ENVOn: An ontology of 3D environment where a simulated manipulation task takes place

Yingshen ZHAO[a], Arkopaul SARKAR[a], Linda ELMHADHBI[b], Mohamed Hedi KARRAY[a], Philippe FILLATREAU[a], Bernard ARCHIMEDE[a]
[a]*INP-ENIT, Université Fédérale de Toulouse Midi-Pyrénées, Tarbes, France*
[b]*INSA-Lyon, University of Lyon, Lyon, France*

Abstract. Thanks to the advent of robotics in shopfloor and warehouse environments, control rooms need to seamlessly exchange information regarding the dynamically changing 3D environment to facilitate tasks and path planning for the robots. Adding to the complexity, this type of environment is heterogeneous as it includes both free space and various types of rigid bodies (equipment, materials, humans etc.). At the same time, 3D environment-related information is also required by the virtual applications (e.g. VR techniques) for the behavioural study of CAD-based product models or simulation of CNC operations. In past research, information models for such heterogeneous 3D environments are often built without ensuring connection among different levels of abstractions required for different applications. To address such multiple points of view and modelling requirements for 3D objects and environments, this paper proposes an ontology model that integrates the contextual, topologic, and geometric information of both the rigid bodies and the free space. The ontology provides an evolvable knowledge model that can support simulated task-related information in general. This ontology aims to greatly improve interoperability as a path planning system (e.g., robot) and will be able to deal with different applications by simply updating the contextual semantics related to some targeted application while keeping the geometric and topological models intact by leveraging the semantic link among the models.

Keywords: Ontology, 3D environment, rigid bodies, free space, simulation

1. Introduction

Modern manufacturing and supply-chain industries are increasingly using robots to automate material handling and machining in the shopfloor and warehouse environment. The key part of the information related to any 3D environment, where a simulated manipulation task is carried out by CNC machines, robots and automated vehicles [1], may include both rigid bodies and free space. Different levels of abstractions, such as context, topology, and geometry are used in modelling the 3D objects depending on whether they represent designed artefacts (e.g. CAD models) or obstacles in a simulated landscape. For the latter, the relation of the 3D objects with the space in which they are located is also important. Distributed manufacturing and supply-chain operations require a common set of vocabulary for such a 3D environment to exchange information regarding tasks and path planning for the robots and equipment in this heterogeneous environment between control rooms, responsible for planning, and the control at the shopfloor or the warehouse. Ameri et al. discussed why interoperability of the exchanged information is the key to the digital transformation of modern manufacturing and supply chain, additionally showing why ontology fares better than traditional manufacturing data standards, and finally suggesting pervasive adoption of integrated ontologies based on a common foundation [2].

The interest of modelling and using the environment information from heterogeneous viewpoints have been found in some previous work [3]–[5] related to path planning and have been used in the practical robotics domain [6] [7] [5]. However, these heterogeneous models are often built without ensuring connection among different levels of abstractions. For example, geometrical models are not linked to spatial and topological information, and the geometrical objects do not express how these objects are viewed in different contexts. This type of disconnected modeling approach does not allow fast queries, such as “what is the central axis of *Object A* or *Hole B*?”, which mixes both an geometrical entity (e.g., central axis) and context dependent entity (hole). Moreover, without adhering to formal logic, these models do not provide inference support to determine queries like “whether *Object A* fits *Hole B?*” as such query needs a model of ‘fitness’ to be successful.

The traditional product development process in PLM (Product lifecycle management), which heavily rely on 3D modelling for designing products, is driven by the customers' needs to meet their expectations with some targeted functionalities. These basic functions are then defined, developed, and validated individually before being integrated into the final expected product. It allows the company to adopt a commonly-used and rigorous development lifecycle model during product development, called the V-cycle model [8]. Therefore, the complete product model from PLM's perspective is not only about the structural information of the physical product and its components but also their kinematic and static behaviours in relation to the environment, other objects that the product will interact with, and the functional requirements that the product aims to satisfy.

Answering these requirements, 3D Computer-Aided Design (CAD) modelling, starting from the mid of the 1960s, uses the power of computers to support the creation, modification, analysis, and optimization of a product using virtual prototypes [9]. Functional and integration tests performed on virtual prototypes require both accurate geometrical information and a higher-level functional description of the product and its parts. Lacking this level of completeness in the product model, the physical prototypes are often interleaved with the virtual prototypes in the V-cycle product development process [10] with the latter being used only at the conceptualization phase and the earlier for rigorous quality assurance.

Virtual prototypes are widely used for simulation purposes, too. Different simulation software (e.g. DMU kinematics for CATIA®[1], Motion modules for Solidworks®[2]) carry out the kinematic analysis of a system and verify whether the system can function correctly. Recently, thanks to the progress made by sensorimotor interfaces and their coupling with 3D content, the emergence of VR techniques allows immersive and interactive task simulations while considering the human operator in the loop. Virtual Reality techniques allow the human operator in the loop to be in immersion and interaction in the virtual environment or product virtual prototype, and to take advantage of the cognitive capacities of the operator during the simulations performed. Furthermore, up to date products are more and more integrated, and industrial companies express the need to validate the tasks associated with their lifecycle from the design stage on, as those tasks may be performed under very strong geometric constraints. Showing the feasibility of motion is then a key issue. Automatic motion planning techniques, developed by the robotics community from the 1980s on, may help but show serious limitations when using purely geometric models of the product or the environment. Considering higher abstraction level information than the purely geometric models traditionally used should be considered [11].

Furthermore, complex task and path planning required for industrial robotics and automated vehicles need a symbolic level representation of the environment and location of various objects situated in it. In these types of problems, the topological representation of the environments and objects is more important than intricate geometrical information.

To address such multiple points of view and modelling requirements for 3D objects and environments, we propose an ontology model that integrates the contextual, topologic, and geometric information of both the rigid bodies and the free space. This environment ontology is inspired by the multi-level environment model, proposed by Cailhol et al. [11]. The aim is to construct an evolvable knowledge model of the environment, composed of both rigid models and free space, that can support simulated task-related information. Such kind of integrated knowledge model will greatly improve interoperability as a path planning system (e.g., robot) will be able to deal with different applications by simply updating the contextual semantics related to some targeted application while keeping the geometric and topological models intact, thanks to the semantic link among the models. Therefore, the originality of the proposed ontology lies in the fact that it conceptualizes heterogeneous knowledge about both the obstacles (rigid bodies) and the free space.

The rest of the paper is organized as follows. Section 2 overview the existing research on the related topics. Section 3 gives a detailed description of the proposed 3D environment ontology. We discuss why different abstraction levels of environment information should be considered (i.e. context, topology, and geometry) in an integrated framework. We apply a modular architecture in the proposed ontology. We will describe how a concept might have different meanings at different abstraction levels. Section 4 presents the ontology validation results. Two simulation scenarios are presented. The evaluation of each of the two scenarios is presented and discussed.

2. State of the art

In the following, we explore past studies in data models that were developed for both robotic

[1] http://www.3ds.com/fr/produits-et-services/catia/fonctionnalites/ingenierie-mecanique/

[2] http://www.solidworks.fr/sw/products/3d-cad/cad-animation.htm

applications focusing on an environment composed of both free space and rigid bodies, and CAD applications focusing on geometric information of 3D design of solid bodies.

The semantic information of an environment is made of the types of objects, their locations in the space, and their semantic identity. This kind of information model, which is formally called 'semantic environment map' [12], has already been discussed in knowledge representations for path planning in robotics, such as navigation maps and collision maps, especially for the manufacturing environment [13]. Rusu presented a semantic 3D object map to annotate environment objects and their surfaces with semantic labels [14]. It consists of a 3D point cloud perceived from robot perception (e.g. vision[15], touch[16]), polygonal models of objects constructed from clustering and segmentation of the point cloud, and a semantic interpretation of objects and their surfaces. It serves as semantic resources to determine the final grasp or placement position for a manipulation. For example, in an indoor kitchen environment, it allows a robot to locate the hinge of a drawer when a robot is given a high-level command to open the drawer. Certain works of Marton [17] and Blodow [18] only concern modelling environment from a specific context, e.g., kitchen.

Among the models that use richer semantics, KnowRob [4], a Prolog-based knowledge processing system capable of accessing OWL ontologies, captures the encyclopedic knowledge to describes the types and the properties of objects (e.g. refrigerator, drawer, micro-oven in a kitchen) as well as the commonsense knowledge to describe the everyday usages for these objects.

In an ontology-based multi-layered robot-knowledge framework, called OMRKF[19], the environment knowledge model for the robot has three levels: 1) object features level describes the visual attributes, e.g., colours, textures, and features, that are used to recognize an object, 2) object identity level that forms the taxonomy of these objects, and 3) space level that describes the taxonomy of locations, e.g. living room and bedroom. In both KnowRob and OMKRF, the environmental model is application-specific and only concerns information related to the operations of the robot. However, they do propose basic connectivity among the locations of free space models. For example, KnowRob model defines the concept 'Place' to designate relevant locations in an environment and the concept 'Map' as an abstract symbol for topologically mapping those places in an environment. OMKRF model describes the taxonomy of different locations of an environment. A topological map is used to describe the connectivity among the locations. However, in both cases, the structure of the topological map is not defined. They are also incomplete as only relevant locations are identified as places.

The most important facet of knowledge modelling for the robotic environment is to utilize the geometric constraints to determine accessibility and localization with fast queries. For example, to localize the top face of a table so that a robot can put down a bottle on it or understand the constraints of holding a cup upwards. The environment where a simulated manipulation task takes place is mostly considered as a closed part of the 3D Cartesian space cluttered with mobile/fixed obstacles (regarded as a rigid body). These rigid bodies are built on CAD models. Recently, the environment model proposed by Cailhol et al. [11] consists of a rigid bodies model and a free-space model. Both of them involve different levels based on semantic, topologic, and geometric information. Regarding formal schema for capturing the geometry of rigid bodies, STEP (STandard for Exchange of Product model data) [20] is developed by the ISO organization (referenced as ISO 10303) to meet the needs of modern industry to facilitate the exchange of product data (including the CAD models) among different phases of product's development or different organizations. Among various schema developed under the aegis of STEP, AP203 (Configuration-controlled 3D design of mechanical parts and assemblies) is the most widely used application protocol [21] that closely follows Boundary Representation (BREP) for 3D models. OntoSTEP is an effort by Barbau et al. to translate the STEP schema directly into an ontology model formalized in OWL [22]. An implementation of a particular product thus can be instantiated in the defined ontology model. However, the automated extracted taxonomy of AP203 is meaningless as STEP standards often employ concepts that do not have any semantic relevance but are only used to better organize the data.

Perzylo et al. construct an ontology model defining boundary representations (BREP) of objects from scratch without referring to STEP [23]. This ontology consists of a topological part, illustrating the topological connectivity and orientations of vertices, edges, and faces of a geometric part, describing the geometric primitives relating to the topological part (i.e., points, curves, surfaces). Yet, it is still limited to the boundary representation of CAD objects, whereas other models (such as Constructive Solid Geometry, abbv. CSG) are also possible to illustrate the geometries of CAD objects. Ontology for 3D shapes can also be found in the work of Sarkar and Sormaz [24]who used foundational concepts and relationships described by the top-level ontology BFO (Basic Formal Ontology) in their model to ensure the interoperability of the 3D design information. This ontology also makes a distinction among the geometric entities and various representational schemes by which they are encoded, e.g. various types of polynomial equations for representing a curve in space, and BREP or CSG techniques for representing complex solids.

The difference in the viewpoints in modelling the 3D objects is also apparent in the models of free space. Two main techniques have been used to synthesize the geometries of a free space model: cell decomposition and roadmap model. According to how the cells are formed, the related works can be classified in mainly four categories: Exact cell [25], Rectangular Cells [26], Regular Cells [27], Unbalanced Tree (quadtree in 2D space [27] and octree in 3D space [27]). Roadmap models the points of interest and interconnects them as a graph to describe the connectivity of the free space. This technique reduces the amount of information required by cell decomposition of the entire space as only relevant portions of space are included.

From the topological viewpoint, free space needs to be synthesized to represent connectivity, such as the reachability between different locations. In the studies on robotic applications, the arc connecting two views is represented as the state transition of the environment as the robot's sensory perception changes. Kortenkamp et al., Dedeoglu et al., and Kuipers et al. [28]–[30] propose the construction of a topological map using distinct views as nodes and their transitions as arcs. Similar to the efforts of focusing only on the interesting part of the views, Hirtle et al. and McNamara [31], [32] define 'region' as a unique location in space. Mozos et al. [33] propose to construct spatial regions by detecting doorways. Cailhol et al. [11] define a place as a topological graph connecting places borders built on octree decomposition.

Similar to rigid bodies, the semantic information of the free space model varies among applications and it must be adapted to the tasks proceeded by applications. For example, for indoor robotic applications, both common-sense and encyclopedic knowledge [4] can be used to annotate different household locations, such as kitchen, corridor, bedroom. 'Semantic map' in the literature [34]–[36] captures the human's point-of-view of the environment where tasks are performed. It associates the semantic information (the taxonomy of locations, like room, corridor, and their properties) with the places constructed at the topologic level and also their geometric description at the geometric level. Although these works substantiate the efficacy of admitting multiple viewpoints in the model, they do not adopt a rigorous knowledge modelling framework to link these viewpoints.

Finally, despite their acknowledgement of the need of taking different aspects of the environment into account, none of these aforementioned ontologies has considered modelling the environment information from different viewpoints (i.e., context, topology, geometry) for both the rigid bodies and the free space models together in an evolvable ontology. Such an integrated semantic model will also allow fast queries to be executed in any environment information and possibly infer new knowledge. More importantly, rather than manually assigning the tedious geometric constraints to a primitive action, they can be automatically inferred based on the information related to a primitive action to be performed (e.g., the final location of a manipulated object) and task-related geometric constraints. Therefore, the integrated knowledge model should carefully identify and distinguish the information related to a domain from those related to an application.

Furthermore, the existing environment models focus on individual layers (geometric, topological, context) separately. This necessitates additional mapping functions (often hard-coded) to be developed in the path planning systems. These functions must be managed by applications and pose difficulty in times of upgrading the applications. A connected model will let users encode the knowledge for querying and reasoning based on the common ontology model without needing to develop additional mapping functions using code (e.g., Java). The inference rules realize the functions can be easily changed and adapted to the targeted applications. This loosely coupled architecture between the knowledge model and the path planning system will then enhance the reusability of the planning algorithm (task or path) in different applications or tasks. Finally, such an ontology will open the possibility to study the use of the task-related information (e.g., finding the goal to reach, inferring the geometric constraints to be obeyed) in the path planning of a given primitive action of a task plan.

3. Specifications of the 3D Environment (ENVOn)

ENVOn aims to capture the core notions and relations related to a 3D environment where a simulated manipulated task takes place, i.e., context, topology, and geometry of both the rigid bodies and the free space models. The proposed ontology reuses the concepts already defined in the multi-level environment model proposed in [11] and other existing standards and ontologies related to the modelling of a manipulation environment, such as the geometries of CAD models defined in the STEP standard [21].

To extract the requirements for developing the model for the robotic environment by considering geometric, topological, and contextual aspects jointly, an example use case is formulated below.

Let us consider an environment, composed of a *Cylindrical* object (*cylinder_obj*) with a radius (*radius_obj*), a Panel (*panel*), two *Cylindrical* holes (*hole1*, *hole2*) with different radius (i.e., *radius_hole1, radius_hole2*) and a *TriangularPrism* hole (*hole3*) on the Panel (*radius_hole1 > radius_obj; radius_hole2 < radius_obj*).

To correctly process a primitive action of "*Insert (cylinder_obj, panel)*" in the above environment, the path planning system must be able to answer the following questions:

- To which place of the topologic level belongs the holes?
- What are the shapes of these holes? i.e., *hole1* and *hole2* have the shape of *cylinder_obj* that matches with the shape of the inserting object but not hole 3 which has the shape of *TriangularPrism*. Such information belongs to the semantics associated with objects and holes.
- What are the dimensions of the holes? This information is required to check if the inserting object has a smaller diameter than the hole as it cannot be inserted, otherwise. The geometric information is mandatory in solving such issues.

In the following table, a set of competency questions (CQs) [37] are provided with that address the requirements from multiple points of view (i.e., context, topology, and geometry).

Table 1. Competency Questions

Querying geometric details	CQ1	What is the central axis of (X) or (Y)?
	CQ2	What is the opening direction of (X) or (Y)?
	CQ3	What is the pointing direction of (X) or (Y)?
	CQ4	What is the volume of (X) or (Y)?
	CQ5	What is the sweeping plane of (X) or (Y)'s volume?
	CQ6	What is the symmetric vector of the sweeping plane of (X) or (Y)'s volume?
	CQ7	What is the sweeping direction of (X) or (Y)'s volume?
	CQ8	What is the central axis of (X) or (Y)?
	CQ9	What is the origin of (X) or (Y)?
	……	……
Localization	CQ10	Where is (X) in the 3D environment?
	CQ11	Which places (Y) are inside a rigid body(X), such as *Panel*?
	……	……
Navigation	CQ12	Does (Y) is the right hole to insert (X)?
	CQ13	Does (X) fit the hole (Y)?
	CQ14	Which places (Y) are the least complex to across?
	……	……

(X: Rigid Body, Y: Place)

4. The general architecture of the ontology model

Figure 1 shows the general architecture of the proposed ontology with some key concepts and relations. A key objective of building ENVOn is to have a common vocabulary to be reused by different applications concerning manipulation tasks. The knowledge at each level should be easily extracted, updated, and reused by other domain ontologies. Therefore, we consider a modular architecture of the ontology model, where each level represents a module. In our design, three different modules are proposed:

- The geometry description module groups the concepts and relations related to the geometries of the rigid bodies and the free space module. *RB Geo Model* consists of two possible geometric models (i.e., CSG and BREP) of rigid bodies (*Rigid Body*) based on CAD. *3D Space Geo Model* concerns the cell decomposition (Cell decomposition 3D) of the free space model (3D free space). *Area* represents a bounded volume of the *3D free space*.
- The topology description module describes places (*Place*) and borders (*Border*) identified in the 3D simulation environment. It also illustrates their connectivity by constructing a topological graph (*TopoGraph*).
- The context description module provides the semantic description of rigid bodies (*Rigid body*), places, and borders. Such a kind of description includes the potential taxonomy (e.g., *Container, Opening, Hole*) and the related properties (*Function, Color, Shape*). We must note that this level of information heavily relies on the application (i.e., manipulation task to be performed).

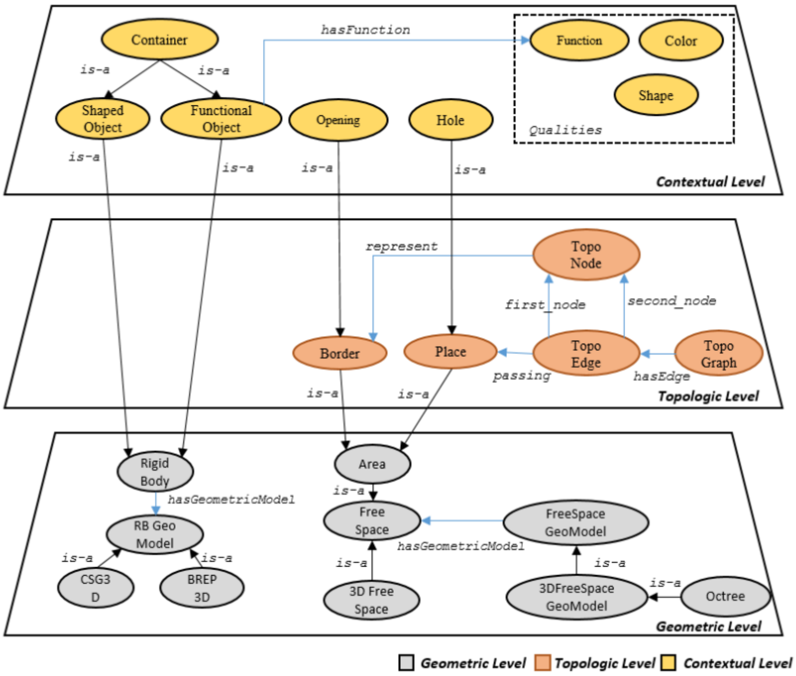


Figure 1. The general architecture of ENVOn

Although the modular structure of the proposed ontology can facilitate its reusability by other domain ontologies, the modules rely so heavily on one another that the reused concepts and relations should be carefully considered by the ontology engineers while importing the ENVOn modules in their works. Please note that Figure 1 is only an example of the proposed ontology (previously introduced in [52]) and more detailed concepts will be given in the following sections.

4.1 The geometric description module

4.1.1 The common concepts and relations of the geometric description module

First, we defined some mathematical concepts that are the foundation of geometric information, e.g., the coordinate of *Point*, the direction of *Axis*, the local reference frame of *Rigid bodies*. In Figure 2, the conceptual map illustrates the main defined concepts.

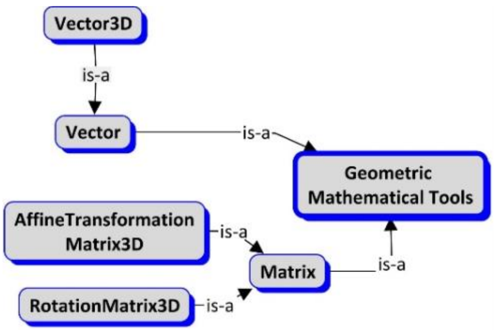


Figure 2. The basic mathematical concepts for the geometric description

The *Vector* concept is formally defined as R_n, which is specified as $(a_1, a_2, \ldots, a_n)$ – each element denoting one coordinate of an n-dimensional vector. An n-dimensional vector is often called an n-vector. The *Matrix* concept was introduced in 1851 by [38] to represent an array of determinants of a system, with m lines and n columns. In linear algebra, it is a very useful tool to represent various transformations e.g., translation, rotation, and scaling. In the geometric description of a 3D environment, the *Vector* and *Matrix* concepts are further classified, e.g., the coordinates of Points and transformation between two reference frames can be described. We will first provide some descriptions before providing formal definitions in Table 1.

- *Vector3D* is defined in the 3-dimensional Cartesian space, given by the x, y, and z coordinates and specified as three real numbers.

- *RotationMatrix3D* represents a rotation between two frames of reference in 3-dimensional Cartesian space (R_3), any rotation can be given by a composition of rotations of the “x, y, z” axis, given in the form as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a11 & a12 & a13 \\ a21 & a22 & a23 \\ a31 & a32 & a33 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ z' \end{bmatrix}$$

- *AffineTransformationMatrix3D* is a combination of rotation, translation, and scaling. It preserves the collinearity (i.e., points on a line remain collinear after transformation) and the proportions on the lines (i.e., the midpoint of the line remains the midpoint after transformation).

Table 1. Axioms – Basic mathematical concepts

Concept	Axiom
Vector3D	**EquivalentTo:** Vector and (x exactly 1 xsd:double) and (y exactly 1 xsd:double) and (z exactly 1 xsd:double)
RotationMatrix3D	**EquivalentTo:** Matrix and (a11 exactly 1 xsd:double) and (a12 exactly 1 xsd:double) and (a13 exactly 1 xsd:double) and (a21 exactly 1 xsd:double) and (a22 exactly 1 xsd:double) and (a23 exactly 1 xsd:double) and (a31 exactly 1 xsd:double) and (a32 exactly 1 xsd:double) and (a33 exactly 1 xsd:double)
AffineTransformation Matrix3D	**EquivalentTo:** Matrix and (a11 exactly 1 xsd:double) and (a12 exactly 1 xsd:double) and (a13 exactly 1 xsd:double) and (a14 exactly 1 xsd:double) and (a21 exactly 1 xsd:double) and (a22 exactly 1 xsd:double) and (a23 exactly 1 xsd:double) and (a24 exactly 1 xsd:double) and (a31 exactly 1 xsd:double) and (a32 exactly 1 xsd:double) and (a33 exactly 1 xsd:double) and (a34 exactly 1 xsd:double) and (a41 exactly 1 xsd:double) and (a42 exactly 1 xsd:double) and (a43 exactly 1 xsd:double) and (a44 exactly 1 xsd:double)

Second, we defined the geometric primitives that are considered as a set of elementary geometric objects [39], the combination of which may be used to represent varieties of complex 3D shapes (e.g., CAD parts). The common set of geometric primitives irrespective of any geometrical modelling techniques includes point, curve, and surface [40]. To further separate the geometric data from geometric modelling techniques as proposed by Kaiser et al. [41], *volume* should also be considered as a geometric primitive. *Volume* can not only be used to describe the primitive shapes in CSG (Constructive Solid Geometry) but also can be used to describe the geometry of any closed part of 3D Cartesian space. Figure 3 represents the geometric primitives using a conceptual map. These primitives are formally defined as follows:

- *Point3D* represents a position in a 3D Cartesian coordinate space. A *Vector3D* describes its position.
- *Curve3D* represents a path of a *Point3D* moving through a 3D Cartesian coordinate space.
- *Surface3D* represents a 2D subspace of a 3D Cartesian space.
- *Volume3D* represents the bounded volume by surface patches. A *Volume3D* might be formed by sweeping a certain *Surface3D* following a certain curve.

Curve3D, *Surface3D*, and *Volume3D* are abstract concepts. They can be further decorated with free parameters to define some pre-defined shapes. However, these concrete shapes need an *AxisPlacement3D* so that we can specify their orientation and location in 3D Cartesian space. Table 3 presents the formal definition of *AxisPlacement3D*.

CircularCurve3D and *CircularPlane3D* are subtypes of *Curve3D* and Surface3D respectively and both are located some *AxisPlacement3D*. *CircularCurve3D* has a radius, whereas *CircularPlane3D* has a radius and is bounded by a *CircularCurve3D* curve. A *CylindricalVolume* is a *Volume3D* that has a sweeping plane, which is a *CircularPlane3D*, a sweeping direction (*Vector3D*), and a sweeping length. The formal definitions of these concepts are given in Table 2. It is to be noted that the geometrical primitives can be used for modelling both the rigid bodies model and the free space model.

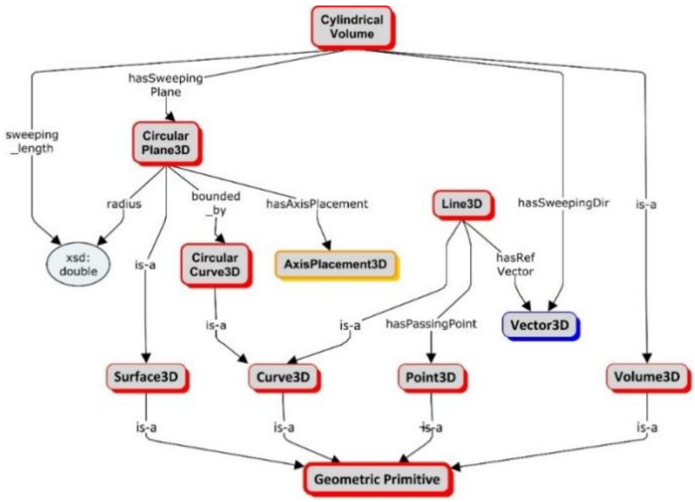


Figure 3. The representation of geometric primitives

Table 2. Geometric Primitives

Concept	Axiom
Point3D	**EquivalentTo:** `GeometricPrimitive and (hasPosition exactly 1 Vector3D)`
Curve3D *Surface3D* *Volume3D*	**SubClassOf** `GeometricPrimitive`
Line3D	**EquivalentTo:** `Curve3D and (hasRefVector exactly 1 Vector3D) and (hasPassingPoint exactly 1 Point3D)`
CircularCurve3D	**EquivalentTo:** `Curve3D and (hasAxisPlacement3D exactly 1 AxisPlacement3D) and (radius exactly 1 xsd:double)`
CircularPlane3D	**EquivalentTo:** `BoundedPlane3D and (hasAxisPlacement3D exactly 1 AxisPlacement3D) and (radius exactly 1 xsd:double)and (bounded_by exactly 1 CircularCurve3D)`
CylindricalVolume	**EquivalentTo:** `Volume3D and (hasSweepingPlane exactly 1 CircularPlane3D) and (hasSweepingDir exactly 1 Vector)and (sweeping_length exactly 1 xsd:double)`

Besides the geometric information about their composition, the rigid body model and the free space model also possibly contain other geometric properties, such as the central axis and the oriented bounding box. The supplementary geometric primitives present a list of geometric elements that do not compose the geometry of the rigid bodies model or the free space but assist in describing and manipulating them. Figure 4 presents four different concepts in a conceptual map, where the formal definitions are provided in Table 3. It is to be noted that *AxisPlacement3D* is also classified as a Supplementary Geometric Primitive as defined below.

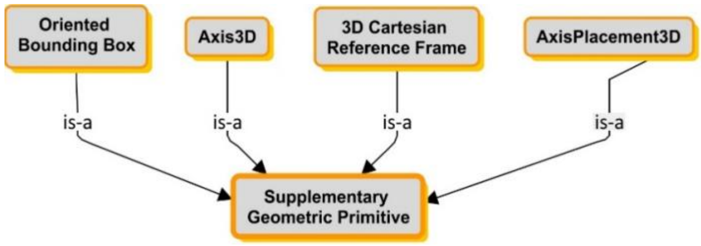


Figure 4. The representation of supplementary geometric primitives

Table 3. Some examples of axioms – Supplementary geometric primitives

Concept	Axiom
Axis3D	***EquivalentTo***: `SupplementaryGeometricPrimitive and (hasPassingPoint exactly 1 Point3D) and (hasRefVector exactly 1 Vector3D)`
AxisPlacement3D	***EquivalentTo***: `SupplementaryGeometricPrimitive and (hasOrigin exactly 1 Point3D) and (hasRefXVector exactly 1 Vector3D) and (hasRefYVector exactly 1 Vector3D) and (hasRefZVector exactly 1 Vector3D)`
3D Cartesian ReferenceFrame	***EquivalentTo***: `SupplementaryGeometricPrimitive and (hasAffineMatrix exactly 1 AffineTransformationMatrix3D or hasAxisPlacement3D exactly 1 AxisPlacement3D)`
Oriented BoundingBox	***EquivalentTo***: `SupplementaryGeometricPrimitive and (hasLocalReferenceFrame exactly 1 3DCartesianReferenceFrame) and (hasMinPoint exactly 1 AxisPlacement3D) and (hasMaxPoint exactly 1 AxisPlacement3D)`

- *3DCartesianReferenceFrame* is a framework to perform measurements on location, distance, angle, etc, precisely and mathematically in a 3D Cartesian space. It is specified either by an *AxisPlacement3D* (an origin point, three orthogonal x, y, z axes), or an affine transformation regarding a world reference frame in 3D Cartesian space.
- *Axis3D* is a *Line3D* to which a point, a curve, a surface, or a rigid body is measured, rotated, etc. For example, a symmetry axis of a surface indicates that each side of the axis is a mirror image.
- *AxisPlacement3D* identifies a reference frame in the3D Cartesian space with a location point (the origin) and three orthogonal axes (i.e., x, y, and z-axis).
- *OrientedBoundingBox* is the minimum enclosing box for a point set of points (such as all points of a rigid body). It is defined by a minimum and a maximum point in the local reference frame.

4.1.2 The geometric representation of the rigid bodies model

In ENVOn ontology, the geometric descriptions of rigid bodies are built by closely following CAD models. Rather than semantically meaningless polygonal meshes, we adopt two main representations: the surface representation, and the volume representation. As discussed in Section 2, Boundary Representation (BREP) and Constructive Solid Geometry (CSG) are two formal schemes for surface and volume representation, respectively.

Concerning the geometric models of rigid bodies, we consider only simple geometries in the ENVOn ontology. For example, only simple kinds of *Surface3D* are used, whereas the NURBS (Non-uniform rational B-spline) surfaces [42] have not been considered at the current state of development. Moreover, besides the geometric models of rigid bodies, we introduce some common geometric properties related to rigid bodies, such as the *central axis*, the *oriented bounding box*, the *origin*. Figure 5 shows the main concepts and relations involved in the geometric representation of the rigid bodies model. The formal definitions of the concepts are given in Table 4.

- *RigidBodyGeometricModel* describes how a rigid body is geometrically composed. Two main modelling techniques: CSG and BREP, are adopted.
    - *BREP3D* concentrates on the boundary description of a rigid body (*bounded_by* attribute). *Solid_Boundary* describes the whole boundary of a rigid body. *BREP_Face*, topologically, represents an oriented 2D-manifold in 3D Cartesian space on the *Solid_Boundary* of a rigid body, and it is geometrically described by a *Surface3D*.
    - *CSG3D* describes the volume representation of a rigid body. The *CSG3D* of a rigid body is constructed using a set of standard primitives (*CSG_Primitive*) and Boolean operations among them. A *CSG3D* representation contains the top-level root *CSG_composite* (root_composite attribute).
- *RigidBody* represents any fixed or mobile obstacle in a 3D Cartesian space, with no deformation allowed. It contains one or more *RigidBodyGeoemtric* models to describe its geometric composition. A *RigidBody* has a *Point3D* as its origin to identify its position in the world reference frame, and a *3DCartesianReferenceFrame* describes the local reference frame of the *RigidBody*; some geometric properties of the *RigidBody* are described in the local reference frame, such as the central axis (Axis3D), the oriented bounding box (*OrientedBouningBox*), the geometric models (*RigidBodyGeometricModel*) and the standard form. We consider the standard form of a *RigidBody* as the volume (*Volume3D*) bounded by the *Solid_Boundary* of the *RigidBody*.

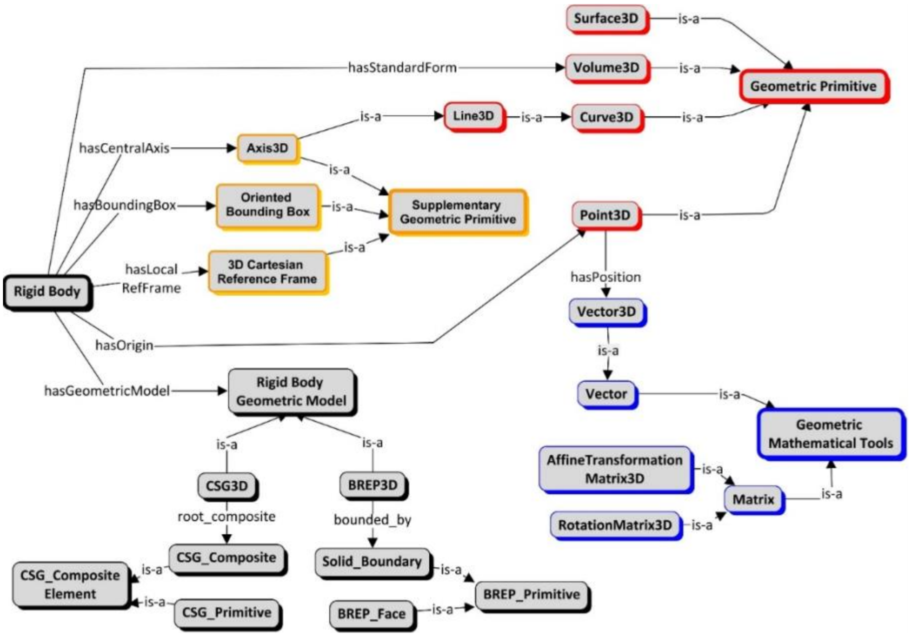


Figure 5. The geometric representation for the rigid bodies model

Table 4. Some examples of axioms – The rigid bodies model

Concept	Axiom
CSG3D	**EquivalentTo:** RigidBodyGeometricModel and (root_composite exactly 1 CSG_Composite)
BREP3D	**EquivalentTo:** RigidBodyGeometricModel and (bounded_by exactly 1 Solid_Boundary)
Rigid Body	**SubClassOf:** (hasStandardForm exactly 1 Volume3D) and (hasCentralAxis exactly 1 Axis3D) and (hasBoundingBox exactly 1 OrientedBoundingBox) and (hasLocalReferanceFrame exactly 1 3DCartesianReferenceFrame) and (hasGeometricModel exactly 1 RigidBodyGeometricModel) and (hasOrigin exactly 1 Point3D)

4.1.3 The geometric representation of the free space model

This work focuses on decomposing the free space model into a set of smaller geometric cells using cell decomposition techniques. Such a representation allows to characterize the geometric volume of the free space model, or even more, to easily find the part of the free space model (volume) in which a simulated task is interested, such as the part that belongs to a hole where a screw should be inserted. Like the rigid bodies model, the geometric representation of the free space model also contains other geometric properties besides its geometric model. Figure 6 shows the main concepts and relations involved in the geometric representation of the free space model. The formal definitions are given in Table 5.

- *3DFreeSpaceGeometricModel* describes the geometric model of free space. We used a classical cell decomposition technique (*Octree*) [11].
- *CellDecomposition3D* is a method that decomposes a closed part of free space into several smaller geometric cells.
- *Octree* is a well-known volumetric representation in which 3D space is recursively divided into eight (hence "oct") smaller volumes by planes parallel to the XY, YZ, and XZ coordinate system planes [43]. Since *Octree* only divides those geometric cells overlapped by obstacles, it is an unbalanced tree.
- *OctreeNode*: Each cell in the *Octree* is called an *OctreeNode*. Geometrically, it has a cuboid volume.
- *3DFreeSpace* represents the obstacle-free part of the 3D Cartesian space. Similar to the *RigidBody* geometric representation, a *3DFreeSpace* might have an origin (*Point3D*) to describe its position in the world reference frame, and a local reference frame (*3DCartesianReferenceFrame*) at this origin so that some geometric properties can be locally described, such as an oriented bounding box, a central axis, and a standard geometric form.

- *Area* represents a continuous closed part of *3DFreeSpace* with a collection of common properties. Semantically, it can be further classified, such as kitchen, corridor.

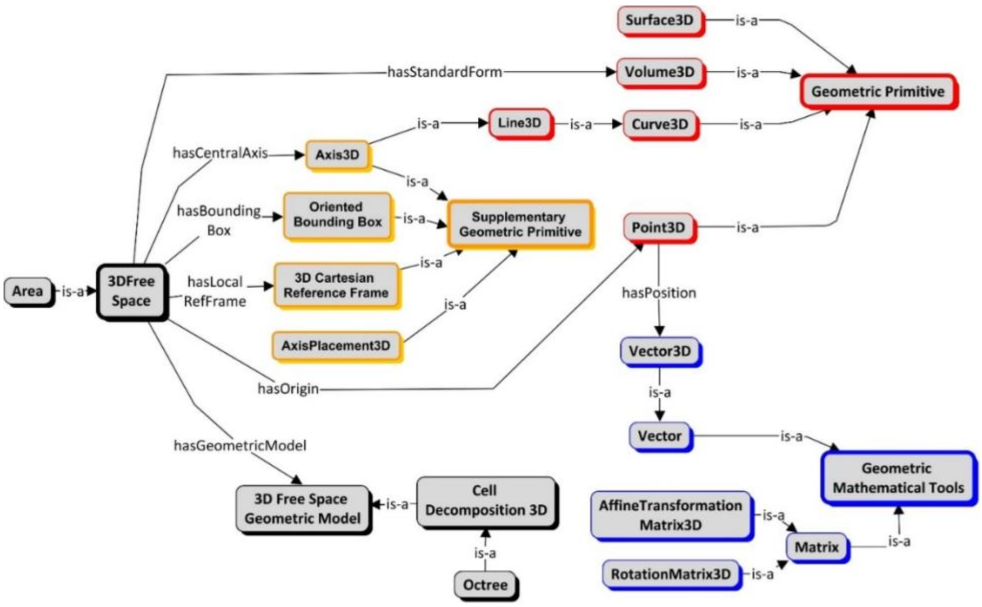


Figure 6. The geometric representation for the Free Space model

Table 5. Some examples of axioms – The free space model

Concept	Axiom
Octree	**EquivalentTo:** CellDecomposition3D and (hasRoot exactly 1 OctreeNode)
3DFreeSpace	**SubClassOf:** (hasStandardForm exactly 1 Volume3D) and (hasCentralAxis exactly 1 Axis3D) and (hasBoundingBox exactly 1 OrientedBoundingBox) and (hasLocalReferanceFrame exactly 1 3DCartesianReferenceFrame) and (hasGeometricModel exactly 1 RigidBodyGeometricModel) and (hasOrigin exactly 1 Point3D)
Area	**EquivalentTo:** 3DFreeSpace and (isCloseBounded exactly 1 true)

4.2 The topological description module

ENVOn needs to be able to answer competency questions related to localization and navigation, such as CQ11 to CQ14. For example, "Is a *Place* Y the right hole to insert a *RigidBody* X?", "Can a *RigidBody* X reach a *Place* Y from its current location?". To answer such questions, the term *Place*, representing different locations of interest in an environment, needs to be tackled first. In the work of Cailhol et al. [11], the topological layer represents places, borders, and the topological relations between them (the border connects two places). Each place or border is associated with a set of geometrical cells in the geometrical layer. The topological model of the environment is static. The adjacency between the identified *Borders* allows the construction of a *TopologicalGraph* that can describe all possible connections to the *Places*. The topological description of the rigid bodies model (i.e., the connectivity of surfaces of a rigid body) is currently out of our scope and this study only concerns free space. Figure 7 shows the main concepts and relations of the topological description module. The formal definitions are given in Table 6.

- *Place* and *Border*: A *Place* represents a location in the environment. It is further classified regarding some concrete properties. For example, bedroom, kitchen, and bathroom are different *Places* specified by their functionality (i.e., sleeping, cooking, bathing respectively). A *Border* is the overlapped *Area* between two *Places*, such as the entrance between a room and a corridor.
- *TopologicalNode* is a basic element of the *Topological Graph* to represent a *Border*.
- *TopologicalEdge* connects two *TopologicalNodes* passing through a certain *Place*.
- *TopologicalGraph* describes the general connectivity of the free space model. It contains several *TopologicalEdges* to describe connections between different *Borders*.

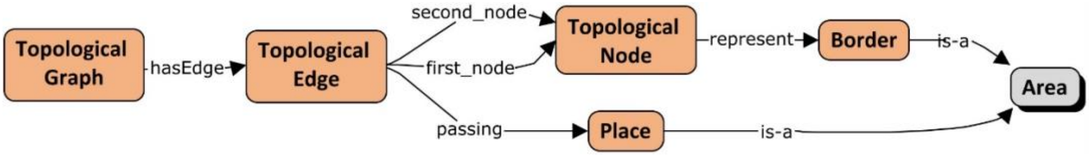


Figure 7. The topological description module

Table 6. Some examples of axioms – The topological description module

Concept	Axiom
Place Border	`SubClassOf: Area`
Topological Node	`SubClassOf: (represent exactly 1 Border)`
Topological Edge	`SubClassOf:` `(first_node exactly 1 TopologicalNode) and (second_node exactly 1 TopologicalNode) and (passing exactly 1 Place)`
TopologicalGraph	`SubClassOf: (hasEdge min 0 TopologicalEdge)`

4.3 The contextual description module

In applications specific to robotics and virtual reality, the simulated environment is rarely seen from the geometric point of view. The *RigidBodies*, *Places*, and *Borders* are identified with contextual semantics. For example, in an indoor household environment, *RigidBodies* can be a table, a door, or a booklet; the identified *Places* can be bedrooms or corridors; their *Borders* can be the entrances of bedrooms. In the construction of the contextual description module, these contextual semantics are also dependent on the specific application.

In the knowledge modelling literature, a distinction between ontologies and contexts has already been discussed in various kinds of research works [44]–[49]. In [50], such a distinction has been formalized as "ontologies are shared models of some domain that encodes a common view of different parties, whereas contexts are local and non-shared models that encode a party's view on a particular domain".

Following the idea of separating an ontology from its context, the contextual description module of the ontology of a 3D environment consists of two major parts: context-independent semantics and context-specific semantics. Table 8 provides some formal definitions of the concepts of the semantic description module.

4.3.1 The context-independent semantics

The context-independent semantics are mostly the description of the characteristics of RigidBodies, Places and Borders. The characteristics are independent of context because of the fact that their identities are recognized uniformly across different applications and measured with internationally standardized scales and units. Two types of characteristics are admitted in this model following the categorization of BFO [51]: *Quality*, which are apparent characteristics that do not depend on a process for their manifestation, and *Realizable entity*, which are characteristics that can only be exhibited through a certain realizing process. Below, we first describe some of the characteristics under *Quality* category and then some from *Realizable entity* category, which can be applied to either rigid bodies and free space specifically or both.

For example, the *Shape* and *FormConvexity* are types of *Quality* that define how a *RigidBody* looks like. *Object Mobility* defines whether a *RigidBody* is movable. The *Function* defines what a *RigidBody* can do (e.g., cooking, heating). *Shape* describes the appearance of an Area. Additionally, the *Presence of Mobile Obstacle* defines whether an *Area* contains *mobile obstacles* (i.e., *Free, Intersected, or Blocked*). The *EnvironmentComplexity* and *EnvironmentCongestion* describe whether an *Area* is difficult to across. Some context-independent characteristics of *RigidBodies* may also be used as a clause to create new sub-categories under rigid bodies, such as *ShapedObject*, *FunctionalObject*. These categories can also be further defined according to the domain of simulations. For example, The construction domain [28] categorizes construction projects into three groups: building construction, infrastructure construction, and industrial construction.

Quality

- *Shape* quality is used to describe the geometric form of *RigidBodies*, *Places*, and *Borders*. Two subgroups, which are *RegularShapeQuality* and *IrregularShapeQuality*, are further obtained depending on whether their *Shapes* are regular or not. Typical *RegularShapeQualities* are *RSQ_Cylinder*, *RSQ_TriangularPrism*, *RSQ_Cuboid*.

- *Form Convexity*: If a *RigidBody* is convex, the line segment between any two points (in the interior or on the boundary of the *RigidBody*) should not go outside of the *RigidBody*. Otherwise, it is concavely formed.

- *Object Mobility*: A *RigidBody* can be fixed to the ground and cannot be moved during the whole simulation. Otherwise, it is a mobile obstacle that can be moved.

- *Environment Congestion* determines whether a *Place* or a *Border* provides enough space for the manipulated *RigidBody* to pass through. *Wide* and *Narrow* are two instances.
- *Environment Complexity* determines whether a *Place* or a *Border* is complex, e.g., filled with moving obstacles as a dynamic environment. *Complex* and *Not complex* are two instances.
- *Presence of Mobile Obstacle* is defined to specify whether a *Place* or a *Border* contains moving obstacles. *Free* means that no moving obstacle is inside of an *Area*, *Intersected* means that one or several moving obstacles is inside of an *Area*, *Blocked* means that an *Area* is completely covered by a moving obstacle.

Realizable entity

- *Color* of a *RigidBody* can only be exhibited through an optical lighting process. Classical colors are *Black*, *White*, *Red*, *Blue*, *Green*, *Orange*, *Yellow*.
- *Function* of a *RigidBody* is determined at the very beginning of the product design stage. However, it is not an intrinsic property of a *RigidBody*, and it can only be realized during a certain process. For example, the *Fasten* function of a screw can only be sensed in an assembly process of a product.

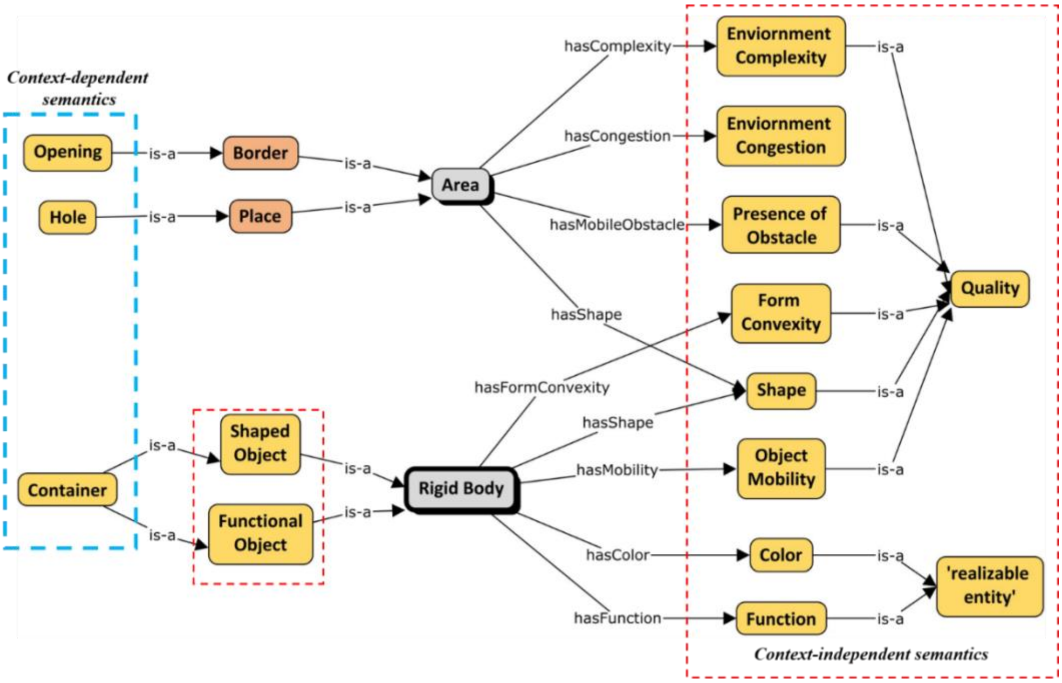


Figure 8. The semantic description module

4.3.2 The context-dependent semantics

The context-dependent semantics of *RigidBodies*, *Places*, and *Borders* relies heavily on the application that a simulated task handles. The concepts and relations are locally defined. The modelling of the context-dependent semantics is a difficult activity as it varies among applications. Currently, context-dependent semantics is not the focus of this study. We only introduce *Hole*, *Opening*, and *Container* as local concepts (Context-dependent semantics in

Figure **8**), so that the environment information of the two scenarios used in this paper (see Section 5.1) can be instantiated in the ontology.

Table 7. Some examples of axioms – The semantic description module

Concept	Axiom
Hole	**SubClassOf:** Place and (hasCentralAxis exactly 1 Axis3D)
Opening	**SubClassOf:** Border and (hasOpeningDirection exactly 1 Vector3D)
ShapedObject	**SubClassOf:** RigidBody and (hasShape exactly 1 Shape)
FunctionalObject	**SubClassOf:** RigidBody and (hasFunction exactly 1 Function)

Container	**SubClassOf:** ShapedObject and FunctionalObject and (hasSpaceInContainer exactly 1 Hole) and (hasOpening exactly 1 Opening)
Area	**SubClassOf:** (hasMobileObstacle exactly 1 PresenceOfObstacle) and (hasComplexity exactly 1 EnvironmentComplexity) and (hasCongestion exactly 1 EnvironmentCongestion) and (hasShape exactly 1 Shape)
Rigid Body	**SubClassOf:** (hasFormConvexity exactly 1 FormConvexity) and (hasColor exactly 1 Color) and (hasFunction exactly 1 Function) and (hasMobility exactly 1 ObjectMobility) and (hasShape exactly 1 Shape)

5. Ontology Validation

The ontology verification evaluates whether an ontology is built correctly against ontology specification documents and correctly represents the intended model of the world aiming to conceptualize. To verify and validate the proposed ontology of the 3D environment for simulating manipulation tasks, we instantiate the ontology with real environment data of two scenarios and we examine whether the instantiated ontology can answer correctly the competency questions listed in section 3. In this research work, SPARQL (SPARQL Protocol and RDF Query Language) is used as the query language to retrieve data from the ontology. More details about the added value of this ontology from a practical point of view (especially on path planning) can be found here in the work of Zhao et al. [52].

5.1 SIMULATION SCENARIOS

The first case study concerns inserting a pen into a narrow penbox. Controlling the path planning process with geometric constraints provides a higher possibility of finding a collision-free trajectory for the insertion. The second case study introduces the shape attribute, which makes the insertion even harder.

5.1.1 Scenario 1: Pen-Penbox Insertion Use Case

The "workspace" of the simulation environment for pen-penbox insertion use case (

Figure 9-a) is the 3D Cartesian Space bounded by a line cube. Two obstacles can be found: *Pen1* is a mobile obstacle and *Penbox1* is a fixed obstacle. The objective of the task simulation is to insert *Pen1* into *Penbox1*, where *pen_goal* is the configuration where the *Pen1* should reach. The *pen_goal* is obtained by pre-sampling within *Penbox1* (*bounding box* or *P2*).

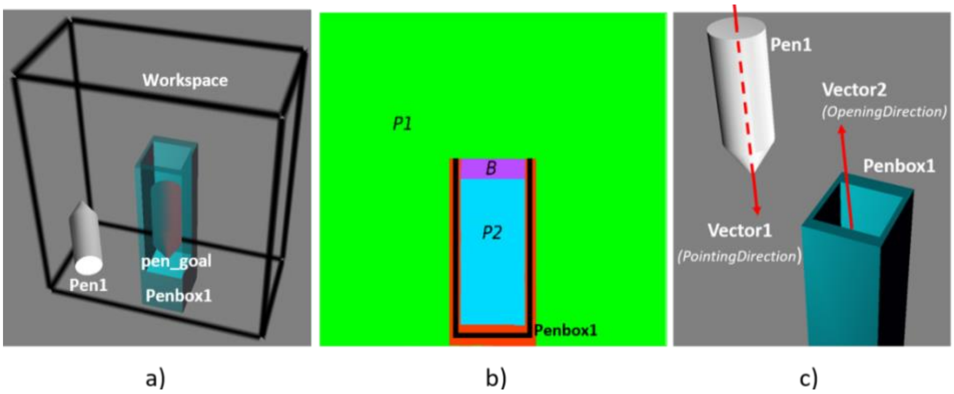


Figure 9. A pen-penbox insertion use case

Figure 9-b demonstrates the construction of the topological level of the free space model (i.e. two places (*P1*, *P2*) and one border (*B*)) from the geometric level of the free space model (i.e., cell decomposition of the workspace). Compared to the size of *Pen1*, *P1* is enriched with the complexity attribute *Free* and *P2* is *Narrow*. This allows applying geometric constraints differently in *P1* and *P2*.

Figure 9-c demonstrates an example when Pen1 is pointed to Penbox1 (i.e. *Vector1* is against *Vector2*). This constraint will be used in identifying the path when *Pen1* is inserted into *Penbox1*.

5.1.2 Scenario 2: Shape Embedding Game

A more complex use case is inspired by the shape embedding game for children. Along with the geometric constraints in the pen-penbox insertion use case, it also requires matching the shape between the hole and the manipulated object.

The 3D environment for the simulation (Figure 10) constitutes a cuboid workspace cluttered with five rigid bodies (*O1* to *O5*). *O1* is fixed and *O2* to *O5* are moveable. Five different places (*P1* to *P5*) are identified at the topological level of the 3D environment's free space model. Semantically, *P2* to *P5* are defined as *O1*'s holes, and they respectively have the shape *(Quality:Shape)* of *RSQ_Cylinder*, *RSQ_Cuboid*, *RSQ_PentagonPrism*, and *RSQ_Triangular Prism*. *O2* to *O5* have the shape *(Quality:Shape)* of *RSQ_Cylinder*, *RSQ_TriangularPrism*, *RSQ_Cuboid*, and *RSQ_PentagonPrism*. The objective of the task simulation is to insert *O2* to *O5* into holes with the same shape (i.e. *O2* into *P2*, *O3* into *P5*, *O4* into *P3*, and *O5* into *P4).*

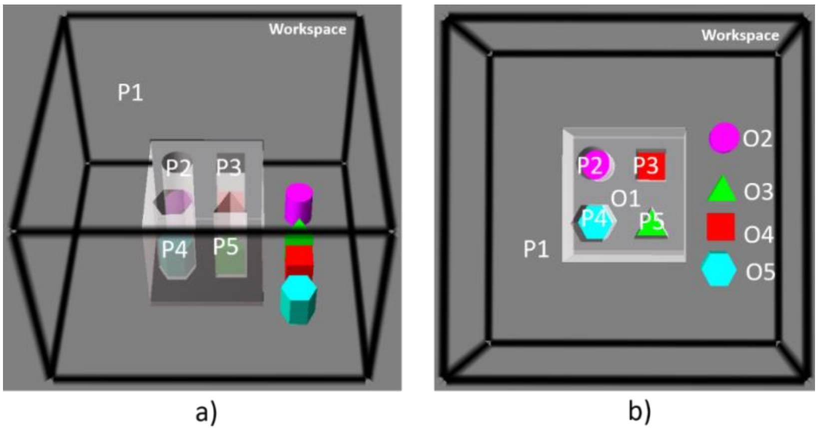


Figure 10 : Shape Embedding Game

5.2 Verification and Validation of the ontology of 3D environment: Scenario 1

Firstly, ENVOn is instantiated with the environment data of the pen-penbox insertion scenario (Figure 11). For example, *Pen1* is an instance of *Pen* and thus an instance of *Rigid body*. *Pen1* has different object properties, such as *CentralAxis_Pen1* (*Axis3D*) as its central axis, *Vector1*(*Vector3D*) as its pointing direction, and *CylindricalVolume1* (*CylindricalVolume*) as its standard form. *SweepingDir_CylindricalVolume1* (*Vector3D*) and *SweepingPlane_CylindricalVolume1* (*Circle3D*) are respectively the sweeping direction and the sweeping plane of the *Pen1*'s *Volume3D* (*CylindricalVolume1*).

After instantiating the environment data in the ontology, we design and define some competency questions in Table 9 to validate the correctness of ENVOn. The evaluation also shows the facility of fast querying the environment data. For example,

- "What is the central axis of Pen1?" is straightforward to search for the *Pen1*'s central axis (i.e., *CentralAxis_Pen1*),
- "What is the opening direction of P2?" and "What is the pointing direction of Pen1?" query the direction (*Vector3D*) where *P2* opens or *Pen1* points, i.e., *Vector1* and *Vector2*.

Table 9. Competency Questions - Querying geometric details

Rigid Body / Place	Competency Questions	Result
Pen1 (Type: Pen)	What is the central axis of Pen1?	CentralAxis_Pen1
	What are the sweeping plane and the sweeping direction of Pen1's Volume?	SweepingDir_CylindricalVolume1 SweepingSurface_CylindricalVolume1
	What is the pointing direction of Pen1?	Vector1 (see Figure 12)
Penbox1 (Type: Penbox)	What is the opening direction of Penbox1?	OpeningDirection_penbox1
P2 (Type: Place)	What is the opening direction of P2?	Vector2
	What are the sweeping plane and the sweeping direction of *P2*'s Volume?	SweepingDir_BlockVolume SweepingSurface_BlockVolume

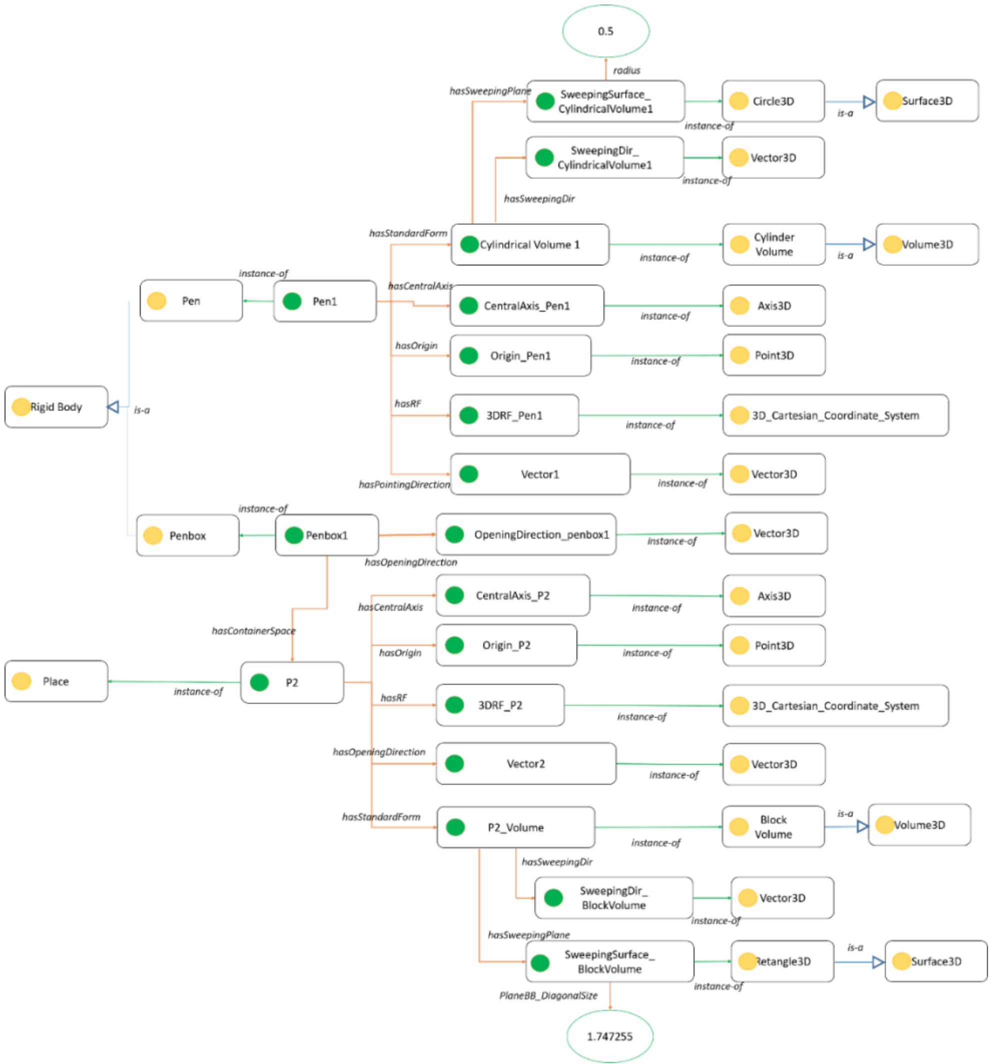


Figure 11 : *Instantiated Ontology for 3D Environment of the Pen-penbox insertion scenario*

In **Error! Reference source not found.**-a, we demonstrate an example of SPARQL query to search for the pointing direction of *Pen1* (*?pointing_dir*) and the local reference frame in which *?pointing_dir* is defined. Figure 12-b and c respectively show the obtained results and their visual display in Virtools.

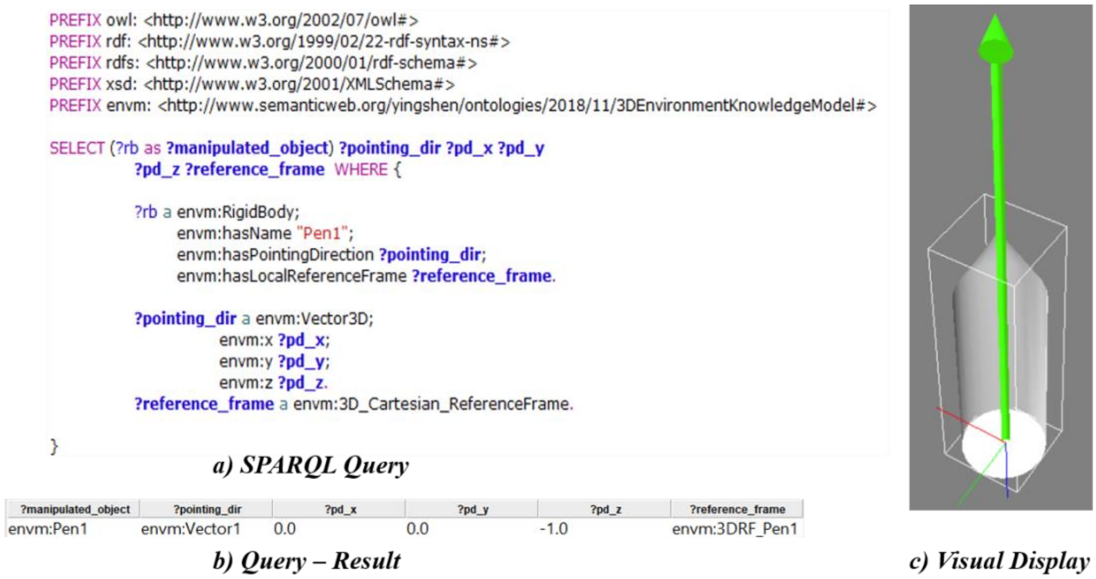
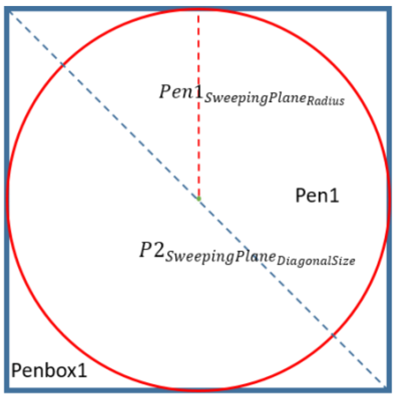


Figure 12. SPARQL Query Result - Pen1

Moreover, to correctly insert *Pen1* into *Penbox1*, we have to determine whether *Pen1* can be inserted into *Penbox1* first. In this scenario, it means whether *Pen1* can be inserted into *P2*. The competency question of this issue is described in Table 10. *P2* is a *Place* that has a standard form *BlockVolume*, and *Pen1* is a *Rigid Body* that has a standard form *CylindricalVolume*. Both standard forms of *P2* and *Pen1* have the regular sweeping plane *Retangle3D* and *Circle3D*, i.e., the length of *P2*'s *Retangle3D* is equal and *Pen1*'s *Circle3D* is round. Therefore, the condition of whether *Pen1* can be inserted into *P2* is shown in Figure 13.



$$Pen1_{SweepingPlane_{Radius}} < 0.35355 * P2_{SweepingPlane_{DiagonalSiz}}$$

Figure 13*. The condition of whether Pen1 can be inserted into P2*

Table 10. Competency Question – The possibility of inserting Pen1 into Penbox1

Competency Question	Result
Whether Pen1 can be inserted into P2?	1 (meaning: Yes) Pen1_SweepingPlane_Radius: 0.5 P2_SweepingPlane_DiagnalSize: 1.747255

In Figure 14-a, we demonstrate an example of SPARQL query to search for the radius of the sweeping plane of *Pen1* (*?rigid_body_sweeping_plane_diagonal_size*) and the diagonal size of the sweeping plane of *P2* (*?place_sweeping_plane_diagonal_size*). In Figure 14-b, we demonstrate an example of SPARQL query to determine whether *Pen1* can be inserted into *P2*. There is one result *(?number_of_result = 1*) so *Pen1* can be inserted into *Penbox1* in this use case.

```
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 xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX envm: <http://www.semanticweb.org/yingshen/ontologies/2018/11/3DEnvironm

SELECT (?rbsp_size as ?rigid_body_sweeping_plane_diagnoal_size)
        (?placesp_size as ?place_sweeping_plane_diagonal_size)
WHERE {
            ?rb_volume a envm:CylinderVolume.
            ?rb a envm:RigidBody;
                envm:hasName "Pen1";

                envm:hasStandardForm ?rb_volume;

             envm:isParallel_Pointing_or_OpeningDir_with_SweepingDir "true".

             ?rb_volume envm:hasSweepingPlane ?rb_sweeping_plane.
            ?rb_sweeping_plane a envm:Circle3D;
                          envm:radius ?rbsp_size.

             ?place_volume a envm:BlockVolume.
            ?place a envm:Place;
                    envm:hasName "P2";
                    envm:hasStandardForm ?place_volume;
                    envm:isParallel_Pointing_or_OpeningDir_with_SweepingDir "true".
            ?place_volume envm:hasSweepingPlane ?place_sweeping_plane.
            ?place_sweeping_plane a envm:Retangule3D;
                                envm:PlaneBB_DiagnalSize ?placesp_size.

}
```

?rigid_body_sweeping_plane_diagnoal_size	?place_sweeping_plane_diagonal_size
0.5	1.747255

a) Sweeping plane's radius or diagonal size

```
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 xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX envm: <http://www.semanticweb.org/yingshen/ontologies/2018/11/3DEnvironm

SELECT (COUNT (?rb) as ?number_of_result)
WHERE {
            ?rb_volume a envm:CylinderVolume.
            ?rb a envm:RigidBody;
                envm:hasName "Pen1";

                envm:hasStandardForm ?rb_volume;

             envm:isParallel_Pointing_or_OpeningDir_with_SweepingDir "true".

             ?rb_volume envm:hasSweepingPlane ?rb_sweeping_plane.
            ?rb_sweeping_plane a envm:Circle3D;
                          envm:radius ?rbsp_size.

             ?place_volume a envm:BlockVolume.
            ?place a envm:Place;
                    envm:hasName "P2";
                    envm:hasStandardForm ?place_volume;
                    envm:isParallel_Pointing_or_OpeningDir_with_SweepingDir "true".
            ?place_volume envm:hasSweepingPlane ?place_sweeping_plane.
            ?place_sweeping_plane a envm:Retangule3D;
                                envm:PlaneBB_DiagnalSize ?placesp_size.
             FILTER (?rbsp_size <= 0.35355*?placesp_size)
}
```

?number_of_result
1

b) Result

Figure 14. *SPARQL Query - – The possibility of inserting a pen into a penbox*

5.3 Verification and Validation of the ontology of 3D environment: Scenario 2

First, the environment data of the shape embedding game scenario is firstly instantiated in the ontology of 3D environment, as shown in Figure 15. This data consists of the contextual, topologic, and geometric information of the 3D environment where the simulated task takes place.

At the geometric level, different geometric properties of rigid bodies and places are captured. In the shaped game scenario, *O3* is an instance of *RigidBody* and it has a pointing direction *Vector2* (type: *Vector3D*), a local reference frame *3DRF_O3* (type: *3DCartesianReference Frame*), an origin *Origin_O3* (*Point3D*), a central axis *CentralAxis_O3* (*Axis3D*), and a standard form *TriangularPrismVolume1* (type: *TriangularPrismVolume*). *O3*'s standard form has a sweeping plane *SweepingSurface_TriangularPrismVolume1* (type: *Triangle3D*) and has a sweeping direction *SweepingDir_TriangularPrimsVolume1* (type: *Vector3D*). The sweeping plane's diagonal size is *0.31145*. *P5* is an instance of *place*; it has a standard form *P5_Volume* (type: *TriangularPrismVolume*) with a different diagonal size 0.*3115* of the sweeping plane *SweepingSurface_P5Volume* (type; *TrangularPrismVolume*). At the topological level, *P1*, *P2*, *P3*, *P4*, *P5* are five different places constructed. *P2-P5* has direct topological connections with *P1*. At the semantic level, *O3* is further defined as an instance of *Pen*. *P2-P5* are instances of *Hole* (they are narrow and have the shape of *RSQ_Cylinder*, *RSQ_Block*, *RSQ_TriangularPrism*, *RSQ_PentagnonPrism*, respectively).

We design and define some competency questions in Table 11 and Table 12. We can see from the obtained results that ENVOn can correctly answer these questions.

Table 11. Competency Question - Query geometric details

Rigid Body	Competency Question	Result
O3	What is the central axis of O3?	CentralAxis_O3
	What is the pointing direction of O3?	Vector2
	What are the sweeping plane and the sweeping direction of O3's Volume?	SweepingDir_TriangularPrismVolume1 SweepingSurface_TriangularPrismVolume1 (see Figure 16)
	What is the symmetric vector for the sweeping plane of P5's Volume	Vector4

Table 12. Competency Question - Query geometric details (P5)

Place	Competency Question	Result
P5	What is the central axis of P5?	CentralAxis_P5

	What is the opening direction of P5?	Vector1
	What are the sweeping plane and the sweeping direction of P5's Volume?	SweepingDir_P5Volume SweepingSurface_P5Volume (see Figure 17Figure 16)
	What is the symmetric vector for the sweeping plane of P5's Volume	Vector3

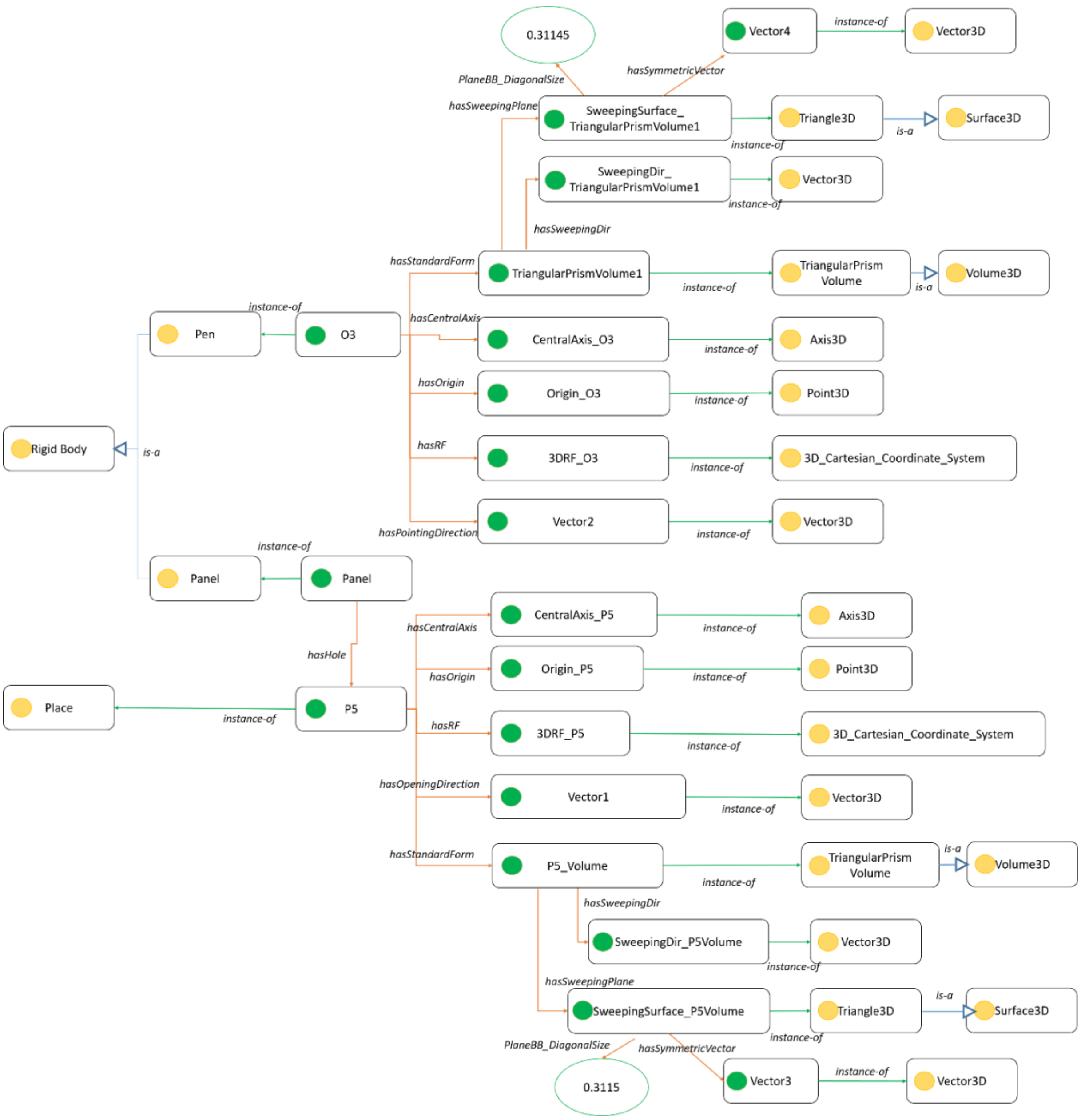


Figure 15*. Instantiated Ontology for 3D Environment of the Shape Embedding Game Scenario*

In Figure 16 and Figure 17, we demonstrate two examples of SPARQL queries to respectively search for the sweeping plane and the sweeping direction of *O3* and *P5*, the obtained results, and their visual display in Virtools.

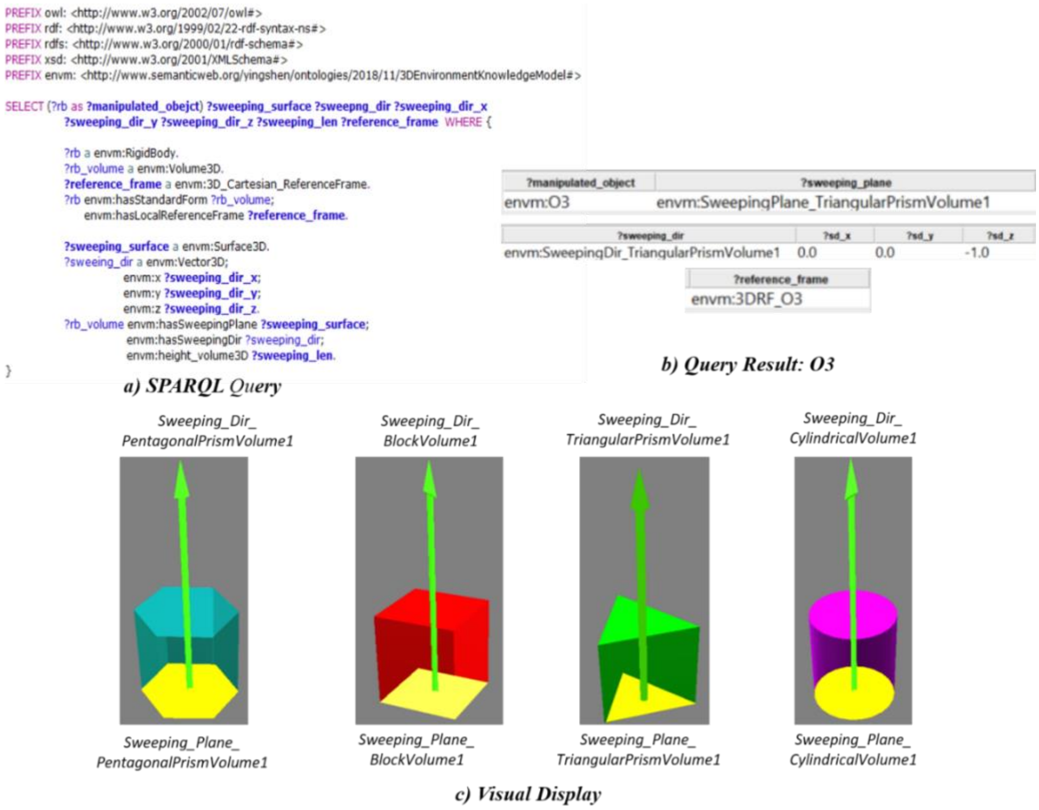


Figure 16. SPARQL Query Result - O3

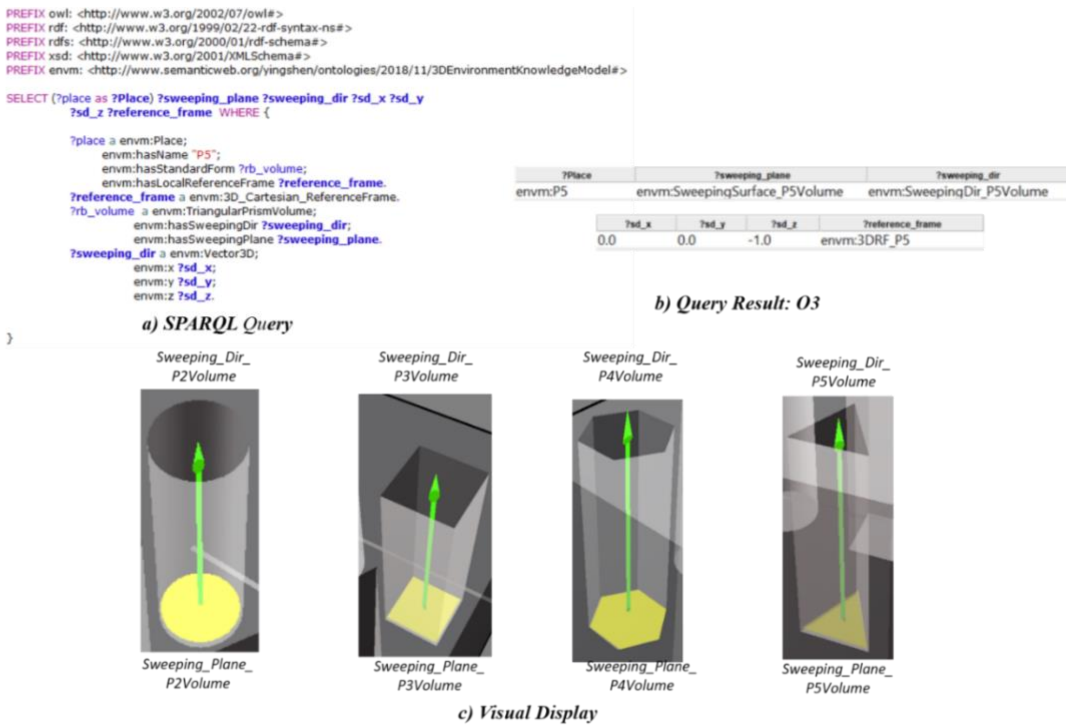


Figure 17. SPARQL Query Result - P5

In Table 9 we propose two derivations: the first one specifies that the diagonal size of *P5*'s sweeping plane is 0.3115 and the second one 0.31135. A competency question is defined to check whether *O3* can be inserted into *P5*. Because both *O3* and *P5* have standard forms *TriangularPrismVolume*, the result relies on the diagonal size of *O3*'s and *P5*'s sweeping planes. Because the diagonal size of *O3*'s sweeping plane is smaller than the one of *P5*, *O3* can be inserted into *P5* in derivation 1. Otherwise, it is impossible to perform this primitive action (see derivation 2).

Table 8. Competency Question – The possibility of inserting the triangular prism pen into P5

Competency Questions	Derivation	Result
Whether O3 can be inserted into P5?	Derivation 1	1 (meaning: Yes) O3_SweepingPlane_DiagnalSize: 0.31145 P5_SweepingPlane_DiagnalSize: 0.3115
	Derivation 2	0 (meaning: No) O3_SweepingPlane_DiagnalSize: 0.31145 P5_SweepingPlane_DiagnalSize: 0.31135

In Figure 18-a, we demonstrate an example of SPARQL query to search for the radius of the sweeping plane of *O3* (*?rigid_body_sweeping_plane_diagonal_size*) and the diagonal size of the sweeping plane of *P5* (*?place_sweeping_plane_diagonal_size*). In Figure 18-b, we demonstrate an example of SPARQL query to determine whether *O3* can be inserted into *P5*. No result can be found (*?number_of_result = 0*) so that *O3* cannot be inserted into *P5* in derivation 2.

```
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 xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX envm: <http://www.semanticweb.org/yingshen/ontologies/2018/11/3DEnvironmentKno

SELECT(?rbsp_size as ?rigid_body_sweeping_plane_diagnoal_size)
      (?placesp_size as ?place_sweeping_plane_diagonal_size)
WHERE {
        ?rb_volume a envm:TriangularPrismVolume.
        ?rb a envm:RigidBody;
            envm:hasName "O3";
            envm:hasStandardForm ?rb_volume;
            envm:isParallel_Pointing_or_OpeningDir_with_SweepingDir "true".
        ?rb_volume envm:hasSweepingPlane ?rb_sweeping_plane.
        ?rb_sweeping_plane a envm:Triangle3D;
                        envm:PlaneBB_DiagnalSize ?rbsp_size.


        ?place_volume a envm:TriangularPrismVolume.
        ?place a envm:Place;
                envm:hasName "P5";
                envm:hasStandardForm ?place_volume;
                envm:isParallel_Pointing_or_OpeningDir_with_SweepingDir "true".
        ?place_volume envm:hasSweepingPlane ?place_sweeping_plane.
        ?place_sweeping_plane a envm:Triangle3D;
                        envm:PlaneBB_DiagnalSize ?placesp_size.

}
```

?rigid_body_sweeping_plane_diagnoal_size	?place_sweeping_plane_diagonal_size
0.31145	0.31135

a) Sweeping plane's radius or diagonal size

```
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 xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX envm:
<http://www.semanticweb.org/yingshen/ontologies/2018/11/3DEnvironmentKnowledgeModel#>

SELECT (COUNT (?rb) as ?number_of_result)
WHERE {
        ?rb_volume a envm:TriangularPrismVolume.
        ?rb a envm:RigidBody;
            envm:hasName "O3";
            envm:hasStandardForm ?rb_volume;
            envm:isParallel_Pointing_or_OpeningDir_with_SweepingDir "true".
        ?rb_volume envm:hasSweepingPlane ?rb_sweeping_plane.
        ?rb_sweeping_plane a envm:Triangle3D;
                        envm:PlaneBB_DiagnalSize ?rbsp_size.


        ?place_volume a envm:TriangularPrismVolume.
        ?place a envm:Place;
                envm:hasName "P5";
                envm:hasStandardForm ?place_volume;
                envm:isParallel_Pointing_or_OpeningDir_with_SweepingDir "true".
        ?place_volume envm:hasSweepingPlane ?place_sweeping_plane.
        ?place_sweeping_plane a envm:Triangle3D;
                        envm:PlaneBB_DiagnalSize ?placesp_size.

        FILTER (?rbsp_size <= ?placesp_size)
}
```

?number_of_result
0

b) Result

Figure 18 SPARQL Query – The possibility of inserting O3 into P5 (derivation 2)

6. Conclusion and future work

The primary motivation for this work is to link different levels of environment information using formal semantics tightly. Such knowledge formalization allows answering semantically meaningful queries, such as "Does *ObjectA* fits *HoleB*?". "What is the central axis of *ObjectA*?". Moreover, to a certain extent, the knowledge reasoning capability using ontology (in terms of DL logics) allows a path planning system to make decisions on its own. For example, answering to a given *Insert* primitive action to be performed, the path planning system can decide the most appropriate *Hole* to reach by exploring the environment ontology.

In the geometric description module, the geometries of rigid bodies (based on CAD models) have not taken into account all the criteria of the STEP standard. Currently, the geometry of the free space model considered only the octree decomposition of the simulation environment. In the topologic description module, "border as node, place as arc" is not necessarily the schema in all cases. In the semantic description module, the taxonomies of rigid bodies, places, and borders are very locally defined and the modelling of the relative locations between rigid bodies or between a rigid body and a place in 3D space should also be considered. Finally, different levels of abstraction (top-level, domain- and application-specific) should be considered in building an ontology in general. By aligning with the top-level ontology (e.g. BFO, DOLCE), the proposed ontology will become interoperable with other domain ontologies by making the correspondence between similar concepts have different names.

One thing that draws our attention, during the ontology development, is that not all environment information is suitable to be instantiated in an ontology, for example, polygonal models based on

Delaunay triangulation that contains a large amount of raw data. The number of these points and lines might be large and they are sometimes semantically meaningless. Saving such geometric information in the ontology makes the knowledge base so overstaffed that the knowledge querying and reasoning can be slow, and even sometimes impossible. Indeed, such a kind of issue does not only happen in our ontology development. Rather, it is a common issue in the scientific community to build proper ontologies. Moreover, we expect that ENVOn can serve as a belief for the planning system so that the ontology can be updated whenever the belief changes: some things are added, and some things are deleted. For OWL and SWRL that are monotonic in terms of logic, it is difficult to modify the already constructed ontology. Currently, our ontology has not taken the iterative environment update into account, and it only concerns the environment state now when a primitive action of manipulating an object takes place. Moreover, the calculation support using OWL and SWRL is limited [53]]. In our research, only simple numerical comparisons are used, e.g., to find out whether a *cylinder* object can fit the *cylinder* hole by comparing its radius. Complex mathematical computations are not suitable to be modelled by logic but rather defined by external functions (e.g. Java), however, OWL and SWRL lack the mechanism to link the predicates with external functions (except simple built-in functions of SWRL).

On the contrary, the geometry of CAD models is semantically meaningful. For example, rather than thousands of meaningless triangular faces, the *Cylindrical* surface can be defined as a surface having an origin point, a central axis, and a radius. Similar semantically meaningful geometric information can be found everywhere in CAD models. However, the existing formats (e.g. STEP) of CAD models do not take advantage of the semantically meaningful data to allow knowledge querying and reasoning. Therefore, conceptualizing CAD models using ontology is worth research.

7. Acknowledgements

This work is partially funded by the European Union's Horizon 2020 project OntoCommons under Grant Agreement no. 958371

8. References

[1] E. Vassev and M. Hinchey, "Knowledge Representation for Cognitive Robotic Systems," in *2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops*, 2012, pp. 156–163, doi: 10.1109/ISORCW.2012.36.

[2] F. Ameri, D. Sormaz, F. Psarommatis, and D. Kiritsis, "Industrial ontologies for interoperability in agile and resilient manufacturing," *Int. J. Prod. Res.*, pp. 1–22, Oct. 2021, doi: 10.1080/00207543.2021.1987553.

[3] I. H. Suh *et al.*, "for Robot Intelligence," pp. 429–436, 2007.

[4] M. Tenorth and M. Beetz, "KNOWROB - Knowledge processing for autonomous personal robots K NOW R OB — Knowledge Processing for Autonomous Personal Robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, no. May, doi: 10.1109/IROS.2009.5354602.

[5] M. Diab, A. Akbari, M. Ud Din, and J. Rosell, "PMK-A Knowledge Processing Framework for Autonomous Robotics Perception and Manipulation.," *Sensors (Basel).*, vol. 19, no. 5, Mar. 2019, doi: 10.3390/s19051166.

[6] A. Akbari and J. Rosell, "Ontological Physics-based Motion Planning for Manipulation," no. September, 2015, doi: 10.1109/ETFA.2015.7301404.

[7] M. Diab, Muhayyuddin, A. Akbari, and J. Rosell, "An Ontology Framework for Physics-Based Manipulation Planning," in *ROBOT 2017: Third Iberian Robotics Conference*, 2017, pp. 452–464.

[8] M. Hirschberg, "The V Model," *Test*, no. 23, pp. 1–9, 2000.

[9] M. M. M. Sarcar, K. MALLIKARJUNA RAO, and L. K. Narayan, *Computer Aided Design and Manufacturing*. PHI Learning Pvt. Ltd., 2008.

[10] S. Cailhol, P. Fillatreau, J.-Y. Fourquet, and Y. Zhao, "A hierarchic approach for path planning in virtual reality," *Int. J. Interact. Des. Manuf.*, vol. 9, no. 4, pp. 291–302, Nov. 2015, doi: 10.1007/s12008-015-0272-5.

[11] S. Cailhol, P. Fillatreau, Y. Zhao, and J. Fourquet, "Multi-layer path planning control for the simulation of manipulation tasks : Involving semantics and topology," *Robot. Comput. Integr. Manuf.*, vol. 57, pp. 17–28, 2019, doi: 10.1016/j.rcim.2018.10.010.

[12] A. Olivares-Alarcos *et al.*, "A review and comparison of ontology-based approaches to robot autonomy," *Knowl. Eng. Rev.*, vol. 34, pp. 1–29, 2019, doi: 10.1017/S0269888919000237.

[13] S. Borgo, A. Cesta, A. Orlandini, and A. Umbrico, "Knowledge-based adaptive agents for manufacturing domains," *Eng. Comput.*, vol. 35, no. 3, pp. 755–779, Jul. 2019, doi: 10.1007/s00366-018-0630-6.

[14] R. B. Rusu, "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments," *KI - Künstliche Intelligenz*, vol. 24, no. 4, pp. 345–348, Nov. 2010, doi: 10.1007/s13218-010-0059-6.

[15] J. P. Dandois and E. C. Ellis, "High spatial resolution three-dimensional mapping of vegetation spectral dynamics using computer vision," *Remote Sens. Environ.*, vol. 136, pp. 259–276, Sep. 2013, doi: 10.1016/j.rse.2013.04.005.

[16] A. Vasquez, A. Dapogny, K. Bailly, and V. Perdereau, "Sequential recognition of in-hand object shape using a collection of neural forests," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2017, pp. 3081–3086, doi: 10.1109/IROS.2017.8206148.

[17] Z. Márton, N. Blodow, M. E. Dolha, M. Tenorth, R. B. Rusu, and M. Beetz, "Autonomous Mapping of Kitchen Environments and Applications," 2008.

[18] N. Blodow *et al.*, "Autonomous semantic mapping for robots performing everyday manipulation tasks in kitchen environments," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sep. 2011, pp. 4263–4270, doi: 10.1109/IROS.2011.6094665.

[19] Il Hong Suh, Gi Hyun Lim, Wonil Hwang, Hyowon Suh, Jung-Hwa Choi, and Young-Tack Park, "Ontology-based multi-layered robot knowledge framework (OMRKF) for robot intelligence," in *2007 IEEE/RSJ International*

Conference on Intelligent Robots and Systems, Oct. 2007, pp. 429–436, doi: 10.1109/IROS.2007.4399082.

[20] M. J. Pratt, “Introduction to ISO 10303—the STEP Standard for Product Data Exchange,” *J. Comput. Inf. Sci. Eng.*, vol. 1, no. 1, pp. 102–103, Jan. 2001, doi: 10.1115/1.1354995.

[21] A. L. Clark and S. M. Staley, “STEP AP203 Data Exchange Study,” in *Proceedings of the Third ACM Symposium on Solid Modeling and Applications*, 1995, pp. 213–224, doi: 10.1145/218013.218063.

[22] R. Barbau *et al.*, “OntoSTEP: Enriching product model data using ontologies,” *Comput. Des.*, vol. 44, no. 6, pp. 575–590, Jun. 2012, doi: 10.1016/j.cad.2012.01.008.

[23] A. Perzylo, N. Somani, M. Rickert, and A. Knoll, “An ontology for CAD data and geometric constraints as a link between product models and semantic robot task descriptions,” in *IEEE International Conference on Intelligent Robots and Systems*, 2015, vol. 2015-Decem, pp. 4197–4203, doi: 10.1109/IROS.2015.7353971.

[24] A. Sarkar and D. Šormaz, “On Semantic Interoperability of Model-based Definition of Product Design,” *Procedia Manuf.*, vol. 38, pp. 513–523, 2019, doi: 10.1016/j.promfg.2020.01.065.

[25] D. Demyen and M. Buro, “Efficient triangulation-based pathfinding,” *Proc. Natl. Conf. Artif. Intell.*, vol. 1, pp. 942–947, 2006.

[26] R. A. Brooks and T. Lozano-Perez, “A subdivision algorithm in configuration space for findpath with rotation,” *IEEE Trans. Syst. Man. Cybern.*, vol. SMC-15, no. 2, pp. 224–233, Mar. 1985, doi: 10.1109/TSMC.1985.6313352.

[27] H. Samet, “The Quadtree and Related Hierarchical Data Structures,” *ACM Comput. Surv.*, vol. 16, no. 2, pp. 187–260, 1984, doi: 10.1145/356924.356930.

[28] D. Kortenkamp and T. Weymouth, “Topological mapping for mobile robots using a combination of sonar and vision sensing,” *Proc. Natl. Conf. Artif. Intell.*, vol. 2, pp. 979–984, 1994.

[29] G. Dedeoglu, M. J. Mataric, and G. S. Sukhatme, “<title>Incremental online topological map building with a mobile robot</title>,” Nov. 1999, pp. 129–139, doi: 10.1117/12.369248.

[30] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon, and F. Savelli, “Local metrical and global topological maps in the hybrid spatial semantic hierarchy,” in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, 2004, pp. 4845-4851 Vol.5, doi: 10.1109/ROBOT.2004.1302485.

[31] S. C. Hirtle and J. Jonides, “Evidence of hierarchies in cognitive maps,” *Mem. Cognit.*, vol. 13, no. 3, pp. 208–217, May 1985, doi: 10.3758/BF03197683.

[32] T. P. McNamara, “Mental representations of spatial relations,” *Cogn. Psychol.*, vol. 18, no. 1, pp. 87–121, Jan. 1986, doi: 10.1016/0010-0285(86)90016-2.

[33] Ó. M. Mozos, P. Jensfelt, H. Zender, G.-J. M. Kruijff, and W. Burgard, “From Labels to Semantics: An Integrated System for Conceptual Spatial Representations of Indoor Environments for Mobile Robots,” *Proc. the{I}nternational{C}onference on{I}ntelligent {R}obots {S}ystems {W}orkshops*, 2007.

[34] C. Galindo, J.-A. Fernández-Madrigal, J. González, and A. Saffiotti, “Robot task planning using semantic maps,” *Rob. Auton. Syst.*, vol. 56, no. 11, pp. 955–966, Nov. 2008, doi: 10.1016/j.robot.2008.08.007.

[35] A. Nüchter and J. Hertzberg, “Towards semantic maps for mobile robots,” *Rob. Auton. Syst.*, vol. 56, no. 11, pp. 915–926, Nov. 2008, doi: 10.1016/j.robot.2008.08.001.

[36] H. Zender, O. Martínez Mozos, P. Jensfelt, G.-J. M. Kruijff, and W. Burgard, “Conceptual spatial representations for indoor mobile robots,” *Rob. Auton. Syst.*, vol. 56, no. 6, pp. 493–502, Jun. 2008, doi: 10.1016/j.robot.2008.03.007.

[37] M. Grüninger and M. S. Fox, “Methodology for the Design and Evaluation of Ontologies,” 1995.

[38] J. J. Sylvester, “Sketch of a memoir on elimination, transformation, and canonical forms,” *Cambridge Dublin Math. J.*, vol. 6, pp. 186–200, 1851.

[39] C. M. Hoffmann, *Geometric and solid modeling*. Morgan Kaufmann Pub., 1989.

[40] G. Roth and M. D. Levine, “Geometric primitive extraction using a genetic algorithm,” in *Proceedings 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1992, pp. 640–643, doi: 10.1109/CVPR.1992.223120.

[41] A. Kaiser, J. A. Ybanez Zepeda, and T. Boubekeur, “A Survey of Simple Geometric Primitives Detection Methods for Captured 3D Data,” *Comput. Graph. Forum*, vol. 38, no. 1, pp. 167–196, Feb. 2019, doi: https://doi.org/10.1111/cgf.13451.

[42] L. Piegl and W. Tiller, “Curve and surface constructions using rational B-splines,” *Comput. Des.*, vol. 19, no. 9, pp. 485–498, 1987, doi: https://doi.org/10.1016/0010-4485(87)90234-X.

[43] R. B. Fisher *et al.*, *Dictionary of Computer Vision and Image Processing*. Chichester, UK: John Wiley & Sons, Ltd, 2016.

[44] P. Bouquet, F. Giunchiglia, F. Van Harmelen, V. U. Amsterdam, L. Serafini, and F. B. Kessler, “C-OWL : Contextualizing ontologies C-OWL : Contextualizing Ontologies,” no. May 2014, 2003, doi: 10.1007/978-3-540-39718-2.

[45] A. Firat and S. Madnick, “Information integration using contextual knowledge and ontology merging,” 2003.

[46] H. Stuckenschmidt, F. Van Harmelen, V. U. Amsterdam, L. Serafini, F. B. Kessler, and P. Bouquet, “Using C-OWL for the Alignment and Merging of Medical Ontologies,” no. May, 2004.

[47] D. Benslimane, A. Arara, G. Falquet, Z. Maamar, P. Thiran, and F. Gargouri, “Contextual Ontologies BT - Advances in Information Systems,” 2006, pp. 168–176.

[48] V. Challam, S. Gauch, and A. Chandramouli, *Contextual Search Using Ontology-Based User Profiles.* 2007.

[49] I. Gursel Dino *et al.*, *Contextual ontology support as external knowledge representation for building information modelling*. 2009.

[50] D. Xu, M. H. Karray, and B. Archimède, “A semantic-based decision support platform to assist products' eco-labeling process,” *Ind. Manag. Data Syst.*, vol. 117, no. 7, pp. 1340–1361, Jan. 2017, doi: 10.1108/IMDS-09-2016-0405.

[51] R. Arp, B. Smith, and A. D. Spear, “Basic Formal Ontology at Work,” in Building Ontologies With Basic Formal Ontology, no. June 2017, The MIT Press, 2015, pp. 1–19.

[52] 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, 102255.

[53] A. Sánchez-Macián, E. Pastor, J. E. de López Vergara, and D. López, “Extending SWRL to Enhance Mathematical Support,” 2007, pp. 358–360.