3D software, except for UR3 where we used the *.step* files that are available on the Universal Robotics website.[14] (Fig. 16). In order to define the constraints imposed by the shape of its own body of the agent. As previously mentioned, all information regarding each object observed in the environment has been stored in the ontology, as shown in Fig. 15 All the information about each object can be consulted at any time to improve the agents' reasoning, whenever necessary.

In Fig. 16 is depicted the scenario created based on the information in Fig. 14a that was previously stored in the ontology. In the scenario, only the objects that are considered fixed are represented because those that you want to manipulate cannot be represented, or it would generate an error in the creation of the plan, that is, only the *"chair_1"* appears defined, the "cell_phone_1" is the object that you want to move to another position.

[13] https://github.com/IntelligentRoboticsLabs/gb_visual_detection_3d

[14] https://www.universal-robots.com/download/mechanical-cb-series/ur3/robot-ur3-step-file-cb-series/



Description: chair_1

Types
chair

Same Individual As

Different Individuals

Property assertions: chair_1

Object property assertions

Data property assertions

```
box_xmin  4.4771037101745605
Pos_z  0.09281530976295471
Pos_y  1.440960168838501
EuclideanDistance  0.2
box_ymax  1.6850996017456055
box_ymin  1.1968207359313965
massOfObject  2
box_xmax  4.926669597625732
Pos_x  4.7018866539001465
```

Fig. 15. Stored information regarding a detected object (*chair_1*).

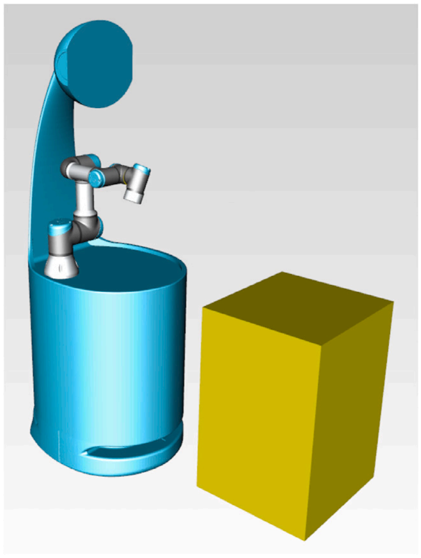


Fig. 16. 3D representation of the agent's geometric model (AMMR), and representation of the obstacles (in yellow) that lie within the agent's workspace.

The ROSPlan framework was used to perform the planning tasks. Different action interfaces were written in C++ to control the robotic agent. These actions are as listed in the *Manipulator_actions*, *Mobile_actions*, and *Gripper_actions* classes (Fig. 8).

Through the inferred knowledge based on semantic knowledge (i.e., the relationships between objects, the relationships between objects and the environment, the current location of the robot, existing constraints, etc.), ROSPlan, generated efficient task plans based on the actions defined in the ontology. Furthermore, ROSPlan and the developed interface actions proved to be very efficient in controlling the low-level interface with the agent's semantic reasoning.

5. Validation of the proposed framework in a real environment

In order to validate the proposed framework, a test was performed in the experimental apparatus (Fig. 17a), where the manipulator UR3 is commanded to pick a known object "*cell_phone_1*" previously observed as a goal (Fig. 17b). The information about it was already in the knowledge base (ontology), along with the position of the robot in the living room. An object "*bottle_1*" was positioned in front of the object to be picked, making it impossible to pick without collision with the object, and still making the "*cell_phone_1*" an unobservable object (Fig. 17c).

For the scenario of Fig. 17b a simple pick and place plan can be obtained. However, for the scenario in Fig. 17c the same plan will fail because the position of the "*bottle_1*", does not allow the pick movement. In the following, will be presented the several steps to implement the first plan and how the system will use the semantic information gathered along with the inference engine to improve the motion planning, and make the robot capable to pick and place the "*cell_phone_1*" when the object "*bottle_1*" is in front of it.

As a first step it was written the problem in PDDL, to pick and place the "*cell_phone_1*", which was later interpreted by the ROSPlan framework (The Metric -FF,[15] planner, is a domain independent planning system developed by Joerg Hoffmann. The system is an extension of the FF planner to (ADL combined with) numerical state variables). The initial generated plan is visible in Fig. 18. After the plan was generated, the interface actions, interconnect the plan with the lower level control actions, allowing the robotic agent (AMMR) to complete the plan (Fig. 19 "rosplan_perceive_object", "rosplan_interface_pick", "rosplan_interface_drop").

During execution, if an action fails due to changes in the environment, the planning agent reformulates the PDDL problem by re-planning. To do this, the dispatch of the initial order is cancelled, and the new plan is dispatched based on the updated information (i.e. a new problem in pddl is written automatically).

Although the position of the object to pick is known, before picking any object, its position is always recalculated in order to correct possible errors in the position of the base in the environment, as well as to detect new objects that may or may not exist in the environment, but that previously had or had not been observed (this is performed by the action: "*perceive_object*"). After, the pick action is used, which, through the known position of the object to be picked, calls the action server of the manipulator (Fig. 20, */ur3_server*), which, in turn, calls the library created to generate the trajectory to perform the pick. After, the drop action, executes the drop of the object in a desired position, in this case we use the AMMR base, which is divided into three zones: "*left_platform*", "*middle_platform*", and "*right_platform*".

Fig. 20 shows a simplified graph, depicting the implementation of the framework. It was obtained based on the rqt_graph,[16] available on ROS, in which the primary nodes and topics of the motion system of the robotic arm are represented. The node */verifier_path* is constantly subscribing to the */path_arm* generated for the robot's movement and the file with the geometrical representation of the space surrounding the manipulator, as shown in Fig. 16. The feedback concerning the path is constantly being published, updating its state in the ontology (Fig. 6). Whenever the path is not verified, the order sent to the manipulator is cancelled, and a new plan for the robot motion is generated based on the recent information perceived.

As described above, Fig. 17c, the "bottle_1" makes picking impossible without a collision, so the initial plan (Fig. 18) is cancelled. Fig. 23 represents the coordinate of the position in the workspace of each object to perform the picking; this coordinate represents the centre of mass of each object ("*cell_phone_1*" and "*bottle_1*"). The coordinate (i.e. [x,y,z] m) of the picking position of the "*cell_phone_1*" is [0.49,

[15] https://fai.cs.uni-saarland.de/hoffmann/metric-ff.html
[16] http://wiki.ros.org/rqt_graph

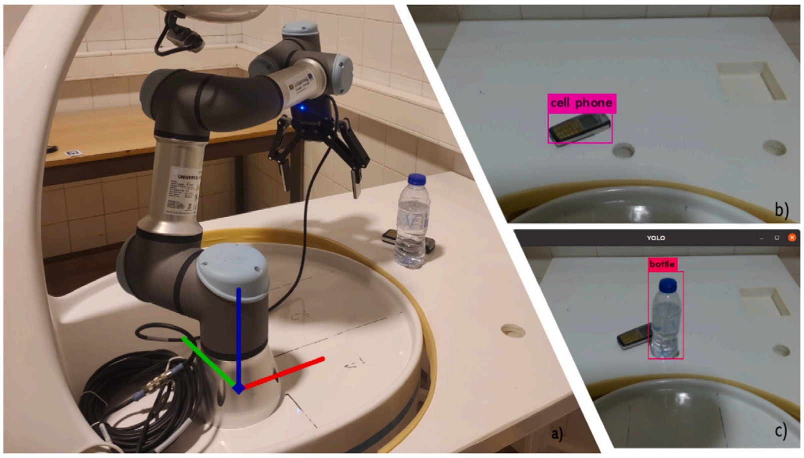


Fig. 17. Practical evaluation setup. (a) Apparatus overview. (b) First observation, of the environment. c) Final observation of the environment, where the cell phone is no longer observed, as a bottle was placed covering it.

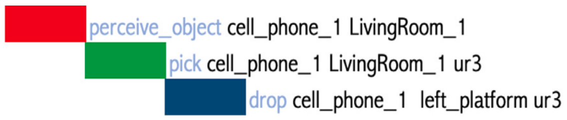

perceive_object cell_phone_1 LivingRoom_1
pick cell_phone_1 LivingRoom_1 ur3
drop cell_phone_1 left_platform ur3

Fig. 18. Generated initial plan.

0.16, 0.09] m, being the picking coordinate of the *"bottle_1"* [0.43, 0.11, 0.11] m. (Note: The Z-coordinate value cannot be used as a reference because it does not correspond to the real value of the centre of mass. It is automatically adjusted, based on the gripper characteristics and the height of the object along the Z-axis, so that the object being gripped is in the centre of the tool when it is being picked). We can thus verify that the distance between each object based on the position of its centre of mass is 6 cm on the X-axis, 5 cm on the Y-axis, i.e., based on the characteristics of the UR3 manipulator, and the dimensions of the gripper with which it is equipped, it is impractical to pick the *"cell_phone_1"* without colliding with the *"bottle_1"*.

The ontology is now consulted to know if the object that makes picking impossible can be manipulated based on the manipulator's and the gripper's characteristics (Fig. 21). As observed in the figure below, the object is manipulable. Based on this information, a new problem is generated to execute a plan that enables picking up the "*cell_phone_1*" without collision (Fig. 22).

Here the strategy/behaviour used is to move the obstacle that causes the collision, in case it is manipulable, to a known position that is free on the manipulator's platform (i.e. "*right_platform*"). If the object that was responsible for the collision is not manipulable, our system will send an error message informing us that it is not allowed to reach the goal.

After re-planning the motion, the initial task to be performed was successfully accomplished. The newly generated plan can be divided into three sets of tasks (Fig. 22); each set is composed of: a *"perceive_object"*, *"pick"* and *"drop"* actions. The first task consists of moving the obstacle ("*bottle_1*") to a known position in the mobile agent's base, which is present in the knowledge base (e.g. "*right_platform*"). The second task consists in picking the "*cell_phone_1*" and dropping it in another known position in the mobile agent's base (e.g. "*left_platform*"). Finally, the last task consists of returning the "*bottle_1*" to the initial position where it was initially observed.

In Fig. 23 the paths taken by the robotic manipulator are depicted. The trajectory refers to the centre point of the tool. The graph at the top of Fig. 23 represents the trajectories performed by the manipulator to execute the pick and drop actions of the "*bottle_1*"; the legend of the figure follows the chronological order in which the movements are performed; first the manipulator moves from the manipulator's home position to an approach point to perform the picking (red trajectory), then the manipulator moves linearly until the position to complete the picking (pink trajectory), after grabbing the "*bottle_1*", it returns to perform the previous trajectory (pink trajectory), in the opposite direction. After this, it begins to move towards the approach point to complete the drop of the "*bottle_1*" (dark green trajectory), then it approaches the drop point of the "*bottle_1*" linearly (blue trajectory); after performing the drop, it returns to complete the previous trajectory (blue trajectory) in the opposite direction. Finally, the manipulator moves to its home position (light green trajectory), which allows it to deviate from the angle of view of the RGBD camera.

The graph at the bottom of Fig. 23 represents the trajectories performed by the manipulator to execute the pick and drop actions of the "*cell_phone_1*". Like the graph's legend above, this one also follows the chronological order of the manipulator's steps. First, the manipulator moves from the manipulator's home position to an approach point to perform the picking (red trajectory), then the manipulator moves linearly to the position to complete the picking (pink trajectory); after picking up the "*cell_phone_1*", it returns to the previous trajectory (pink trajectory), in the opposite direction. After this, it begins to move towards the approach point to perform the drop of the "*cell_phone_1*" (dark green trajectory), then it approaches the drop point of the "*cell_phone_1*" linearly (blue trajectory); after performing the drop, it returns to complete the previous trajectory (blue trajectory) in the opposite direction. Finally, the manipulator moves to its home position (light green trajectory).

In the process of repositioning the "*bottle_1*" in the position in which it was initially observed, the manipulator performed the same trajectories described in the initial process of pick and drop of the "*bottle_1*" only in the inverse process; that is, it starts in the light green trajectory and ends the sequence with the red trajectory, which now goes from the approach point of the drop of the "*bottle_1*" (i.e. initially was the pick approach point of the "*bottle_1*"), to the home position of the manipulator.

The time taken for each set of tasks is shown below; the test was carried out with the manipulator moving at a low speed, i.e. 10% of maximum speed. The execution of the plan had a total duration of 186 s, equivalent to 3 m. The set of tasks executed to move the obstacle to the base of the robot ("*right_platform*") had a duration of approximately 53 s; the set of tasks necessary to perform the pick up of the "*cell_phone_1*" and drop it in the base of the manipulator ("*left_platform*"), had a duration of 60 s; and the tasks necessary for the reposition of the obstacle, in the place initially observed, which had a

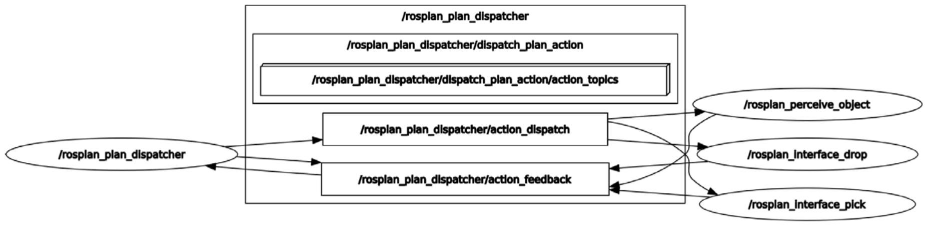

/rosplan_plan_dispatcher
/rosplan_plan_dispatcher/dispatch_plan_action
/rosplan_plan_dispatcher/dispatch_plan_action/action_topics
/rosplan_plan_dispatcher/action_dispatch
/rosplan_plan_dispatcher/action_feedback
/rosplan_plan_dispatcher
/rosplan_perceive_object
/rosplan_interface_drop
/rosplan_interface_pick

Fig. 19. ROSPlan interface actions.

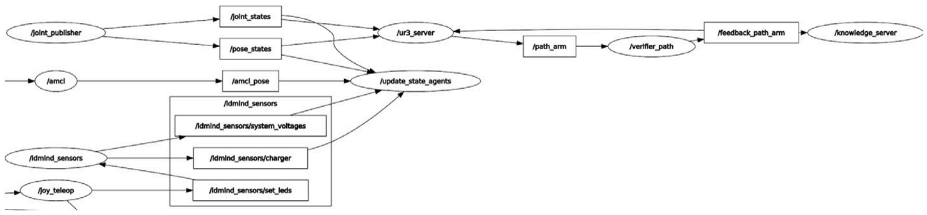
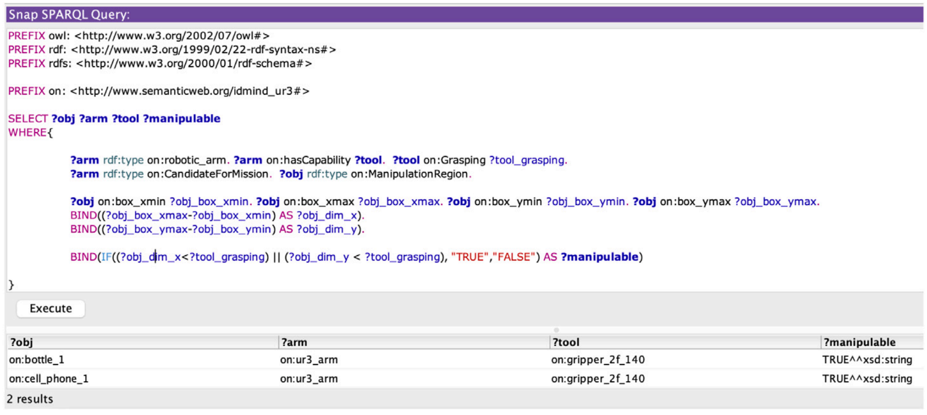


Fig. 20. The topics and nodes created to send and check commands to the manipulator (UR3).



```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

PREFIX on: <http://www.semanticweb.org/idmind_ur3#>

SELECT ?obj ?arm ?tool ?manipulable
WHERE{

        ?arm rdf:type on:robotic_arm. ?arm on:hasCapability ?tool.  ?tool on:Grasping ?tool_grasping.
        ?arm rdf:type on:CandidateForMission.  ?obj rdf:type on:ManipulationRegion.

        ?obj on:box_xmin ?obj_box_xmin. ?obj on:box_xmax ?obj_box_xmax. ?obj on:box_ymin ?obj_box_ymin. ?obj on:box_ymax ?obj_box_ymax.
        BIND((?obj_box_xmax-?obj_box_xmin) AS ?obj_dim_x).
        BIND((?obj_box_ymax-?obj_box_ymin) AS ?obj_dim_y).

        BIND(IF((?obj_dim_x<?tool_grasping) || (?obj_dim_y < ?tool_grasping), "TRUE","FALSE") AS ?manipulable)

}
```

?obj	?arm	?tool	?manipulable
on:bottle_1	on:ur3_arm	on:gripper_2f_140	TRUE^^xsd:string
on:cell_phone_1	on:ur3_arm	on:gripper_2f_140	TRUE^^xsd:string

2 results

Fig. 21. Question. Are the objects present manipulable?.

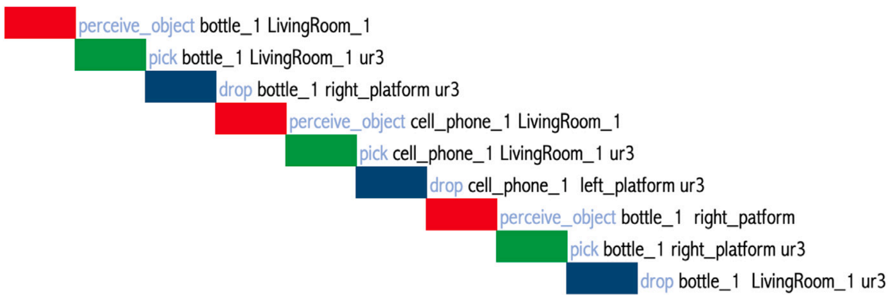

perceive_object bottle_1 LivingRoom_1
pick bottle_1 LivingRoom_1 ur3
drop bottle_1 right_platform ur3
perceive_object cell_phone_1 LivingRoom_1
pick cell_phone_1 LivingRoom_1 ur3
drop cell_phone_1 left_platform ur3
perceive_object bottle_1 right_patform
pick bottle_1 right_platform ur3
drop bottle_1 LivingRoom_1 ur3

Fig. 22. Generated final plan.

duration of 72 s. That is, due to the obstacle, the initial plan had a time cost of approximately 125.64 s, equivalent to 2 m, or 67.52% more time than the initial plan would need.

The framework proved effective in solving the proposed problem; Despite spending 67.52% more time than the initial plan. If an operator had to intervene to divert the obstacle that caused the collision, this cost would be higher. A limitation of the presented framework is that it only works if the objects causing the collision are manipulable. For example, if the object responsible for the collision were not manipulable, our system would send an error message informing us that it is not allowed to reach the goal, being unable to find a solution. This problem could be solved if we change the position of the AMMR's base to a place that will enable the picking of the "*cell_phone_1*". This solution is outside the scope of the work presented here, and will the focus of future work to implement such behaviour using the mobile base capability. This strategy is very promising and is easily transferred to other manipulators, and even can be used by a team of robots. In this last case, and if a manipulator failed to grasp the object, could call, e.g., a mobile manipulator that can do the grasping, if equipped

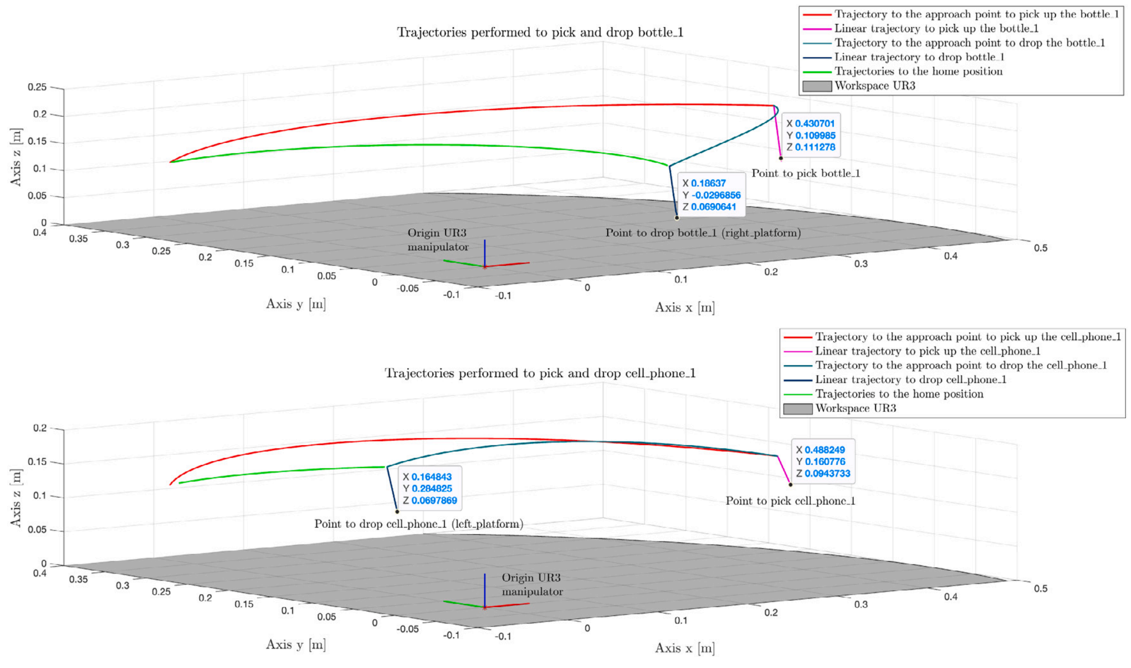


Fig. 23. The paths taken by the robotic arm are represented. The trajectory refers to the centre point of the tool.

with the proper gripper. The presented framework can also be easily integrated with a framework for optimization AMMR agents.

6. Conclusions and future work

This work presented an ontological framework to improve the motion planning process, providing the possibility to reconfigure the initially defined trajectory (e.g. recovery from a situation where an unexpected obstacle appears on the path), for a robotics manipulative agent. Semantic knowledge, and SWRL rules were used to infer new knowledge based on the known environment and the robotic system. For reasoning with the ontology, different SPARQL queries were used. A deep neural network was trained, and used, for the detection and classification of objects present in the environment where the robotic agent is located. Based on this information, a semantic map of the environment was created. Ontological knowledge was then used to improve the motion planning of a manipulator agent by inferring manipulation constraints through reasoning based on logical axioms. Finally, based on the semantic knowledge, the environment corresponding to the agent's manipulation workspace was created so that the planner could infer from it and the knowledge base, to generate a collision-free path.

The proposed framework was implemented in a real-world scenario, and its potential was proven through a manipulation problem. Rule-based prediction of the agent's location in the environment proved to be very robust and effective. Finally, knowledge was inferred based on semantic knowledge, and the ROSPlan framework was used to perform task planning based on the actions defined in the ontology. In summary, the presented framework proved to be a strategy with much potential to improve the planned motion of robotic systems.

In future work, we highlight aspects that we consider relevant to be solved to improve the implemented framework. Using SWRL rules proved to be a promising approach to infer new knowledge. However, the use of rules is still limited due to limitations in rule insertion. For example, implementing a probabilistic extension of SWRL to handle incomplete or partial knowledge would be an advantage in inferring uncertainty and providing a better query service to users, especially when dealing with mobile manipulators. Creating a mechanism to avoid conflicts and redundancies when inserting new rules would also be advantageous. Finally, strategies should be developed following the one presented in this paper, in which, through semantic reasoning/knowledge, the actions performed by AMMR agents are optimized. That is, it is imperative to develop solutions that exploit the capabilities of a AMMR agent, for example, that optimize manipulation tasks based on the movement of the mobile base of the robotic agent, etc.

CRediT authorship contribution statement

Rodrigo Bernardo: Conceptualization, Methodology, Software, Investigation, Validation, Data curation, Writing – original draft, Visualization. **João M.C. Sousa:** Conceptualization, Writing – review & editing, Supervision, Funding acquisition. **Paulo J.S. Gonçalves:** Conceptualization, Methodology, Resources, Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgements

This work is financed by national funds through FCT - Foundation for Science and Technology, Portugal, I.P., through IDMEC, under LAETA, project UIDB/50022/2020. The work of Rodrigo Bernardo was supported by the PhD Scholarship BD\6841\2020 from FCT, Portugal. This work has received funding from: the project 0770_EUROAGE2_4_E (POCTEP Programa Interreg V-A Spain-Portugal), and the European Union's Horizon 2020 programme under StandICT.eu 2023 (under Grant Agreement No.: 951972).

References

Achour, A., Al-Assaad, H., Dupuis, Y., & El Zaher, M. (2022). Collaborative mobile robotics for semantic mapping: A survey. *Applied Sciences*, *12*(20), 10316.

Akbari, A., Muhayyuddin, & Rosell, J. (2019). Knowledge-oriented task and motion planning for multiple mobile robots. *Journal of Experimental & Theoretical Artificial Intelligence*, *31*(1), 137–162.

Akbari, A., Rosell, J., et al. (2015). Ontological physics-based motion planning for manipulation. In *2015 IEEE 20th conference on emerging technologies & factory automation* (pp. 1–7). IEEE.

Akbari, A., Rosell, J., et al. (2018). κ-PMP: Enhancing physics-based motion planners with knowledge-based reasoning. *Journal of Intelligent and Robotic Systems*, *91*(3), 459–477.

Beetz, M., Mösenlechner, L., & Tenorth, M. (2010). CRAM—A cognitive robot abstract machine for everyday manipulation in human environments. In *2010 IEEE/RSJ international conference on intelligent robots and systems* (pp. 1012–1017). IEEE.

Bernardo, R., Sousa, J., & Gonçalves, P. J. (2021). Planning robotic agent actions using semantic knowledge for a home environment. *Intelligence & Robotics*, *1*(2), 116–130.

Bernardo, R., Sousa, J. M., & Gonçalves, P. J. (2022). Survey on robotic systems for internal logistics. *Journal of Manufacturing Systems*, *65*, 339–350.

Bjelonic, M. (2016–2018). YOLO ROS: Real-time object detection for ROS. https://github.com/leggedrobotics/darknet_ros.

Borgo, S. (2019). An ontological view of components and interactions in behaviorally adaptive systems. *Journal of Integrated Design and Process Science*, *23*(1), 17–35.

Borgo, S., & Masolo, C. (2010). Ontological foundations of DOLCE. In *Theory and applications of ontology: computer applications* (pp. 279–295). Springer.

Buric, M., Pobar, M., & Ivasic-Kos, M. (2018). Ball detection using YOLO and mask R-CNN. In *2018 International conference on computational science and computational intelligence* (pp. 319–323). IEEE.

Diab, M., Akbari, A., Rosell, J., et al. (2017). An ontology framework for physics-based manipulation planning. In *Iberian robotics conference* (pp. 452–464). Springer.

Diab, M., Akbari, A., Ud Din, M., & Rosell, J. (2019). PMK—A knowledge processing framework for autonomous robotics perception and manipulation. *Sensors*, *19*(5), 1166.

El-Diraby, T. E. (2013). Domain ontology for construction knowledge. *Journal of Construction Engineering and Management*, *139*(7), 768–784.

Fan, J., Zheng, P., & Li, S. (2022). Vision-based holistic scene understanding towards proactive human–robot collaboration. *Robotics and Computer-Integrated Manufacturing*, *75*, Article 102304.

Fernandez-Chaves, D., Ruiz-Sarmiento, J.-R., Petkov, N., & Gonzalez-Jimenez, J. (2021). ViMantic, a distributed robotic architecture for semantic mapping in indoor environments. *Knowledge-Based Systems*, *232*, Article 107440.

Feyzabadi, S., & Carpin, S. (2014). Knowledge and data representation for motion planning in dynamic environments. In *Robot intelligence technology and applications 2* (pp. 233–240). Springer.

Fiorentini, X., Rachuri, S., Suh, H., Lee, J., & Sriram, R. D. (2010). An analysis of description logic augmented with domain rules for the development of product models. *Journal of Computing and Information Science in Engineering*, *10*(2).

Garg, S., Sünderhauf, N., Dayoub, F., Morrison, D., Cosgun, A., Carneiro, G., et al. (2020). Semantics for robotic mapping, perception and interaction: A survey. *Foundations and Trends® in Robotics*, *8*(1–2), 1–224.

Gomez-Gonzalez, S., Neumann, G., Schölkopf, B., & Peters, J. (2020). Adaptation and robust learning of probabilistic movement primitives. *IEEE Transactions on Robotics*, *36*(2), 366–379.

Goncalves, P., Olivares Alarcos, A., Bermejo Alonso, J., Borgo, S., Diab, M., Habib, M. K., et al. (2021). IEEE standard for autonomous robotics ontology [standards]. *IEEE Robotics & Automation Magazine*, *28*(3), 171–173.

Gualtieri, L., Rauch, E., & Vidoni, R. (2022). Development and validation of guidelines for safety in human-robot collaborative assembly systems. *Computers & Industrial Engineering*, *163*, Article 107801. http://dx.doi.org/10.1016/j.cie.2021.107801.

Guarino, N. (1998). *Formal ontology in information systems: proceedings of the first international conference (vol. 46)*. IOS Press.

Hanheide, M., Göbelbecker, M., Horn, G. S., Pronobis, A., Sjöö, K., Aydemir, A., et al. (2017). Robot task planning and explanation in open and uncertain worlds. *Artificial Intelligence*, *247*, 119–150.

Havard, V., Jeanne, B., Savatier, X., & Baudry, D. (2017). Inoovas-industrial ontology for operation in virtual and augmented scene: The architecture. In *2017 4th international conference on control, decision and information technologies* (pp. 0300–0305). IEEE.

He, Z., Sun, H., Hou, J., Ha, Y., & Schwertfeger, S. (2021). Hierarchical topometric representation of 3D robotic maps. *Autonomous Robots*, 1–17. http://dx.doi.org/10.1007/s10514-021-09991-8.

Horrocks, I., & Patel-Schneider, P. F. (2003). Reducing OWL entailment to description logic satisfiability. In *International semantic web conference* (pp. 17–29). Springer.

Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., Dean, M., et al. (2004). SWRL: A semantic web rule language combining OWL and ruleml. *W3C Member Submission*, *21*(79), 1–31.

Hsu, D., Latombe, J.-C., & Kurniawati, H. (2006). On the probabilistic foundations of probabilistic roadmap planning. *International Journal of Robotics Research*, *25*(7), 627–643.

Jansen, L., & Schulz, S. (2011). The ten commandments of ontological engineering. In *Proceedings of the 3rd workshop of ontologies in biomedicine and life sciences*.

Kanazawa, A., Kinugawa, J., & Kosuge, K. (2019). Adaptive motion planning for a collaborative robot based on prediction uncertainty to enhance human safety and work efficiency. *IEEE Transactions on Robotics*, *35*(4), 817–832.

Karaman, S., & Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, *30*(7), 846–894.

Kitaev, N., Mordatch, I., Patil, S., & Abbeel, P. (2015). Physics-based trajectory optimization for grasping in cluttered environments. In *2015 IEEE international conference on robotics and automation* (pp. 3102–3109). IEEE.

Kockara, S., Halic, T., Iqbal, K., Bayrak, C., & Rowe, R. (2007). Collision detection: A survey. In *2007 IEEE international conference on systems, man and cybernetics* (pp. 4046–4051). IEEE.

Kostavelis, I., & Gasteratos, A. (2015). Semantic mapping for mobile robotics tasks: A survey. *Robotics and Autonomous Systems*, *66*, 86–103.

Lamy, J.-B. (2017). Owlready: Ontology-oriented programming in python with automatic classification and high level constructs for biomedical ontologies. *Artificial Intelligence in Medicine*, *80*, 11–28.

Lasota, P. A., Fong, T., Shah, J. A., et al. (2017). *A survey of methods for safe human-robot interaction (vol. 104)*. The Netherlands: Now Publishers Delft.

Latombe, J.-C. (2012). *Robot motion planning (vol. 124)*. Springer Science & Business Media.

LaValle, S. M., et al. (1998). Rapidly-exploring random trees: A new tool for path planning.

Leidner, D., Borst, C., & Hirzinger, G. (2012). Things are made for what they are: Solving manipulation tasks by using functional object classes. In *2012 12th IEEE-RAS international conference on humanoid robots* (pp. 429–435). IEEE.

Li, C., & Tian, G. (2020). Transferring the semantic constraints in human manipulation behaviors to robots. *Applied Intelligence*, *50*(6), 1711–1724.

Liu, S., & Liu, P. (2021). A review of motion planning algorithms for robotic arm systems. (pp. 56–66). RiTA 2020.

Mascardi, V., Cordì, V., & Rosso, P. (2008). *A comparison of upper ontologies: Technical report disi-tr-06-21*, (p. 16146). Dipartimento di Informatica e Scienze dell'Informazione (DISI), Universitr degli Studi di Genova, Via Dodecaneso, 35.

Moll, M., Kavraki, L., Rosell, J., et al. (2017). Randomized physics-based motion planning for grasping in cluttered and uncertain environments. *IEEE Robotics and Automation Letters*, *3*(2), 712–719.

Mun, D., & Ramani, K. (2011). Knowledge-based part similarity measurement utilizing ontology and multi-criteria decision making technique. *Advanced Engineering Informatics*, *25*(2), 119–130.

Nahavandi, S. (2019). Industry 5.0—A human-centric solution. *Sustainability*, *11*(16), 4371.

Niles, I., & Pease, A. (2001). Towards a standard upper ontology. In *Proceedings of the international conference on formal ontology in information systems (vol. 2001)* (pp. 2–9).

Niloy, M. A. K., Shama, A., Chakrabortty, R. K., Ryan, M. J., Badal, F. R., Tasneem, Z., et al. (2021). Critical design and control issues of indoor autonomous mobile robots: A review. *IEEE Access*, *9*, 35338–35370. http://dx.doi.org/10.1109/ACCESS.2021.3062557.

Nüchter, A., & Hertzberg, J. (2008). Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, *56*(11), 915–926.

O'connor, M., Knublauch, H., Tu, S., Grosof, B., Dean, M., Grosso, W., et al. (2005). Supporting rule system interoperability on the semantic web with SWRL. In *International semantic web conference* (pp. 974–986). Springer.

Olivares-Alarcos, A., Foix, S., Borgo, S., & Alenyà, G. (2022). OCRA–An ontology for collaborative robotics and adaptation. *Computers in Industry*, *138*, Article 103627.

Olszewska, J. I., Barreto, M., Bermejo-Alonso, J., Carbonera, J., Chibani, A., Fiorini, S., et al. (2017). Ontology for autonomous robotics. In *2017 26th IEEE international symposium on robot and human interactive communication (RO-MAN)* (pp. 189–194). IEEE.

Paraschos, A., Daniel, C., Peters, J., & Neumann, G. (2018). Using probabilistic movement primitives in robotics. *Autonomous Robots*, *42*(3), 529–551.

Phanden, R. K., Sharma, P., & Dubey, A. (2021). A review on simulation in digital twin for aerospace, manufacturing and robotics. *Materials Today: Proceedings*, *38*, 174–178.

Pignaton de Freitas, E., Olszewska, J. I., Carbonera, J. L., Fiorini, S. R., Khamis, A., Ragavan, S. V., et al. (2020). Ontological concepts for information sharing in cloud robotics. *Journal of Ambient Intelligence and Humanized Computing*, 1–12.

Polverini, M. P., Zanchettin, A. M., & Rocco, P. (2017). A computationally efficient safety assessment for collaborative robotics applications. *Robotics and Computer-Integrated Manufacturing*, *46*, 25–37.

Prestes, E., Carbonera, J. L., Fiorini, S. R., Jorge, V. A., Abel, M., Madhavan, R., et al. (2013). Towards a core ontology for robotics and automation. *Robotics and Autonomous Systems*, *61*(11), 1193–1204.

Protégé (2022). Protégé. URL https://protege.stanford.edu. (Access June, 2022).

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. ArXiv.

Rickert, M., & Gaschler, A. (2017). Robotics library: An object-oriented approach to robot applications. In *Proceedings of the IEEE/RSJ international conference on intelligent robots and systems* (pp. 733–740). Vancouver, BC, Canada: http://dx.doi.org/10.1109/IROS.2017.8202232.

Rickert, M., Sieverling, A., & Brock, O. (2014). Balancing exploration and exploitation in sampling-based motion planning. *IEEE Transactions on Robotics*, *30*(6), 1305–1317.

Robotics, I., & Society, A. (2015). IEEE standard ontologies for robotics and automation. *IEEE Standard*, *1872*, 1–60.

Rodriguez, S., Tang, X., Lien, J.-M., & Amato, N. M. (2006). An obstacle-based rapidly-exploring random tree. In *Proceedings 2006 IEEE international conference on robotics and automation, 2006* (pp. 895–900). IEEE.

Salzman, O., & Halperin, D. (2016). Asymptotically near-optimal RRT for fast, high-quality motion planning. *IEEE Transactions on Robotics*, *32*(3), 473–483.

Schmidt-Schauß, M. (1988). *Subsumption in KL-ONE is undecidable*. Citeseer.

Schou, C., Andersen, R. S., Chrysostomou, D., Bøgh, S., & Madsen, O. (2018). Skill-based instruction of collaborative robots in industrial settings. *Robotics and Computer-Integrated Manufacturing*, *53*, 72–80.

Sim, R., & Little, J. J. (2009). Autonomous vision-based robotic exploration and mapping using hybrid maps and particle filters. *Image and Vision Computing*, *27*(1–2), 167–177. http://dx.doi.org/10.1016/j.imavis.2008.04.003.

Siméon, T., Laumond, J.-P., Cortés, J., & Sahbani, A. (2004). Manipulation planning with probabilistic roadmaps. *International Journal of Robotics Research*, *23*(7–8), 729–746.

Simões, A. C., Pinto, A., Santos, J., Pinheiro, S., & Romero, D. (2022). Designing human-robot collaboration (HRC) workspaces in industrial settings: A systematic literature review. *Journal of Manufacturing Systems*, *62*, 28–43.

Sirin, E., & Parsia, B. (2007). Sparql-dl: sparql query for owl-dl.. *258*, In *OWLED*.

Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., & Katz, Y. (2007). Pellet: A practical owl-dl reasoner. *Journal of Web Semantics*, *5*(2), 51–53.

Sucan, I. A., Moll, M., & Kavraki, L. E. (2012). The open motion planning library. *IEEE Robotics & Automation Magazine*, *19*(4), 72–82.

Sun, X., Zhang, Y., & Chen, J. (2019). High-level smart decision making of a robot based on ontology in a search and rescue scenario. *Future Internet*, *11*(11), 230.

Toscano, C., Arrais, R., & Veiga, G. (2017). Enhancement of industrial logistic systems with semantic 3D representations for mobile manipulators. In *Iberian robotics conference* (pp. 617–628). Springer.

Trojahn, C., Vieira, R., Schmidt, D., Pease, A., & Guizzardi, G. (2021). Foundational ontologies meet ontology matching: A survey. *Semantic Web*, 1–20, (Preprint).

Tuli, T. B., Kohl, L., Chala, S. A., Manns, M., & Ansari, F. (2021). Knowledge-based digital twin for predicting interactions in human-robot collaboration. In *2021 26th IEEE international conference on emerging technologies and factory automation* (pp. 1–8). IEEE.

Umbrico, A., Cesta, A., & Orlandini, A. (2022). Deploying ontology-based reasoning in collaborative manufacturing.

Umbrico, A., Orlandini, A., & Cesta, A. (2020). An ontology for human-robot collaboration. *Procedia CIRP*, *93*, 1097–1102.

Wang, H., Lv, L., Li, X., Li, H., Leng, J., Zhang, Y., et al. (2023). A safety management approach for industry 5.0′ s human-centered manufacturing based on digital twin. *Journal of Manufacturing Systems*, *66*, 1–12.

Wang, G., Wong, T., & Wang, X. (2010). A negotiation protocol to support agent argumentation and ontology interoperability in mas-based virtual enterprises. In *2010 Seventh international conference on information technology: new generations* (pp. 448–453). IEEE.

Yassin, A., Nasser, Y., Awad, M., Al-Dubai, A., Liu, R., Yuen, C., et al. (2016). Recent advances in indoor localization: A survey on theoretical approaches and applications. *IEEE Communications Surveys & Tutorials*, *19*(2), 1327–1346.

Zhai, Z., Martínez Ortega, J.-F., Lucas Martínez, N., & Castillejo, P. (2018). A rule-based reasoner for underwater robots using OWL and SWRL. *Sensors*, *18*(10), 3481.

Zhao, Y., Fillatreau, P., Elmhadhbi, L., Karray, M. H., & Archimede, B. (2022). Semantic coupling of path planning and a primitive action of a task plan for the simulation of manipulation tasks in a virtual 3D environment. *Robotics and Computer-Integrated Manufacturing*, *73*, Article 102255.

Zheng, L., & Terpenny, J. (2013). A hybrid ontology approach for integration of obsolescence information. *Computers & Industrial Engineering*, *65*(3), 485–499.

Zheng, C., Xing, J., Wang, Z., Qin, X., Eynard, B., Li, J., et al. (2022). Knowledge-based program generation approach for robotic manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, *73*, Article 102242.