Extensions to the Core Ontology for Robotics and Automation

Sandro Rama Fiorini[a,*], Joel Luis Carbonera[a], Paulo Gonçalves[b,c], Vitor A. M. Jorge[a], Vítor Fortes Rey[a], Tamás Haidegger[d,e], Mara Abel[a], Signe A. Redfield[f], Stephen Balakirsky[g], Veera Ragavan[h], Howard Li[i], Craig Schlenoff[j], Edson Prestes[a]

[a] *Instituto de Informática, UFRGS, Brazil*
[b] *Polytechnic Institute of Castelo Branco, School of Technology, Portugal*
[c] *LAETA, IDMEC, Instituto Superior Técnico, Universidade de Lisboa, Lisboa, Portugal*
[d] *Óbuda University, Budapest, Hungary*
[e] *Austrian Center for Medical Innovation and Technology (ACMIT), Wiener Neustadt, Austria*
[f] *Naval Research Laboratory, USA*
[g] *Robotics and Autonomous Systems Division, Georgia Tech Research Institute, USA*
[h] *School of Engineering, Monash University, Sunway Campus, Malaysia*
[i] *Dept. of Electrical and Computer Engineering, University of New Brunswick, Canada*
[j] *Intelligent Systems Division, NIST, USA*

Abstract

The working group *Ontologies for Robotics and Automation*, sponsored by the *IEEE Robotics & Automation Society*, recently proposed a Core Ontology for Robotics and Automation (CORA). This ontology was developed to provide an unambiguous definition of core notions of robotics and related topics. It is based on SUMO, a top-level ontology of general concepts, and on ISO 8373:2012 standard, developed by the ISO/TC184/SC2 Working Group, which defines — in natural language — important terms in the domain of

*Corresponding author

Email addresses: `srfiorini@inf.ufrgs.br` (Sandro Rama Fiorini), `jlcarbonera@inf.ufrgs.br` (Joel Luis Carbonera), `paulo.goncalves@ipcb.pt` (Paulo Gonçalves), `vamjorge@inf.ufrgs.br` (Vitor A. M. Jorge), `vfrey@inf.ufrgs.br` (Vítor Fortes Rey), `haidegger@ieee.org` (Tamás Haidegger), `marabel@inf.ufrgs.br` (Mara Abel), `signe@ieee.org` (Signe A. Redfield), `stephen.balakirsky@gtri.gatech.edu` (Stephen Balakirsky), `veera.ragavan@monash.edu` (Veera Ragavan), `vhoward@unb.ca` (Howard Li), `craig.schlenoff@nist.gov` (Craig Schlenoff), `edson.prestes@ieee.org` (Edson Prestes)

Preprint submitted to Robotics and Computer-Integrated Manufacturing August 13, 2014

Robotics and Automation (R&A). In this paper, we introduce a set of ontologies that complement CORA with notions such as industrial design and positioning. We also introduce updates to CORA in order to provide more ontologically sound representations of autonomy and of robot parts.

Keywords: Ontologies for robotics and automation, Ontology-based standards, Core ontology, Ontology engineering, Knowledge representation.

---

1. Introduction

A well-structured *body of knowledge* for robotics and automation (R&A) is a crucial requirement not only for unambiguous communication and reasoning for robots, but also for knowledge and information sharing about robots among humans and for interaction between robots and humans. Recently, such bodies of knowledge have been successfully developed using ontologies. Ontologies are information artifacts that specify in a *formal* and *explicit* way the domain knowledge *shared* by a community [1]. The availability of well-founded methodologies allow us to develop ontologies in a principled way. The artifacts that result from this process ensure mutual agreement among stakeholders, increase the potential for reuse of the knowledge, and promote data integration.

In order to specify and clarify the meaning of the core notions common in R&A, the Working Group (WG) *Ontologies for Robotics and Automation* (ORA), sponsored by the *IEEE Robotics & Automation Society*, have proposed a *Core Ontology for Robotics and Automation* (*CORA*). This ontology is meant to be used by robots and roboticists in tasks that require explicit knowledge about robots, such as robot-robot and robot-human communication, robot design, and integration of data about robots. The aim of the ORA WG is to standardize knowledge representation in the R&A field [2]. Within this broad context, CORA is intended to provide the core conceptual structure that will integrate other specific ontologies developed for the domain of R&A.

CORA has been developed taking into account theories of the discipline of Formal Ontology [3]. In particular, many of our ontological choices were evaluated based on guidelines from known methodologies, such as METHONTOLOGY [4] and OntoClean [5]. Besides that, CORA was developed based on SUMO [6]; a top-level ontology that aims to define the main ontological categories describing the world. Such an approach is new in developing standards in R&A and has the advantage of producing a better founded standard, which requires less work to use, maintain and extend.

This work reports the recent developments within the ongoing CORA project, and provides an overview of its current state. The prior version of CORA [7] has been extended, implementing changes in modeling decisions and introducing new concepts and relations. Thus, this paper presents some changes in modelling decisions that have been implemented since the previous version. The major new contributions can be divided into two broad areas. First, we propose CORAX, an ontology that covers concepts too general to be part of CORA, and that are not covered by SUMO. These include knowledge about *design* (as in the case of product design), *physical environment*, *interaction*, and *artificial systems*. Second, we propose extensions and changes to CORA itself, in order to improve its ontological commitment to the domain. We are primarily concerned with representation of *operation modes* and *robot parts*. Finally, we discuss some directions regarding new, yet to be covered topics (such as control and planning).

2. Ontology Engineering

We developed CORA using several ontology tools and frameworks. The main methodology is based on METHONTOLOGY [4], which supports the development of ontologies either from scratch, by reuse, or by re-engineering existing ones. It consists of a set of guidelines about how to carry out the activities identified in the ontology development process, the kinds of techniques that are the most appropriate for each activity, and the resulting products.

We also based many of the underlying *ontological commitments* on *OntoClean* [5]. Ontoclean is a methodology for validating the ontological adequacy of taxonomic relationships, based on highly generic ontological notions drawn from philosophy, like *essence*, *identity* and *unity*. These notions are used to characterize relevant aspects of the intended meaning of the properties, classes, and relations that compose an ontology. OntoClean requires the ontology engineer to explicitly identify the ontological commitments underlying the concepts that are being modelled. As a result, OntoClean allowed us to identify ambiguities in the definitions of core notions provided by other standards of R&A (see [7] for more details).

In addition, as a result of an evaluation process carried out in [7], we selected the *Suggested Upper Merged Ontology* (SUMO)[1] [6] as the most suitable top-level ontology for supporting the development of CORA. SUMO was developed by an IEEE working group, and according to our analysis, it is flexible enough to fit the purposes of the project. It includes the main notions and distinctions we would like to introduce in our ontology, such as *agent*, *device* and *agent group*. All concepts in CORA and related ontologies are specializations of concepts in SUMO.

SUMO defines the basic ontological categories across all domains. The remainder of this section gives a brief overview of its main concepts, illustrated in Fig. 1. Detailed information can be found in [6].

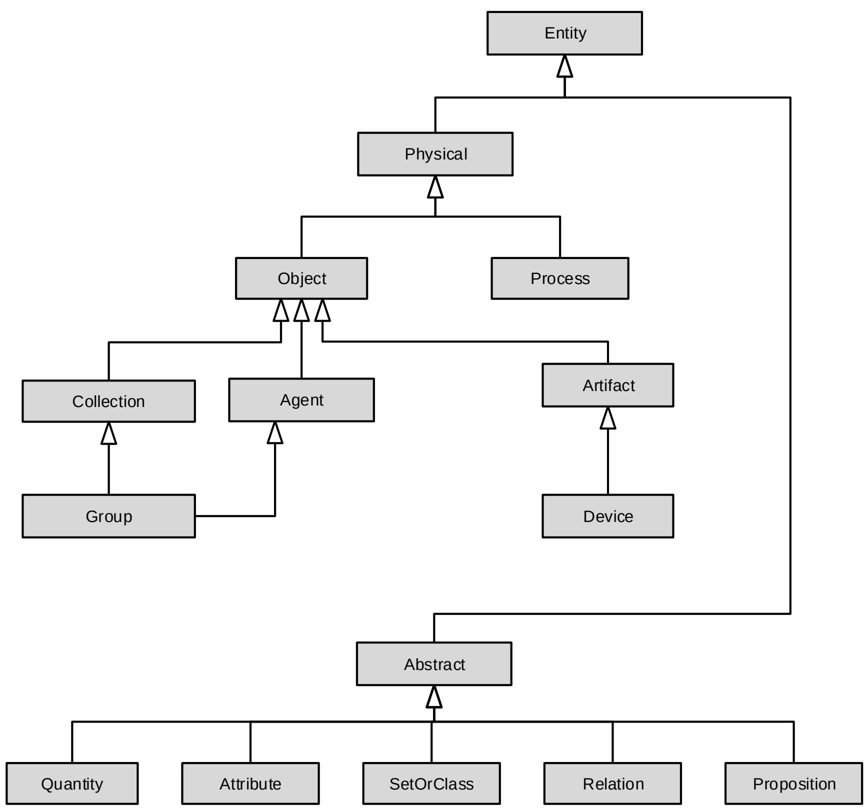

Figure 1: Overview of top-level concepts of SUMO.

[1] `http://www.ontologyportal.org/`

The main SUMO category is *Entity*, which is a disjoint partition of *Physical* and *Abstract* entities. Physical represents entities that have a location in space-time. Abstract describes entities that do not have a location in space-time.

Physical is further partitioned into *Object* and *Process*. Object exists in space, keeping its identity in time, and has spatial parts but not temporal parts. Process is the class of instances that happen in time and have temporal parts or stages. This means SUMO follows an *endurantist* perspective instead of a *perdurantist* one. For a perdurantist, an object is composed by every temporal part it has at all times. On the other hand, for an endurantist, an object changes through time, but keeps the essential parts that define its identity. A good analogy is to think that perdurantists see objects as tunnel-like regions in a 4D space, while endurantists see them as a 3D region that travels through the time dimension.

Abstract is further partitioned into *Quantity, Attribute, SetOrClass, Relation* and *Proposition*. Quantity abstracts numeric and physical quantities. Attribute abstracts qualities that cannot or are chosen not to be considered as subclasses of Object. SetOrClass abstracts entities that have *elements* (in the case of sets) or *instances* (in the case of classes). Relation generalizes n-ary relations, functions and lists. Finally, Propositions are entities that express a complete thought or a set of such thoughts.

3. Overview of CORA

CORA aims to describe what a robot is and how its concept relates to other concepts. It defines three broad entities: *robot*, *robot group* and *robotic system* (Fig. 2). In this paper, we are not going to delve into the details of each concept, since they were presented in [7]. Instead, we provide a short description of each domain entity.

The term *robot* may have as many definitions as there are people writing about the subject. This inherent ambiguity in the term might be an issue when specifying an ontology for a broad community. We, however, acknowledge this ambiguity as an intrinsic feature of the domain, and therefore have decided to use a definition based purely on necessary conditions, without specifying sufficient conditions. Thus, our goal is to ensure that CORA's definition of robot includes most of the entities that the community actually considers as robots, at the cost of classifying as robots some entities that actually would not be considered as robots in the point of view of

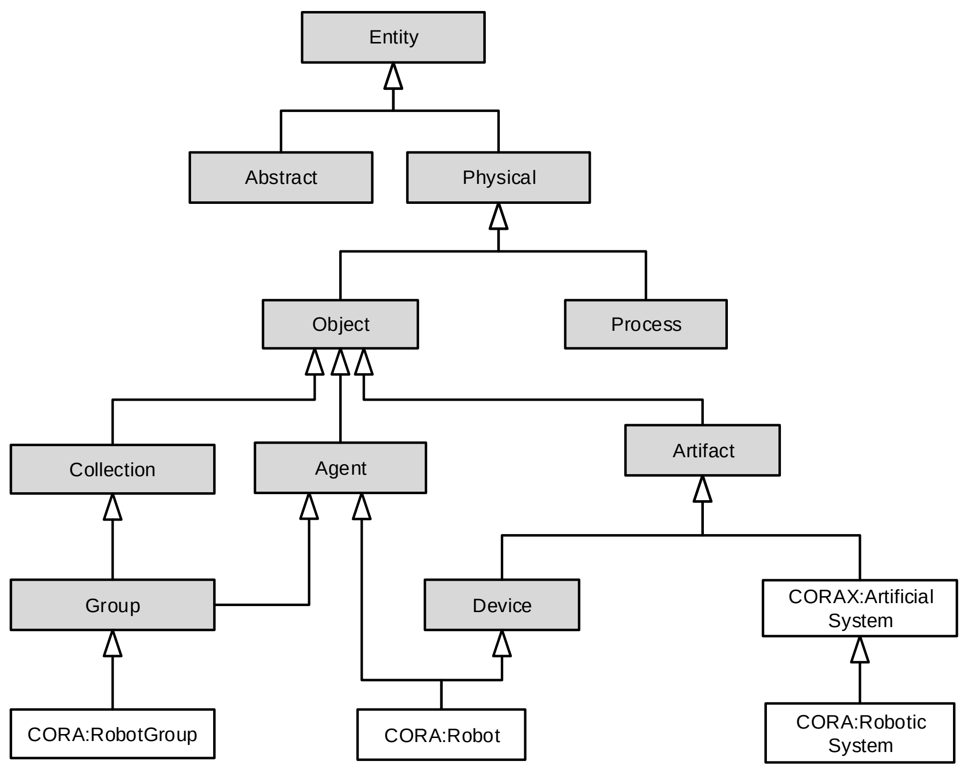

Figure 2: Overview of the main concepts in CORA: *robot*, *robot group* and *robotic system*

some roboticists. However, the concepts in our ontology could be extended according to the needs of specific sub-domains or applications of R&A.

More importantly, our definition of robot emphasizes its functional aspects. For our general purposes, *robots are agentive devices* in a broad sense, designed to perform purposeful actions in order to accomplish a task. In some cases, the actions of a robot might be subordinated to actions of other agents, such as software agents (bots) or humans. Robots are also *devices*, composed of suitable mechanical and electronic parts. Robots can form *social groups*, where they interact to achieve a common goal. A robot (or a group of robots) can be combined with other devices to form robotic systems. An environment equipped with a robotic system is a robotic environment.

A *robot* is a *device* in the sense of SUMO. According to SUMO, a device is an artifact (i.e., a *physical object product of making*), which participates as a tool in a process. Being a device, robot inherits from SUMO the notion that devices have parts. Therefore, CORA allows one to represent complex robots with robot parts.

A robot is also an *agent*. SUMO states that agent is "*something or someone that can act on its own and produce changes in the world*". Robots perform tasks by acting on the environment or themselves. Action is strongly related to agency, in the sense that the acting defines the agent. A robot can form robot groups. A *robot group* is also an agent in the sense that its own agency emerges from its participants. This notion can be used to describe robot teams, or even complex robots formed by many independent robotic agents acting in unison.

Robotic systems are systems composed of robots (or robot groups) and other devices that facilitate the operations of robots. A good example of a robotic system is a car assembly cell at a manufacturing site. The environment is equipped with actuated structures that manipulate the car body in a way that the industrial robots within the system can act on it. Finally, as previously stated, an environment equipped with a robotic system is a *robotic environment*. See [7, 8] for a more detailed discussion on CORA's main concepts. Next, we describe new notions that have been integrated into CORA.

4. Updating CORA

CORA has been updated since its initial proposal in [7, 8]. The main driving force behind these changes came from aligning it with existing ontologies and more expert involvement in the development process. We compared CORA with an *ontology for kitting* developed within the group [9].This enabled us to investigate whether or not both ontologies could be merged, and to check whether all notions in the kitting ontology were represented in the combination of SUMO and CORA. We found that important concepts and relations present in the kitting ontology were not covered. Due to this, we developed new ontology modules to bridge the gap between SUMO and the kitting ontology, which are mostly covered by CORAX and the POS ontologies.

Furthermore, after the preliminary draft standard was completed, we experienced increased involvement of independent experts and received additional feedback. Apparently, experts were more comfortable discussing concepts and relations, after a first set of ontological commitments were made and the scope of the project was established. The initial model served as a reference to articulate new requirements on the ontology. Since the initial model was based on well-founded ontological commitments, the model was more resilient to ad-hoc proposals to change it, translating into a more stable evolution of the ontology. Notably, changes were more prominent in aspects of the ontology that had a less solid foundation in the first version of the ontology, such as autonomy.

In the following sections, we describe the changes made in and around CORA as a result of that process. They consist mostly of sub-ontologies complementing or extending CORA (see Fig. 3).

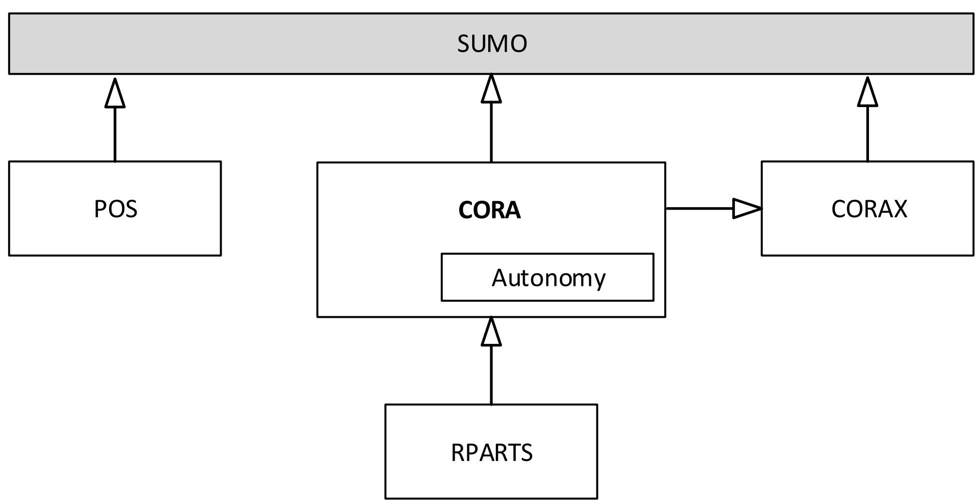

Figure 3: Extensions made to CORA and SUMO. CORAX, POS and RPARTS are extensions made to SUMO and CORA. The way CORA represents autonomy was also updated.

5. CORAX: connecting CORA and SUMO

Naturally, SUMO does not cover every possible aspect of reality, even when we restrict ourselves to R&A. At the same time, some of parts of reality are too general to be included in CORA. We introduced the CORAX ontology to address this problem by bridging SUMO and CORA. In particular, CORAX includes concepts and relations associated with design, interaction, and environment, which are not covered in SUMO.

5.1. Design

Design is an important concept in engineering, specially in manufacturing. In R&A, the concept is frequently related to industrial robotics, where robots perform the job of building artifacts. Those robots have to know the design of the artifacts they are building in order to coordinate their actions.

A design is an abstract entity; it does not have materiality in itself. Rather, *content-bearing objects* (in SUMO), such as manuals and blueprints, give materiality to a design. One could reason this in another way: a design is what links a series of related *blueprints*; it is the common abstract content that is represented in different blueprints. Furthermore, an artifact is related to a particular design, so that one should expect that the *artifact* realizes the design.

From our point of view, SUMO does not provide a good specification of design. One of its sub-ontologies—namely the engineering ontology—defines the concept *Model*, which is an abstract entity that seems to capture the notion of design described above. However, a model is not clearly related to content bearing objects, or to *artifacts* in general. SUMO defines a relationship called *models*, which is held between *Model* and *Engineering Component*. However, this relationship is too restrictive for our purposes, since we would like to represent models of any kind of artifact.

In response to this, we defined the concept of *Design*, which is a kind of *Proposition*. According to SUMO, a *proposition* is an abstract entity that expresses a complete thought or a set of thoughts. For instance, the phrases "*the cat is on the mat*" and "*o gato está no tapete*" express the *same proposition* in English and in Portuguese, respectively. In much the same way, different *blueprints* might express the same *design*.

Furthermore, the properties of the object must be expressed in its design. For instance, the design of a phone is about an ideal (*idealized*) phone that is materialized in the individual realizations of the design. This ideal phone has ideal properties, such as ideal weight and shape. There are many ways of representing an idealized object within an ontology. For instance, one could represent it as a special instance of the concept *Phone*, called prototype. Another alternative is to collapse both the design and the ideal object into the same entity. This is exactly the approach that was adopted in the design ontology that is presented in [10], which is also based on SUMO. However, since the ideal object is also a proposition, there might be issues when modelling its attributes and parts. For instance, if both the design of a phone and the ideal phone (the content of the design) are the same entity, this entity,

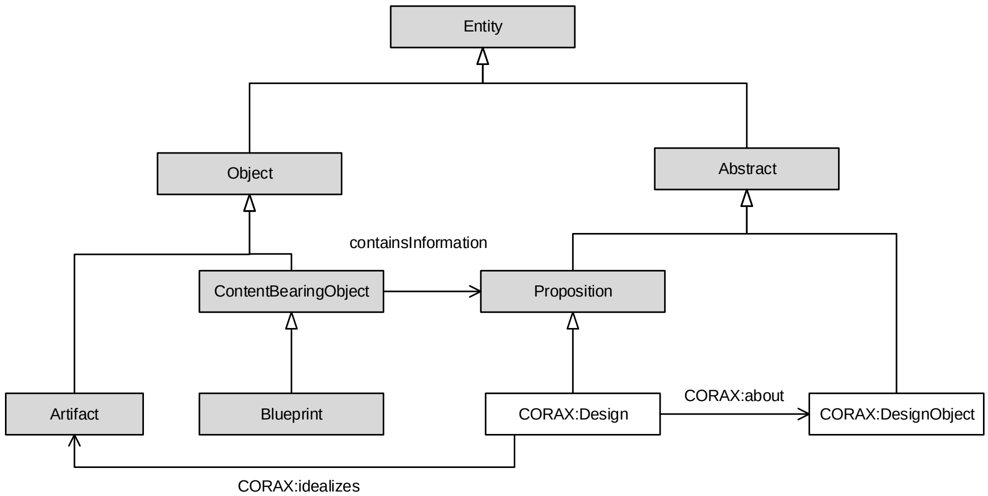

Figure 4: Entities associated with Design in CORAX.

as a proposition, will have a designed color and a designed shape. However, *a proposition cannot have a color or a shape*. Thus, we model the ideal object as a separate abstract entity called *Design Object*, which specifies the idealized object that is the *content* of a Design. We believe this definition better matches the experts' intuitive notion of an engineering model; it also eliminates the need for a new metacategory in SUMO (such as prototype). As with physical objects, design objects have properties such as weight and shape. SUMO provides two main relations to represent properties, namely *attribute* and *measure*, but these can only predicate physical objects. We therefore created the relations *designAttribute* and *designMeasure*, which are analog to attribute and measure in SUMO, allowing the reuse of their domain values. In this way, we can specify that, for instance, an idealized phone (an instance of *Object Design*) has a *design shape* and a *design weight*.

Designs *idealize* artifacts (therefore, the relation *CORAX:idealizes* in Figure 4). It is important to note that it is the *design* that idealizes the artifact, and not the design object. The properties of the design object and those of the artifact may correlate, but we will not provide a theory about how this correlation occurs at this stage.

5.2. Physical Environment

Another important notion missing in SUMO is that of *physical environment*. We added this concept to CORAX in order to support specification of *robotic environments*. In our view, an *environment* is intuitively composed of a physical region, plus other eventual physical entities that characterize the environment. In addition, the definition of physical environment depends on the presence of a landmark (another physical entity) from which it is possible to define the main region of an environment. Landmarks may or may not be located within the region of interest of the environment. For instance, an office room environment depends on the physical configuration of its walls, which are located in the environment. But we can also define an arbitrary environment consisting of a cube in outer space that depends on Earth as a landmark. In this case, Earth does not need to be located within or at the borders of the region.

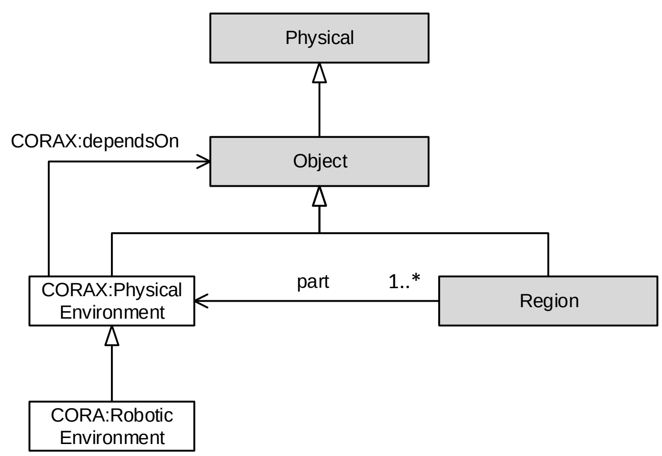

Figure 5: Concepts and relations of Physical Environment in CORAX.

More formally, we define a physical environment in CORAX as a physical object that has at least one region as *part* and that depends on another entity. All other physical objects that are part of an environment must be located within a region that is part of the environment.

5.3. Interaction and Artificial Systems

In order to properly define a *robotic system*, we have to specify what is an *artificial system*. An artificial system is simply an artifact formed from various devices and other objects that interact with each other and with the environment in order to fulfill a function.

This requires a basic definition of *interaction*. We define interaction as a process in which two agents participate, where an *action* generated by one agent causes a *reaction* by the other. More specifically, an interaction process is composed by two sub-processes corresponding to action and reaction. The action sub-process initiated by x on y causes a reaction sub-process, where y acts upon x.

6. CORA: Autonomy revisited

Autonomy is one of the most important terms in R&A, yet one of the hardest to define precisely. In the previous version of CORA, we advocated for a flexible definition that — while not being precise — could distinguish between robots that were clearly autonomous from others with questionable autonomy. In CORA, it has now been pushed a step further in order to make the modelling more versatile.

In this new version, our definitions are aligned with those from the ALFUS [11] framework, which was the result of an extensive study on autonomy in unmanned vehicles. In short, ALFUS states that autonomy is generally dependent on the *degree of human intervention* and *context*, where the latter is characterized by *type of mission* and *environment*.

CORA's definition of autonomy is closely related to what ALFUS defines *modes of operation for unmanned systems*. These modes stretch from fully autonomous to remote controlled, representing the degree of human interaction needed for the robot to perform its task. In our view, they encapsulate the experts' intuitive notion of autonomy in R&A[2]. More specifically, CORA includes:

***Fully autonomous robots*:** A role for a robot performing a given task, in which the robot solves the task without human intervention, while adapting to operational and environmental conditions.

***Semi-autonomous robot*:** A role for a robot performing a given task, in

[2]ALFUS goes a step further in trying to characterize absolute levels of autonomy, which correlates with the modes of operation presented here. However, the exact nature of this relation is not clarified.

which the robot and a human operator jointly plan and conduct the task, requiring various levels of human interaction.

***Teleoperated robot*:** A role for a robot performing a given task, in which a human operator either directly controls the actuators using sensory feedback, or assigns incremental goals on a continuous basis. A teleoperated robot will complete its last command after the operator stops sending commands, even if that command is complex and time-consuming.

***Remote controlled robot*:** A role for a robot performing a given task, in which the human operator controls the robot on a continuous basis, from a location off the robot via only her/his direct observation. In this mode, the robot takes no initiative, and relies on continuous, or nearly continuous input from the human operator.

***Automated robot*:** A role for a robot performing a given task, in which the robot acts as an automaton, following pre-defined (scripted) plans, not adapting to changes in the environment.

It is important to note that *automated robot* is not part of ALFUS' modes of operation. Experts in our groups determined that certain robots require little human interaction, but at the same time are too simple to be characterized as autonomous. This is the case of automatons, including automated dolls and toys, which *cannot* react to changes in environment. Relatively simple code scripts or mechatronics determine the behavior of these robots.

One could mention at this point that some robots are inherently autonomous, or at least, are made with this purpose in mind. Therefore, autonomy would not depend on context. Indeed, there is a correlation between purpose and physical capabilities of a robot, and the modes of operation it can achieve in certain tasks. Yet, this is not the definitive factor in how the robot will operate during its lifetime. It only means that such a robot *can* play a role of autonomous robots.

The fact that this classification of autonomy is context-dependent also affected our modelling choices. In a modelling sense, a mode of operation is a *role*. A role can predicate a given entity at a given time, but it can cease to predicate it at a later time. For instance, the canonical example of role is *Student*: one can predicate a person as a student at a given time, and later cease to do so. This contrasts with rigid types, such as *Person*. Someone cannot cease to be a person without ceasing to exist. In general, a role is also dependent on another entity. For instance, a person must be enrolled at an educational institution in order to be predicated as a student.

A modeler can specify roles in many ways. The earlier version of CORA specified the various modes of operation as concepts. However, SUMO does not support roles as concepts (contrary to other ontologies [3]). For that reason, we modified the modelling of operational modes so that they became a specific type of relation present in SUMO, namely *Case Role*.

A case role in SUMO is a *relation* between an entity and a process. It describes a role that an entity plays in the process in which it participates. In order to define autonomy levels as case roles, we specialized the relation *agent* present in SUMO into the relation *robotAgent*. The relation agent links entities to the processes where they have an "active determinant" behavior. The relation *robotAgent* applies to robots and the processes in which the robot is the active determinant. A given operational mode depends on the way a robot determines the outcome of the processes it is involved in. We represent the operational modes as subrelations of robotAgent: *fullyAutonomousRobot*, *semiAutonomousRobot*, *teleoperatedRobot*, *remoteControlledRobot* and *automatedRobot*. When a particular robot assumes a particular operational mode for a particular task, it is predicated with the appropriate relation. For instance, a robot that can drive autonomously, assumes the role *fullyAutonomousRobot* for the autonomous driving process. The same robot can assume different operational modes in different processes, depending on the context. Interestingly, since processes can have sub-processes, a robot can assume different roles for different sub-processes. For instance, a cleaning robot might be fully autonomous as it detects dirty places to clean, but simultaneously be semi-autonomous with respect to planning routes around the house, or vice versa.

7. RPARTS: Robot parts and extensibility

RPARTS is a sub-ontology of CORA that specifies the notions related to specific kinds of robot parts.

According to CORA, robots are (agentive) devices *composed of* other devices. A myriad of devices can be robot parts, and we cannot determine in advance what *kinds* of devices can or cannot be robot parts. Notice that this is an issue that arises at the *conceptual level*. This is a consequence of the "open-ended" nature of robots, whose designs are only constrained by human needs, human creativity and available technological resources. Therefore, a type of device that has never been considered as a potential robot part can be used as a robot part by some future designer. An ontology for R&A, as CORA is, must take this issue into account.

Furthermore, there is another issue regarding the notion of robot parts that arises at the *instance level*. According to our analysis, none of the instances that can be classified as robot parts are *essentially* robot parts, since they can exist by themselves when they are not connected to a robot (or when they are connected to other complex devices). For instance, a power source is essentially a device, and we cannot consider power source as a subclass of the class of robot parts, because this would imply that all instances of power sources are always robot parts. This is not true, since a specific instance of power source can be dynamically considered as a part of different complex devices during different specific time intervals. Due to this, CORA assumes that the notion of "robot part" is a *role* (in the sense previously discussed) that can be played by other devices.

In the earlier version of CORA [7], the notion of robot part was considered as a *class*, whose instances are not *essentially* instances of it. Thus, instances of robot part could cease to be robot parts, without ceasing to exist. In this sense, for example, an instance of power source that is considered as a robot part at a given moment (when it is connected to a robot) could cease to be a robot part in another moment without ceasing to exist (as an instance of power source). Thus, *Robot part* was considered as an *anti-rigid* class, in the sense of [5, 3]. Our modelling pattern [7] was developed accordingly, inspired by [3]. It represents how a specific instance of a specific kind of device (e.g., power source) could be classified as a robot part.

This pattern becomes complex when we take into account the principles advocated in [5, 3]. According to these frameworks, an anti-rigid class (e.g., robot part) cannot subsume a rigid one (e.g., power source). Considering this principle, for each rigid class c that can play the role of robot part, we must create another specific anti-rigid class (a specific role) that will be subsumed by both c and *Robot Part*. For example, an instance of the rigid class *Wheel* only becomes a robot part when it is attached to a particular robot. Given this condition, it becomes a member of the more specific class (e.g., "*Wheel as Robot Part*"), which is subsumed by the rigid class *Wheel* and the anti-rigid class *Robot Part* (see [7] for further details.)

The representation of robot parts in the new edition of CORA was changed, mainly because the modelling pattern proposed for representing robot parts results in domain models that are overwhelmingly complex. Some classes that must be created in order to maintain the consistency of the model do not fit well into the domain conceptualization, and the resulting complexity is hard to manage. Therefore, this modelling pattern could hinder the broad adoption of the ontology in the domain. Another factor leading to the revision was that it is not clear how to fit the dynamical behavior that is expected from roles in the framework of SUMO. The modelling of roles adopted in [5, 3] relies on the notion of *possibility* (a *modal* notion). However, as pointed out in [12], the treatment of possibilities in SUMO is not clear.

In the current version of CORA, we have modeled the notion of robot part as a relationship between a given device d and a robot r, indicating that d is playing the role of robot part when it is connected to r. During the analysis of the domain literature, we identified some specific types of parts that are important to distinguish within the notion of robot part. These types of parts — according to our analysis — would be different sub-roles of robot part, which could be played by devices with specific features. Thus, robot parts in CORA can be:

***Robot sensing part*:** responsible for sensing the surrounding environment. Formally, robot sensing parts must be measuring devices connected to the robot. A measuring device, according to SUMO, is *any device whose purpose is to measure a physical quantity*. For example, a *laser sensor* can play the role of robot sensing part, when connected to a robot.

***Robot actuating part*:** responsible for allowing the robot to move and act in the surrounding environment. Formally, robot actuating parts must be devices that are instruments in a process of robot motion, which is any process of movement where the robot is the agent and one of its parts is acted upon.

***Robot communicating part*:** responsible for providing communication among robots and humans, by allowing the robot to send (or receive) information to (or from) a robot or a human.

***Robot processing part*:** responsible for processing data and information. Formally, robot processing parts must be processing devices connected to the robot. A processing device is any electric device whose purpose is to serve as an instrument in a subclass of computer process.

It is important to emphasize that although these different types of robot parts are modeled as relations between specific devices and robots, they are intended to behave as roles.

This modelling choice also provides interesting modularity characteristics. It keeps CORA as a minimal core of high-level concepts that provide the structure to the domain without going deep into details regarding the myriad of different devices that could play the roles specified here. In this sense, this structure of roles can be viewed as an interface (in the sense of *object oriented programming paradigm*) that can be implemented in different ways. Naturally, this schema poses the need for sub-ontologies to define the taxonomies of devices that can play the roles specified in CORA, such as an *ontology of sensors, ontology of grippers*, etc.

8. POS: Position, orientation and pose

The position (POS) ontology is an ontology that extends SUMO and complements CORA. POS was developed for capturing the main concepts and relations underlying the notions of *position*, *orientation* and *pose*. These are essential for dealing with information about the relation between the robot and its surrounding space. In this section, we summarize the main concepts relating to positional information. Figure 6 presents an overview of some of the main notions captured in POS, showing their relationships with concepts of SUMO.

According to the literature, roboticists and other domain experts usually utilize two kinds of positional information [13]: *quantitative* and *qualitative*. In the quantitative case, a position is represented by a *point* in a given coordinate system. In the qualitative case, a position is represented as a *region* defined as a function of a reference object. For instance, one can describe a robot as being positioned at the coordinates (x, y) in the global coordinate system, or that the robot is positioned *at the front of the box*, where "front" comprises a conical region centered on the box and pointed forward.

We consider that a *position* can be attributed to a (physical) *object*. In this sense, when we say that "a robot x is positioned at y", this means that there is a *measure* that relates a given "robot x" to a *position measurement* y.

Position measurements are *physical quantities* that can be *position points* or *position regions*. A position point refers to a point in a *coordinate system*

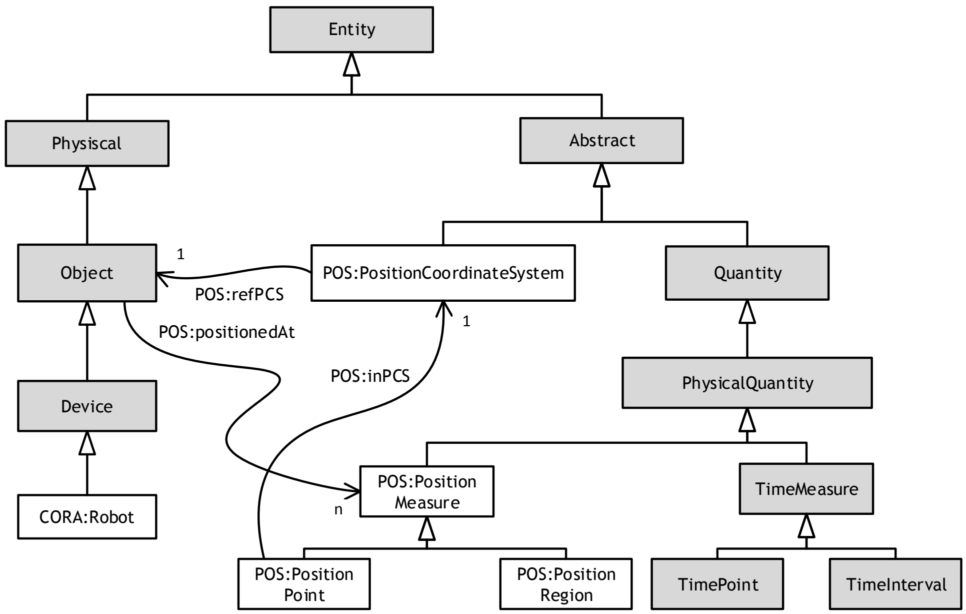

Figure 6: Fragment of POS ontology, presenting the main concepts and relations underlying the notion of *position*.

projected on the physical space. A position region is an *abstract region* in a *coordinate system* defined with reference to a series of position points.

A position point denotes the *quantitative* position of an object in a coordinate system. More specifically, position points are always defined in a single coordinate system.

A *coordinate system* is an *abstract* entity that is defined in relation to *a single reference* object, i.e., there is an object that is the reference for each coordinate system. For instance, the local coordinate system of a robot is referenced by the robot itself. Additionally, the reference object does not need to be at the origin of the coordinate system.

This ontology does not commit to a particular kind of coordinate system. It can be stated however, that a coordinate system defines at least one dimension in which points get their coordinate values. An n-dimensional coordinate system, c, is homeomorphic to a subset of $\mathbb{R}^n$, such that a coordinate $p \in c$ can represented as n-tuple $\phi(p) = (x_1(p), x_2(p), \ldots, x_n(p))$. The functions $x_1, x_2, \ldots, x_n$ are coordinate functions that attribute to p a real value in the dimension n of the coordinate system [14].

A fundamental aspect of coordinate systems is the notion of *transformation*, which maps position points in one coordinate system to position points in another coordinate system. Transformations can be composed generating new transformations. In our ontology, an object can display multiple positions in different coordinate systems only if there is a transformation that can map between the two.

In addition, coordinate systems are related through *hierarchies* (i.e. trees). We say that a given coordinate system c_1 is parent of a coordinate system c_2 if there is a transformation t_1 that maps the points of c_1 to points in c_2, and there is a transformation t_2 that maps the points of c_2 to points in c_1. According to this, if two coordinate systems share a parent node in the hierarchy tree, there is a transformation between them. Usually, an agent chooses a coordinate system as the global reference frame that constitutes the *global coordinate system* (GCS) for that agent. This GCS can be *arbitrarily* chosen and does not have reference to a particular coordinate frame. *Local coordinate systems* (LCS) are defined in relation to GCS by hierarchical links. This hierarchy is arbitrary, in the sense that it can be defined by the designer or agent.

As already stated earlier, besides the quantitative position, our ontology also provides concepts about qualitative positions that are defined in terms of position regions. Example of qualitative positions are "left of", "in front of", "on top of", etc. These expressions define regions in relation to a reference object o_r in which other objects are placed. More specifically, a *position region* is composed of poses in the coordinate system generated by a *spatial operator* on the reference object. The spatial operator is a *mathematical function* that maps reference objects to regions in a coordinate system in arbitrary ways.

Our ontology also allows for the representation of *relative positions* of objects with respect to a given reference object. In general, this kind of information is represented through *spatial relations* that hold between objects. An example of this kind of information is the relation $\mathrm{leftOf}(o, o_r)$, which represents that the object o is positioned to the left of the object o_r. This kind of relation can be defined in our framework using the notions of *relative position* and *spatial operator*. For example, the relation $\mathrm{leftOf}(o, o_r)$ holds when there is a qualitative position s (a position region) that was generated by the spatial operator leftOfOp over the reference object o_r, and the object o has the relative position s regarding o_r. Through this mechanism, our ontology provides the semantics for spatial relations like “to the left of”.

The usual notion of orientation is similar to position as far as its conceptual structure is concerned. Due to this, we will provide only a brief overview. An object can have a quantitative orientation defined as a value in an orientation coordinate system, as well as a qualitative orientation defined as a region in relation to a reference object. For example, orientation is used in the phrase “the robot is oriented at 54 degrees”; the orientation value in this case is 54 in the circular, one-dimensional coordinate system of a compass. On the other hand, orientation regions capture a less intuitive notion. The expression “the robot is oriented to the north of the Earth” allows for interpretations where the robot has a range of possible orientation points around 0 degrees. Thus, we model “north” as a region (or interval) in the one-dimensional compass coordinate system that overlaps with the general orientational extension of the object.

A position and an orientation constitute a pose. The pose of an object is the description of any position and orientation simultaneously applied to the same object. Often, a pose is defined with a position and an orientation referenced to different coordinate systems/reference objects. In addition, since objects can have many different positions and orientation, they can also have many different poses.

It is important to note that the current version of the POS ontology is *synchronic*. That is, it considers only facts about a single time point, just like a snapshot in time. One of the future extensions to this ontology will consider dynamic world modelling, eventually producing a *diachronic* version of the POS ontology.

9. Discussion

The importance of information sharing in R&A emphasizes the necessity of standardization in the field. These standards must be *clear*, *precise* and *easy to use*. CORA is designed to meet that need: it specifies the central concepts of R&A and related fields. In this paper, we presented new additions to CORA and its adjoint domains, providing concepts about positioning, autonomy (including modes of operation), and interaction. These can already be used for building more detailed sub-domain ontologies and algorithms.

Several scenarios could take advantage of CORA (and the related ontologies) in R&A. Firstly, CORA can be immediately applied in *meaning negotiation* among roboticists. That is, our ontologies could be used as *reference* *conceptual models* for ensuring mutual agreement among humans regarding the meaning of concepts of R&A domains.

Moreover, used as a *software component*, the ontology can naturally be applied for enhancing communication among (heterogeneous) robots, as well as among robots and humans. For example, a straightforward application for CORA is as a tool developing a *middleware* for communication, ensuring semantic interoperability between the members of a robot group.

Our ontologies can be used as *reusable knowledge components* in *knowledge-based problem-solving processes*. Using CORA, thus, a robot can apply high-level logical reasoning capabilities, taking advantage of its high-level knowledge about the world to decide which action it should perform in order to achieve its goal. In general, robots can use ontologies to support tasks such as *planning* [15, 16, 17] and *navigation* [18]. Other ontologies can also be integrated with our ontologies, providing a wide range of concepts and relations that allow richer descriptions of the robot's world. Such semantic descriptions can be used by the robot in perception processes such as [19, 20, 21, 22] for enhancing tasks that require *object recognition* through *visual perception*. These semantic descriptions can be used also for specifying tasks to the robot, as in [23].

Furthermore, our ontologies can be used for defining the notions underlying *robot programming frameworks*. CORA could provide these frameworks with a conceptual structure that fits the conceptualization that is shared among the roboticists. For instance, an object-oriented programming framework for robots based on concepts and relations in CORA would be more easily assimilated by new programmers. In this way, dealing with these frameworks would become more natural for the practitioners of R&A. In addition, our ontologies could define standard *interfaces* for these frameworks, promoting the semantic interoperability among them.

CORA can also be used for promoting *data integration* and *semantic interoperability* among robot databases. This could have positive impacts to the *knowledge management* process of companies that commercialize products and components for the R&A field.

10. Future work: what should we expect next?

CORA and related ontologies still do not cover some important areas in R&A. For instance, *control* still needs to be taken into account. This issue is complex, since it involves other important concepts in robotics, such as perception, planning, and action. CORA should also incorporate information ranging from simple classical controllers — such as proportional-integral-derivative controllers (PID) — to complex non-linear control. In addition, it should also account for different control strategies.

The notion of *task* is also important in this domain. Since robots should be able to operate in complex scenarios, task definitions must be clear to allow robots to communicate with each other, other machines, and humans. In this sense, ontologies play a clear role in task specification. CORA must be designed to allow several types of tasks in various environments, e.g., grasp, move, scan, and so on. Future work will be devoted to the ontological characterization of what kind of entity a task is. For example, we believe that a good starting point is to separate *tasks* from *task executions*. With this distinction, we acknowledge that tasks are *abstract* entities that describe goals to be reached; while tasks executions are *events* composed by *actions* that are performed by robots in the world in order to reach a given goal. Moreover, in future steps it is necessary to identity the basic kinds of tasks that robots usually perform. These task definitions will be the basis of more complex task definitions. CORA must define clearly the interfaces to domain ontologies, like industrial [24] or surgical [25] [26].

Furthermore, planning is also an important related issue. Given a task, the *plan* is an abstract partially ordered set of references to actions, which when performed, contribute to the task execution. Possibly, any development in this area should take into account SUMO concepts related to plan.

Finally, CORA and related ontologies do not represent changes in time (e.g. changes in sensor data). We envisage a *diachronic* version of CORA, where time is taken into account.

Acknowledgment

The IEEE-SC WG is supported by the IEEE Robotics and Automation Society. This work is partially supported by FCT, through IDMEC, under LAETA Pest-OE/EME/LA0022. The authors acknowledge the support of Brazilian CNPq, Petrobras PRH PB-17 and the Hungarian Eötvös Scholarship. T.H. is a Bolyai Fellow of the Hungarian Academy of Sciences.

References

[1] Studer, R., Benjamins, V.R., Fensel, D.. Knowledge engineering: Principles and methods. Data and Knowledge Engineering 1998;25(1-2):161–197. URL `http://dx.doi.org/10.1016/S0169-023X(97)00056-6`.

[2] Schlenoff, C., Prestes, E., Madhavan, R., Gonçalves, P.J.S., Li, H., Balakirsky, S., et al. An IEEE standard ontology for robotics and automation. In: Proc. of the 2012 IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems,Vilamoura. 2012, p. 1337–1342.

[3] Guizzardi, G.. Ontological foundations for structural conceptual models. PhD thesis; University of Twente; The Netherlands; 2005.

[4] Fernández, M., Gómez-Pérez, A., Juristo, N.. METHONTOLOGY: from ontological art towards ontological engineering. In: Ontological Engineering; vol. 6 of *AAAI Spring Symposium*. 1997, p. 33–40.

[5] Guarino, N., Welty, C.A.. An overview of OntoClean. In: Staab, S., Studer, R., editors. Handbook on Ontologies. Intl. Handbooks on Information Systems; Springer Berlin Heidelberg. ISBN 978-3-540-70999-2, 978-3-540-92673-3; 2009, p. 201–220. URL `http://link.springer.com/chapter/10.1007/978-3-540-92673-3_9`.

[6] Niles, I., Pease, A.. Towards a standard upper ontology. In: Proceedings of the international conference on Formal Ontology in Information Systems - Volume 2001. FOIS '01; New York, NY, USA: ACM. ISBN 1-58113-377-4; 2001, p. 29. doi:\bibinfo{doi}{10.1145/505168.505170}. URL `http://doi.acm.org/10.1145/505168.505170`.

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

[8] Carbonera, J.L., Fiorini, S.R., Prestes, E., Jorge, V.A., Abel, M., Madhavan, R., et al. Defining positioning in a core ontology for robotics. In: Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on. IEEE; 2013, p. 1867–1872.

[9] Balakirsky, S., Kootbally, Z., Kramer, T., Pietromartire, A., Schlenoff, C., Gupta, S.. Knowledge driven robotics for kitting applications. Robotics and Autonomous Systems 2013;61(11):1205–1214.

[10] torga, M., Andreasen, M.M., Marjanovi, D.. The design ontology: foundation for the design knowledge exchange and management. Journal of Engineering Design 2010;21(4):427–454. doi:\bibinfo{doi}{10.1080/09544820802322557}.

[11] Huang, H.M., Messina, E., Albus, J.. Toward a generic model for autonomy levels for unmanned systems (alfus). Tech. Rep.; DTIC Document; 2003.

[12] Oberle, D., Ankolekar, A., Hitzler, P., Cimiano, P., Sintek, M., Kiesel, M., et al. Dolce ergo sumo: On foundational and domain models in the smartweb integrated ontology (swinto). Web Semantics: Science, Services and Agents on the World Wide Web 2007;5(3):156–174.

[13] Ye, J., Coyle, L., Dobson, S., Nixon, P.. A unified semantics space model. In: Location-and context-awareness. Springer; 2007, p. 103–120.

[14] Morita, S.. Geometry of differential forms. No. v. 201 in Translations of mathematical monographs; Providence, R.I: American Mathematical Society; 2001. ISBN 0821810456.

[15] Provine, R., Schlenoff, C., Balakirsky, S., Smith, S., Uschold, M.. Ontology-based methods for enhancing autonomous vehicle path planning. Robotics and Autonomous Systems 2004;49(1):123–133.

[16] Galindo, C., Fernández-Madrigal, J.A., González, J., Saffiotti, A.. Robot task planning using semantic maps. Robotics and Autonomous Systems 2008;56(11):955–966.

[17] Belouaer, L., Bouzid, M., Mouaddib, A.. Ontology based spatial planning for human-robot interaction. In: Temporal Representation and Reasoning (TIME), 2010 17th International Symposium on. IEEE; 2010, p. 103–110.

[18] Bateman, J., Farrar, S.. Modelling models of robot navigation using formal spatial ontology. In: Spatial Cognition IV. Reasoning, Action, Interaction. Springer; 2005, p. 366–389.

[19] Modayil, J., Kuipers, B.. Autonomous development of a grounded object ontology by a learning robot. In: Proceedings of the national conference on Artificial intelligence; vol. 22. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999; 2007, p. 1095.

[20] Suh, I.H., Lim, G.H., Hwang, W., Suh, H., Choi, J.H., Park, Y.T.. Ontology-based multi-layered robot knowledge framework (omrkf) for robot intelligence. In: Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on. IEEE; 2007, p. 429–436.

[21] Johnston, B., Yang, F., Mendoza, R., Chen, X., Williams, M.A.. Ontology based object categorization for robots. In: Practical Aspects of Knowledge Management. Springer; 2008, p. 219–231.

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

[23] Stenmark, M., Malec, J.. Knowledge-based industrial robotics. In: SCAI. 2013, p. 265–274.

[24] Balakirsky, S., Kootbally, Z., Kramer, T.R., Pietromartire, A., Schlenoff, C., Gupta, S.. Knowledge driven robotics for kitting applications. Robotics and Autonomous Systems 2013;61(11):1205–1214.

[25] Gonçalves, P.. Towards an ontology for orthopaedic surgery, application to hip resurfacing. In: Proceedings of the Hamlyn Symposium on Medical Robotics. London, UK. ISBN 978-0-9563776-4-7; 2013, p. 61–62.

[26] Haidegger, T., Barreto, M., Gonçalves, P., Habib, M.K., Ragavan, V., Li, H., et al. Applied ontologies and standards for service robots. Robotics and Autonomous Systems 2013;61(11):1215–1223. doi:\bibinfo{doi}{10.1016/j.robot.2013.05.008}. URL `http://dx.doi.org/10.1016/j.robot.2013.05.008`.