Semantic coupling of path planning and a primitive action of a task plan for the simulation of manipulation tasks in a virtual 3D environment

Yingshen Zhao, Philippe Fillatreau, Linda Elmhadhbi, Mohamed Hedi Karray *, Bernard Archimede

INP-ENIT, Université Fédérale de Toulouse Midi-Pyrénées, Tarbes, France

Keywords:
Path planning
Joint task and path planning
Ontology
Task-oriented knowledge
Knowledge reasoning

A B S T R A C T

This work deals with the simulation of complex manipulation tasks in virtual environments. Validating such complex tasks, possibly to be performed under strong geometric constraints, requires considering task and path planning jointly. The contribution of this work focuses on using task-related information at the path planning level. We propose an ontology-based approach to a) model the 3D environment where the simulated task is executed, based on an original multi-level environment model involving higher abstraction level data than the purely geometric models traditionally used, and b) automatically define path planning queries for the primitive actions of a task plan, together with task-related geometric constraints on these queries. This approach allows the improvement of the state of the art from two points of view. First, our joint task and path planning approach allows the improvement of path planning through better semantic control of the path planning process. Second, if compared to hard-coded geometric constraints, the proposed ontology-based approach introduces a more flexible way of defining geometric constraints through an inference process, and can be adapted to different applications of manipulation tasks.

1. Introduction

The increasingly competitive market and economic competition push industrial companies to reduce the development time and costs of new products. However, today's industrial products are growing in complexity. Accordingly, industrial companies have pointed out the strong need to validate not only their products' static models but also all tasks related to their Product Lifecycle Management (PLM). To tackle these problems, complex industrial operations under strong geometric constraints (e.g. assembly, disassembly, or maintenance) should be simulated and then validated by means of digital prototypes before starting the manufacturing process. That is, digital prototypes enable us to virtually explore a complete product and its possible abnormalities before it is built.

In this work, we are particularly interested in simulating and validating manipulation tasks under strong geometric constraints. In the literature, researchers studied manipulation tasks according to two different approaches: task planning [14,25] and path planning [32,34]. On one hand, task planning decomposes a manipulation task into a feasible sequence of primitive actions (i.e. a task plan). Task planning cannot verify the motion feasibility of primitive actions. On the other hand, path planning demonstrates the feasibility of manipulations by computing collision-free trajectories of the manipulated objects. Traditionally, path planning uses purely geometric data which may lead to some classical limitations such as high processing time, low path relevance of the task to be performed, or failure. Moreover, the task level information has not yet been considered at the path planning level. Beyond attempts for task-oriented motion planning [53], solving a complex manipulation task requires considering task planning and path planning jointly [23,52].

The use of task and path planning jointly is still not common in the state of the art. Most existing solutions start by performing a classical task planning and then check the feasibility of path planning requests associated to the primitive actions [28,35]. However, the link between task planning level and path planning level should be improved, notably because of the lack of loopback between the two levels. Specifically, information used for the path planning query of a task plan is limited to the motion feasibility of primitive actions. We believe that computing an optimal task needs different kinds of task-related information including:

* Corresponding author.
E-mail address: mkarray@enit.fr (M.H. Karray).

- The information of the environment in which the manipulation task to be performed
- The start and goal locations of path planning queries to be executed.
- The geometric constraints (associated to the path planning query) that restrict the movement of a manipulated object.

This work focuses on path planning for a primitive action of a task plan. We propose to use task-related information at the path planning level in order to obtain better path planning results (i.e., lower processing time, better path relevance regarding the task to be performed).

Indeed, the modeling of task-related information varies from one application to another. To build a path planning system that can be used for different applications, it is necessary to have an evolvable knowledge model of task-related information. Different levels of abstraction should be considered. Domain-independent information is commonly shared among applications, whereas application-specific information is updated regarding the task to be performed. Moreover, by using such a knowledge model, the path planning system should be able to infer new information to generate a relevant path planning query.

Accordingly, we propose an ontology-based approach to use task level information to specify path planning queries for the primitive actions of a task plan. It consists of three main parts:

- An ontology that conceptualizes the knowledge of the 3D environment in which the simulated task will take place. The 3D environment is represented by a digital model relying on a multilayer architecture involving contextual, topologic, and geometric data. The originality of the proposed ontology is that it defines knowledge about both the obstacles and the free space models.
- An ontology that defines action-specific knowledge based on the 3D environment ontology. It models the spatial constraints needed for the specification of a primitive action and the geometric constraints that will be used in the description of a path planning query.
- An ontology-based method that generates a path planning query for a primitive action of a task plan. Based on a primitive action instantiated in the action-specific knowledge ontology, by exploring the instantiated 3D environment ontology, and through a reasoning process, we can infer the start, the goal configurations, and the task-related geometric constraints on the path planning query.

The rest of the paper is organized as follows. Section 2 provides an overview of existing works on joint task and path planning, multi-level environment modeling, and existing ontologies in the field. Section 3 details the proposed approach. The implementation and the obtained results are discussed in Section 4. Two use-cases are used to validate the proposed approach. For each use-case, the generation of a path planning query for a primitive action of a task plan is presented and the path planning results are compared. In Section 5, we conclude with a brief discussion of future work.

2. Related works

2.1. The joint use of task planning and path planning

Solving a complex manipulation task problem requires considering task planning and path planning jointly [44]. In the literature, different joint task and path planning techniques are proposed. We categorized them into two main classes according to when the geometric feasibility of a primitive action in the task plan is verified; "task planning then path planning iteratively" [15,19,35], and "using path planning during task planning" [11,12,18,20,26,50].

A key issue of joint task and path planning concerns the nature of the exchanged information between task planning level and path planning level and how this information is used. The most commonly used information feedback is the motion feasibility of primitive actions of a task plan [31,48]. If a path planner fails to verify the motion feasibility of a candidate primitive action, the joint task and path planning process successively performs the following steps:

- It generates an alternative path planning query for the current candidate primitive action.
- If the number of path planning query attempts exceeds a given threshold, it looks for an alternative primitive action or an alternative task plan.

In the literature, two main kinds of task-related information are used at the path planning level. First, *task-related symbolic environmental information* often comes along with the task description to enable the interpretation of primitive actions [22,40]. It defines the types of objects, locations (e.g. refrigerator), their inner space (e.g. the refrigerator is in a kitchen), and their properties (e.g. shape of the refrigerator). It is achieved by a proper semantic mapping between the symbolic environment representation and the geometric models [36]. Second, *task-related geometric constraints* determine the final placement of a manipulated object [4] or search for the final pose of the assembly objects [38].

Both task planning and path planning have been well studied by the artificial intelligence and robotics community. However, there is a lack of loopback between these two levels, specifically;

- The path planning information used to question the task plan is limited to the motion feasibility while richer information is needed such as the relevance and the complexity of the proposed path.
- Path planning queries mainly use purely geometric data and/or "blind" path planning methods (e.g. RRT [33]), and no task-related information has been considered at the path planning level to control the motions of a manipulated object.

2.2. Task-related information

There are two different kinds of information at the task level: the *3D environment information* that is related to a simulated manipulation task, and the t*ask-related geometric constraints* that restrict the final pose of a manipulated object. The environment where a simulated manipulation task takes place is considered as a closed part of 3D Cartesian space cluttered with mobile/fixed obstacles (regarded as rigid bodies). Cailhol et al. [10] propose an environment model that consists of a rigid bodies model and a free-space model. Both of them involve different levels based on semantic, topologic, and geometric information. This work is especially interested in such a multi-level environment model and studies the nature of the environment information at each level.

Two main classes of information associated with rigid bodies have been considered in the Computer-aided Design (CAD) models and the robotics literature:

- Geometric information can be modeled by two major representative schemata; volume representation [41] or surface representation [27]. In Boundary Representation (BRep), the surface description can be polygonal [8] or parametric [6]. In particular, our work concerns more polygonal representation. In such a representation, the surface is characterized by meshes consisting of vertices, edges, and faces. The accuracy of this representation heavily relies on the size of the faces and the sampling resolution on the modeled surface.
- Semantic information of rigid bodies must be adapted to the applications' tasks. It includes the categorization of rigid bodies, their functionalities, and the hierarchical part structure.

The human's understanding of an environment has different levels of abstraction. Indeed, a free space model of an environment should consider all the geometric, topologic, and contextual information. In the robotic applications, the contextual information is used together with the geometric and the topologic levels of the free space model. The

resultant model allows a path planning system to find the goal location to reach.

In order to associate semantic information with the geometric description of 3D object models, the 'Semantic (3D) object map' has been developed [42]. The resultant model allows finding the geometric description for a given symbolic term, e.g. the position of the handle of 'Container A'.

The geometric constraints, that restrict the movement of a manipulated object, are formulated as equality or inequality mathematical expressions [26] and must be adapted to the applications. The distance and the reference angle constraints proposed in [26] are used to compute the destination pose that a robot has to reach. In [46], the geometric constraint is specified as a kind of relationship that exists between a part of a tool and a part of an object in the world ('left of', 'right of', 'above of', 'below of', 'in front of' and 'behind'). In [4], a placement constraint is used to determine how a water cup should be put on a table. It represents the relatively stable position of a manipulated object at a given location. Moreover, the grasp constraint determines how an object can be grasped and the kinematic constraint determines a robot's kinematic capability [51]. In our work, these two constraints are out of consideration and the manipulated objects are studied as free-flyers.

To summarize, whenever an environment changes (e.g. the shape of the cup, the table), the definition of task-related information should be reconsidered. The modeling of task-related information relies on the task to be performed. It is often locally defined and not evolvable.

2.3. The knowledge modeling of task-related information

The knowledge modeling of task-related information is needed for a path planning system to act intelligently [49]. The decidability of the description logic (DL) makes reasoning on the taxonomic knowledge powerful enough to discover implicit and hidden relationships between concepts. Such a capability allows extending the use of ontologies to more applications, such as cleaning and fixing the knowledge of a mobile robot in a dynamic and unknown environment. The ontology models based on DLs can provide a precise and easy way to model concepts of a given domain and their relations.

In manipulation task simulations, a proper knowledge model of task-related information allows a path planning system to automatically generate a path planning query for a primitive action of a task plan. The start and the goal configurations, as well as task-related geometric constraints, can be generated. Such information could then be used to control the path planning of this primitive action [24]. In the literature, several works showed the added value of ontology driven approaches for task planning and path planning [5,24,29].

Although various knowledge models have already been developed [17,37,45,47] and used [1,2,16] in the robotics domain, they rarely consider in-depth how to use task-related information to control motions of a manipulation object or a robot in path computation.

Several works have conceptualized the geometries of rigid bodies based on CAD models such as OntoSTEP [39] and OntoBREP [38], whereas conceptualizing the geometries of the free space model is still lacking.

Few works have discussed the modeling of geometric constraints using ontologies. To properly generate task-related geometric constraints on a path planning query associated with a primitive action, the study of the modeling of action-specific knowledge is needed [13]. The definitions of geometric constraints on a primitive action are difficult because these constraints are often hard to be understood by common users. In order to make the constraints definition of a primitive action more intuitive, this work is interested in taking advantage of human's spatial knowledge of a 3D environment where a simulated manipulation task takes place. The spatial knowledge provides a task/path planning system valuable spatial information on where primitive actions take place [3]. We focus on the spatial knowledge describing how two objects are relatively located in a 2D/3D environment, noted as "Spatial Relation" [43]. Spatial relations consist of three main categories; distance relation, topological relation, and orientation / directional relation [7].

To summarize, no work has considered all kinds of task-related information jointly (i.e. semantics/context, topology, geometry) for both the rigid bodies and the free space models in an evolvable ontology. This kind of ontology will enable fast query of any environment information to infer new knowledge in different kinds of applications. Moreover, rather than manually assigning the geometric constraints to a primitive action, they can be automatically generated referring to the information related to a primitive action to be performed.

2.4. Multi-level environment modeling and path planning

In this work, we use a multi-level approach for environment modeling and path planning proposed in [10]. The multi-level architecture allows using higher abstraction level information than the purely geometric models traditionally used, for improved path planning performances. The concepts introduced here are illustrated in a 2D environment example, made of 4 rigid bodies in a squared space (see Fig. 1 (a)) but stand similarly for 3D cases [9].

2.4.1. Multi-level environment model

The environment is made of two parts: the rigid bodies (which can be static, e.g. static obstacles, or mobile, e.g. mobile obstacles or manipulated rigid bodies), and the free space.

The *rigid bodies model* is composed of two layers. In the geometrical layer, the rigid bodies are represented using classical geometric primitives e.g. polyhedral models (vertices and edges displayed in white on Fig. 1 (a)). The semantic layer associates to each rigid body some attributes describing high-abstraction level information (e.g. shape).

The *free space model* is composed of 3 layers. The *geometrical layer* describes the geometry of the free-space as a cell decomposition based on an unbalanced tree (quadtree in 2D or octree in 3D [21]) (see Fig. 1 (b)). The *topological layer* is based on a graph called *topological graph* (see Fig. 1 (c)). This graph models places (the arcs of the graph represent the places called Pi) and borders (the nodes of the graph represent the borders called Bjk) and the topological relations between them (the border Bjk connects the two places called Pj and Pk). Each place and border is associated with a set of geometrical cells in the geometrical layer. The *semantic layer* (see Fig. 1 (d)) is made of attributes attached to the topological graph of the free space. The semantic information represented here may relate for example to the complexity of a place or border to be crossed (e.g. because of its narrowness).

2.4.2. Multi-level path planning strategy

We use a multi-level path planning strategy taking advantage of the multi-layer environment model and based on 2 steps performed consecutively:

- first, a coarse planning is performed. It finds a (minimum cost) path within the topological graph. This topological path is made of topological steps: each step is composed of a place to cross and a border to reach. The semantic layer is used to control the topological planning through the definition of the costs associated to the nodes and arcs of the graph as functions of the semantic information (notably the complexity to be crossed, see Fig. 1 (d)).
- second, a fine planning step uses the geometrical models of the rigid bodies and of the free-space to solve classical path planning queries for each topological step.

Three path planning strategies have been defined in [10]. The G (geometric) strategy involves Geometrical information only, and performs motion planning on the geometrical model of the environment. The strategy called GT (Geometry and Topology) involves Geometric and Topological information; it corresponds to the two-phases planning

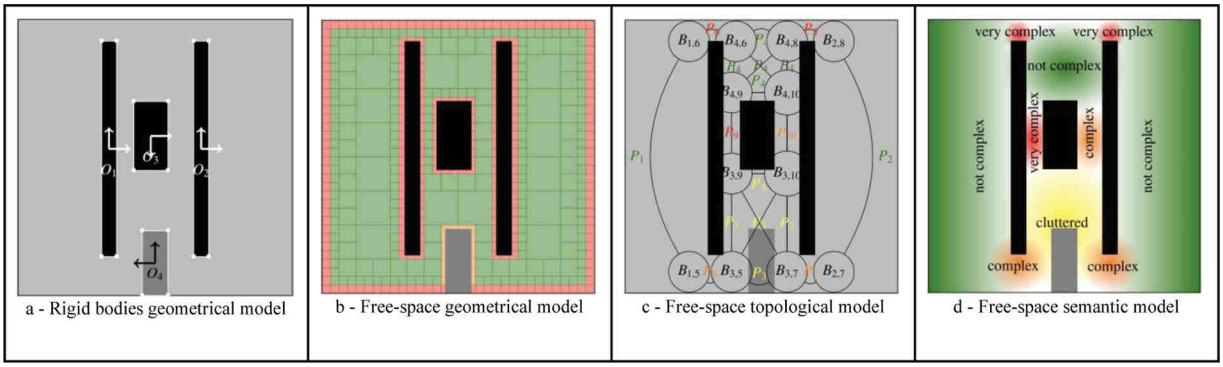

Fig. 1. Multi-layer environment model .

process presented here, but without using the semantic information, i.e. the costs in the topological graph are not depending on the semantic information, but on the distances between the borders only. The strategy called GTS (Geometry, Topology and Semantics) performs the full two-phases strategy presented here.

As we intend to validate complex tasks in potentially highly geometrically constrained environments, the geometrical path planner we use at the fine planning step and throughout the works described in this paper is a well-known probabilistic algorithm namely Rapidly-exploring Random Tree (RTT). This technique performs a probabilistic exploration of the free space to find a collision-free trajectory and iteratively extends a roadmap. Newly randomly drawn configurations are integrated locally to the roadmap under construction. The RRT algorithm is probabilistically complete but does not guarantee finite-time solutions.

3. The proposed approach

In our approach, we consider jointly task and path planning and focus on generating a specific path planning query for a primitive action using action-specific knowledge. Specifically, we are interested in generating geometric constraints from spatial constraints, where geometric constraints are specified in a path planning query and spatial constraints are defined in a specification of a primitive action. Then, the generated geometric constraints are used to guide the search in path planning. Fig. 2 shows the general architecture of the proposed ontology-based approach. It consists of two parts:

- The *Knowledge Base* includes the instances of the ontologies of task-related information (the 3D environment ontology and the action-specific knowledge ontology), involves several pre-defined SWRL Rules, and a reasoner.
  - The 3D environment ontology formalizes the heterogeneous environment data for the rigid bodies and the free space model. It considers the semantics of the contextual (types of rigid bodies, places, borders, and properties), the topological (places and borders of an environment and their connectivity), and the geometric levels.
  - The ontology of action-specific knowledge defines the world knowledge of the environment of a particular task, and the spatial relation knowledge that describes relative location between two objects, between an object and an area, and between two geometric elements.
- The *Path Planning* system manages the primitive action requests sent to the path planning manager. Any primitive action request is then instantiated in the action-specific knowledge ontology. The spatial constraints are assigned to the specification of a primitive action through an external user interface. A reasoner is invoked to perform the reasoning process of generating a path planning query for this primitive action request. As a result, the start, the goal locations, as well as the geometric constraints are generated. Then, the collision-free trajectories of this path planning query are computed.

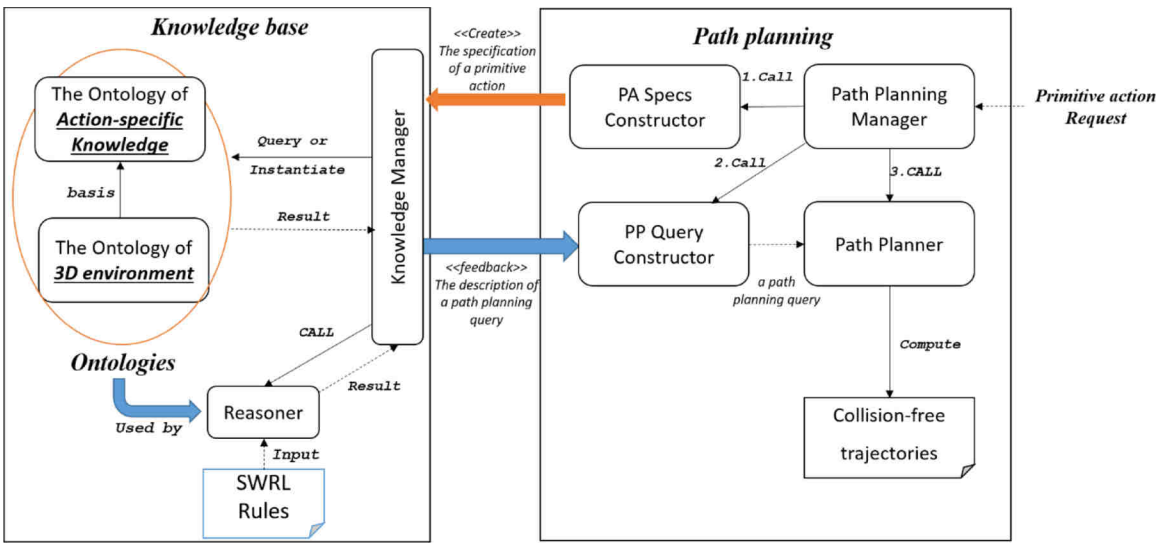

Fig. 2. The general architecture of the proposed ontology-based approach.

3.1. The 3D environment ontology (ENVOn)

The proposed 3D ENVironment ONtology (ENVOn) is an application ontology which captures the core concepts and relations of the 3D environment where a simulated manipulated task takes place for both the rigid bodies and the free space models. ENVOn reuses concepts from the multi-level environment model [10] and other existing standards and ontologies related to the modeling of a manipulation environment field such as the geometries of CAD models defined in the STEP standard. Specifically, we reused the multi-level environment model proposed in [10] and we extended its semantic level. Rather than simple textual attributes, the semantic level (called contextual level in our approach) of our proposed ontology describes in detail the contextual semantics of a 3D environment. Fig. 3 shows the general architecture of the proposed ontology with a partial view of each level with some key classes that are defined in Table 1 and their relations. Each level is formalized by an ontological module;

Table 1
Definition of some key classes in ENVOn

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)

- The geometry description module defines the geometries of the rigid bodies and the free space module. The "RB Geo Model" of rigid bodies consists of two possible geometric models; it can be either a Constructive Solid Geometry (CSG) or BRep. The class "Area" represents a bounded volume of "3D free space". The "3D Free Space Geo Model" class concerns the cell decomposition of the "3D free space" model.
- The topology description module describes the topological graph (see Section 2.4) identified in the 3D simulation environment. It also defines the connectivity of the places using the constructed topological graph (TopoGraph).
- The contextual description module provides the semantic description of rigid bodies, places, and borders. Besides, it defines container, opening, hole, and their properties (function, color, shape). In fact, this level of information heavily relies on the application (i.e. the manipulation task to be performed).

To summarize, the 3D environment ontology is a common

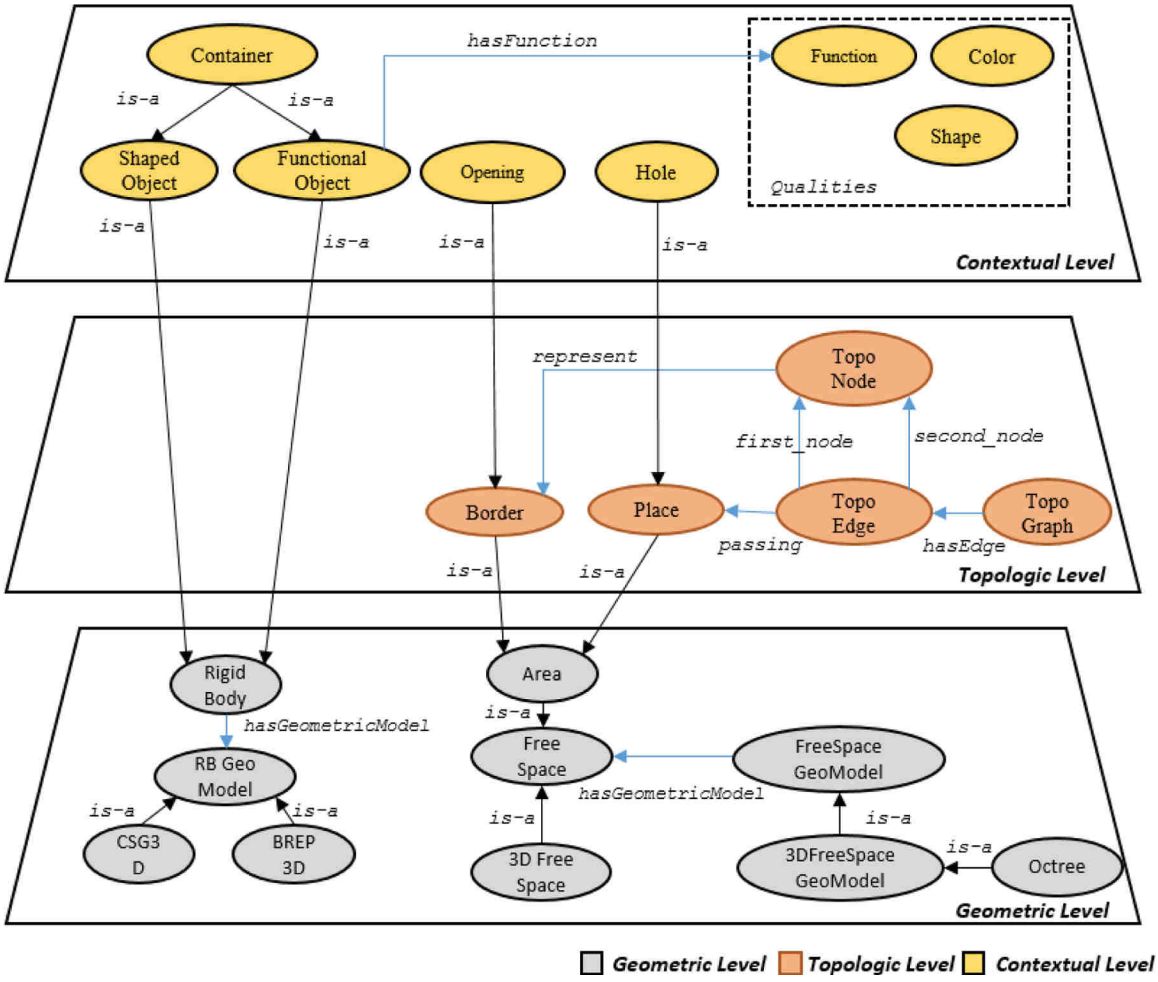

Fig. 3. The conceptual map of the 3D environment ontology.

vocabulary that can be reused by different applications of manipulation tasks. The knowledge of each level can be easily enriched, extracted, updated, and reused by other domain ontologies.

3.2. The action-specific knowledge ontology

The spatial relation knowledge is formalized in order to describe how two objects are relatively located to each other in a 2D/3D environment and also to provide specific constraints on the way that the object should be manipulated. The ontology of action-specific knowledge is based on ENVOn. The proposed ontology defines three types of relations [7] as shown in Fig. 4:

- Topological relation class describes how the boundaries and the interiors of two geometries (object or area) are related. Disjoint, Touch, Overlap, Inside and Equal are all instances of this class.
- Orientation relation class describes how two geometries (object or area) are placed relative to one another. Top, Down, Front, Back, Left, Right, and PointTo are all instances of this class.
- Geometric relation class describes how two geometric elements (e.g. line, plane) relate to each other. Coplanar, Perpendicular, Offset, Collinear, Against, Parallel are all instances of this class.

A "Spatial Constraint" is defined as a "Spatial Relation" that exists between two geometries, more specifically, between two "RigidBodies" or between a "RigidBody" and a "Place". A "Geometry Constraint" is defined as a "Spatial Relation" between two geometric elements for example "Left" (Point1, Point2), "Parallel" (Line1, Line2).

To generate a path planning query for a primitive action in a task plan, it is necessary to model a primitive action and its associated path planning query appropriately. To do so, we defined the following two classes:

- "Primitive Action Specification" class related to:
  - Primitive action identity (I) class: the kind of action that a system is executing (e.g. insert, put).
  - Manipulated Object class: the rigid body that a system manipulates.
  - Reference Geometry class: the rigid body or place to which this primitive action references.
  - Primitive action constraint specification class: the spatial constraints for the primitive action.
- "Path planning query description" class related to:
  - Manipulated object class.
  - Goal Configuration class: the configuration to reach.
  - Path planning constraint specification class: the geometric constraints in a path planning query.

3.3. The instantiation of a 3D simulation environment

Before starting a simulation, the related environment should be instantiated in the knowledge base. The objective of such an instantiation is to specify a primitive action and its associated *Spatial Constraints* and also to support the inference of *Geometric Constraints* for the corresponding path planning query. The instantiated environment data includes the contextual semantics (e.g. the top face of a table, the opening direction of a container), the topological representation of the 3D simulation environment, and the geometric information (e.g. the standard form of a *RigidBody*, the central axis of *Place*) of the *RigidBodies, Places,* and *Borders*. Instantiating the environment data is necessary so that a *Geometric Constraint* between two *Geometric Elements* can be inferred from a *Spatial Constraint* between two *Geometries* (*RigidBody* or *Place*).

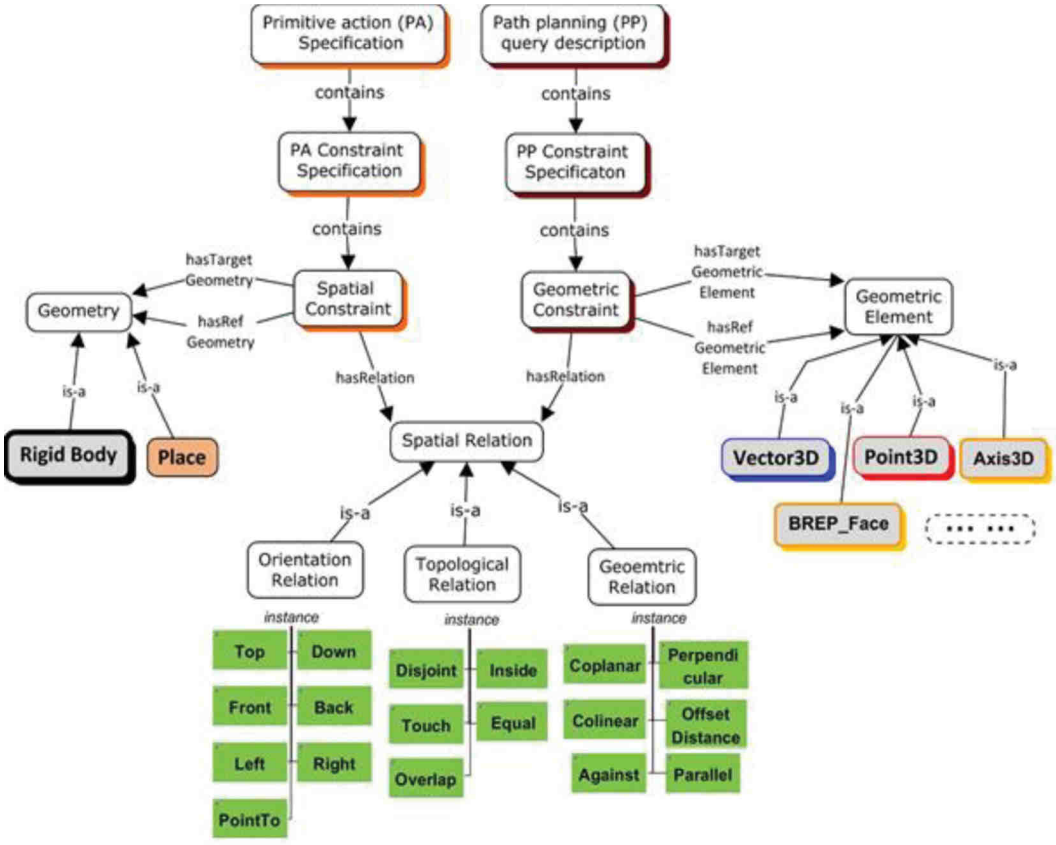

Fig. 4. The conceptual map of the action-specific knowledge ontology.

3.4. The generation of a path planning query description through an inference process

The first step of the generation process of a path planning query is to construct the primitive action specification according to the targeted primitive action to be executed. The primitive action identity, the manipulated object, and the reference geometry are automatically extracted from the primitive action to be performed. A list of spatial constraints is manually assigned by a human operator and is associated with the primitive action specification.

Then, the Path Planning Query Constructor (PPQC) takes the primitive action specification as input and generates a related path planning query description (PPQD). The manipulated object in this PPQD is directly extracted from the primitive action specification. The goal configuration is randomly sampled in the goal region. This goal configuration usually should be compliant with the geometric constraints that are inferred from the Spatial Constraints of the primitive action specification. In fact, the reasoner is invoked to generate the related geometric constraints from the spatial constraints using the predefined SWRL rules. This constraint generation process is defined as the "Spatial-Geometric Constraint Mapping". Fig. 5 presents the whole process.

3.5. Interfacing with the multi-level path planning process

The feasibility of a primitive action needs to be validated by a path planner through generating a collision-free trajectory. Taking the parameterized path planning query as input (see Fig.6), the multi-level path planning process is run to construct a collision-free geometric path to control the movement of the manipulated rigid body.

The geometric constraints inferred as explained in Section 3.4 are applied to the (probabilistic) motion planning performed at the geometric level (G) of our multi-level motion planning architecture.

Our proposed action-specific knowledge ontology may allow manipulating or inferring spatial and geometric constraints of different kinds (orientation, topological or geometric relations). In this paper, we evaluate and validate our proposed contributions with two use cases. The spatial constraints manually defined at this stage deal with orientation or geometric relations allowing to infer geometric constraints related to the alignment of reference vectors associated to the manipulated object or the goal, and to their orientations.

Such geometric constraints are used to improve control on the sorting of random configurations at the geometric motion planning level. Inside a sphere centered on the position of the goal configuration, the range of possible orientation angles around the target ones is progressively reduced as the distance to the target configuration decreases.

3.6. Ontology-based control on motion planning for a primitive task: the GO ang GTO strategies

In addition to the G, GT and GTS motion planning strategies introduced in Section 2.4.2, we propose two additional strategies. They are named GO and GTO and respectively perform the G and GTS strategies, where the random sampling of configurations at the geometrical level is done under the geometric constraints inferred as presented in Section 3.4 and applied as defined in Section 3.5.

For all these strategies, the motion planning algorithm used at the geometrical level (G) is the RRT algorithm. Other probabilistic planners may be used, like the RRT-connect algorithm [30], but the focus of our contribution here is to show how high abstraction level control improves the performances of motion planning algorithms. The improvements brought by the GO and GTO strategies would have been as significant if we had chosen other motion planning algorithms.

4. Implementation, evaluation, and validation

To develop the proposed approach, we have used the 3D simulation software *Virtools* and the ontology editor *Protégé*. In addition, we have used *FCL (Flexible Collision Library)* to check collisions and OWL API to manipulate the ontologies. The proposed system architecture consists of three main modules (see Fig. 7): First, the knowledge base module manages the task-related information in ontologies. Second, the Java GUI module provides a user interface to facilitate task-related information instantiation in ontologies. Third, the planning module generates the path planning query for a primitive action using task-related information and computes the relevant trajectories.

Specifically, the instantiation process is initialized by the construction of the multi-level environment model using *Environment Constructor*. The process consists of two different steps: First, once the multi-level environment model is constructed, the *Rigid Bodies, Places* and *Borders* of the simulated environment and their geometric models are instantiated in the knowledge base through the *Knowledge Manager*.

Secondly, thanks to an intuitive user interface, different properties of *Rigid Bodies, Places* and *Borders* can be defined and conserved in the knowledge base. In this step, when the user opens the interface, the conserved *Rigid Bodies, Places* and *Borders* and the applicable properties to them are queried from the knowledge base and then displayed. The user adds the primitive action specification by choosing any one of them

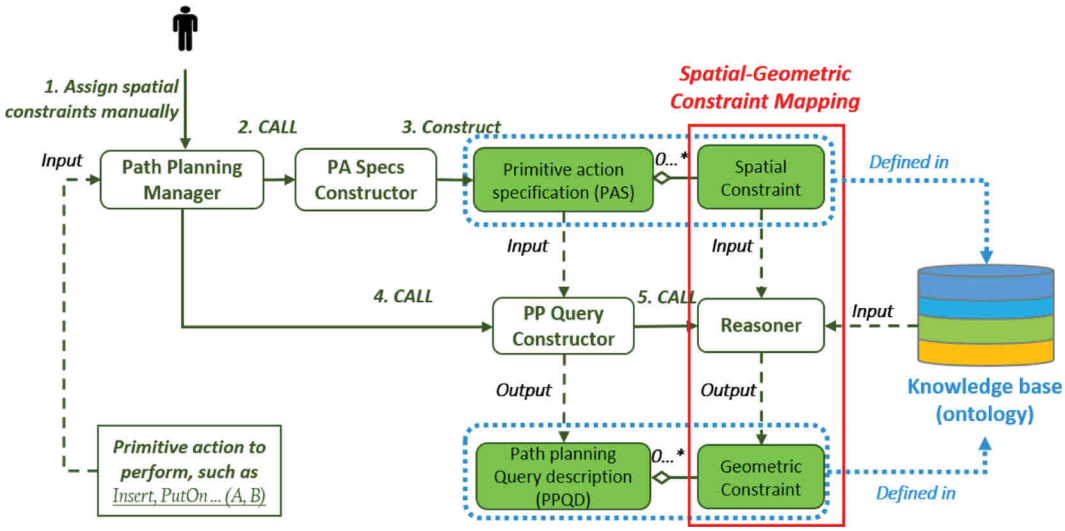

Fig. 5. The process to construct a PAS and to generate a related PPQD.

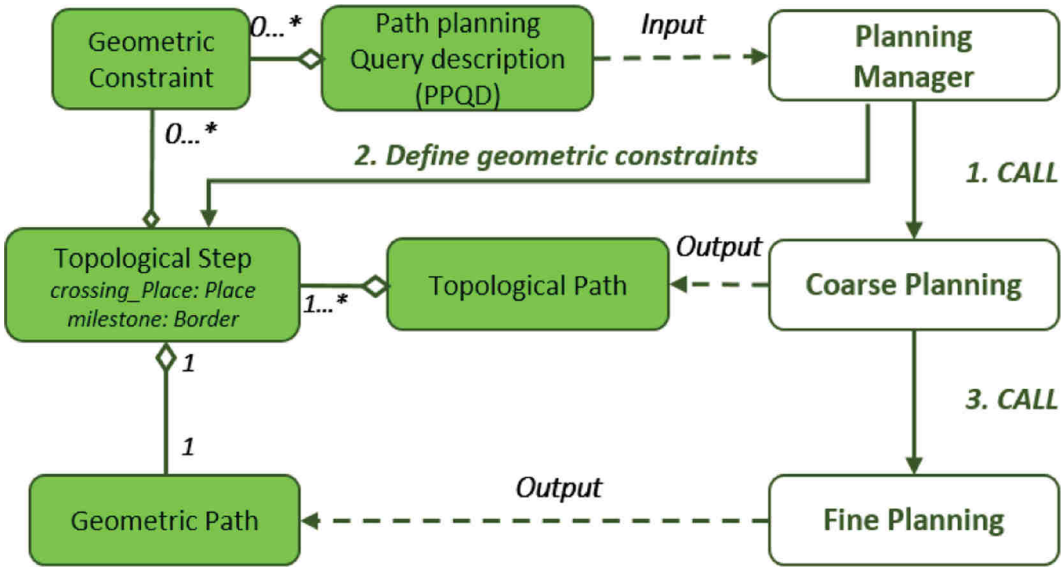

Fig. 6. Interfacing with the multi-level path planning architecture.

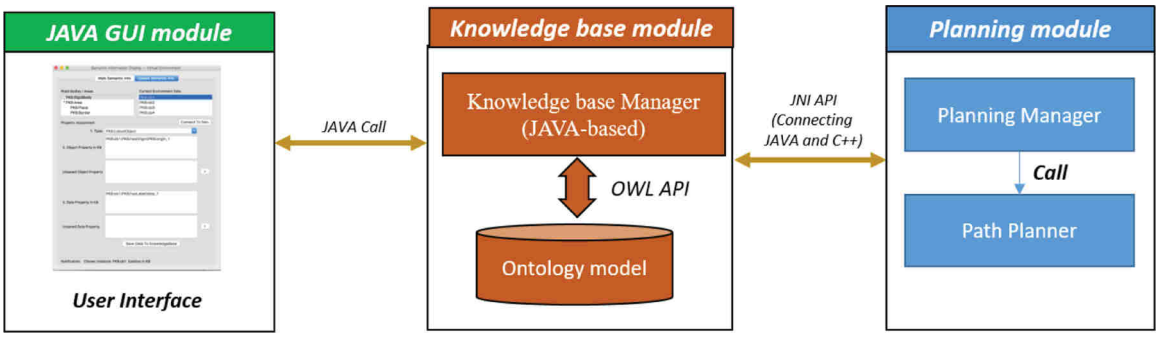

Fig. 7. The architecture of the path planning system.

(Rigid Bodies, Places, or Borders) and assigning properties to them.

The instantiation of the environment model in the knowledge base is vital for the definition of the *Path Planning Query* for a primitive action, in particular for generating the related *Geometric Constraints* for the path computation.

4.1. Context of the validation: use-cases and planning strategies

We propose two use-cases to evaluate and validate the proposed approach. The first use-case consists in inserting a pen into a penbox. The second, inspired by shapes embedding games for babies, consists in inserting objects of specific shapes into a corresponding hole of a complex object. Both use cases are highly relevant. Indeed, they correspond to very challenging scenarios for joint task and path planning as they feature the manipulation of objects under very strong geometric constraints. Furthermore, they present close similarities with many industrial manipulations, as they involve (1) standard shaped objects, representative of manufactured parts to be assembled, and (2) tasks like the manipulation in constrained places or insertion in properly shape holes, which is representative of generic tasks to be performed when (e.g.) assembling, disassembling or maintaining an industrial system. Last but not least, the case studies are not specific to any particular application, therefore generic enough to validate our approach objectively.

For each use-case, we compare five planning strategies:

- The G, GT, and GTS strategies introduced in Section 2.4.2.
- The GO and GTO strategies introduced in Section 3.6.

4.2. Validation of the proposed approach through the first use-case

The first use-case concerns inserting a pen into a narrow penbox. To interface with the multi-level path planning architecture, we constructed the multi-level environment model (see Fig. 8). The topological level of the free space model contains two places (P1, P2) and one border (B). *P1* is enriched with the complexity attribute "Free" and *P2* with "Narrow" in order to use different geometric constraints. Fig. 8 (c) shows that "Pen1" is pointed to "Penbox1" (i.e. "Vector1" is against "Vector2"). This constraint will be used in the path computation to insert "Pen1" into "Penbox1".

ENVOn is instantiated with the environment data of the pen-penbox insertion scenario (see Fig. 9). "*Pen1*" is an instance of the concept *Pen*, it has an origin (*Point3D*) "*Origin_pen_1*", a central axis (*Axis3D*) "*axis1*" and a pointing direction (*Vector3D*) "*Vector1*".

4.2.1. Validation of the generation of the path planning query for a primitive action

The performed action of the first use-case is "Pen1 should be inserted into Penbox1" along with the spatial constraint that Pen1 should point to Penbox1. The corresponding path planning query is built from the specification of this primitive action. Firstly, the goal configuration of the path planning query is obtained by random sampling in the goal place. The sampling process must respect the geometric constraints associated with the goal place. Fig. 10 (a) shows an example of a SPARQL query that finds the right place to insert Pen1. Fig. 10 (b) shows the obtained result; P2 is the goal place. Fig. 10 (c) shows the goal

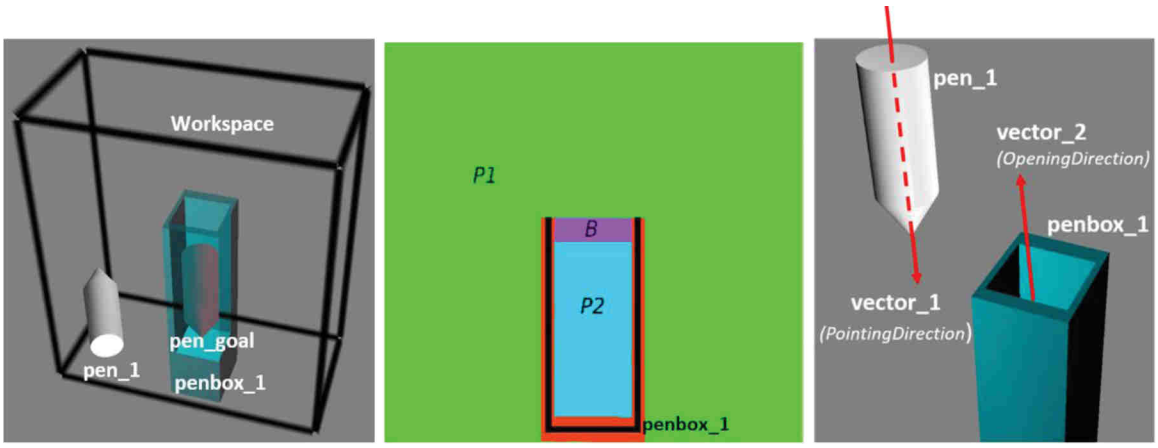

Fig. 8. Narrow pen-penbox insertion use-case.

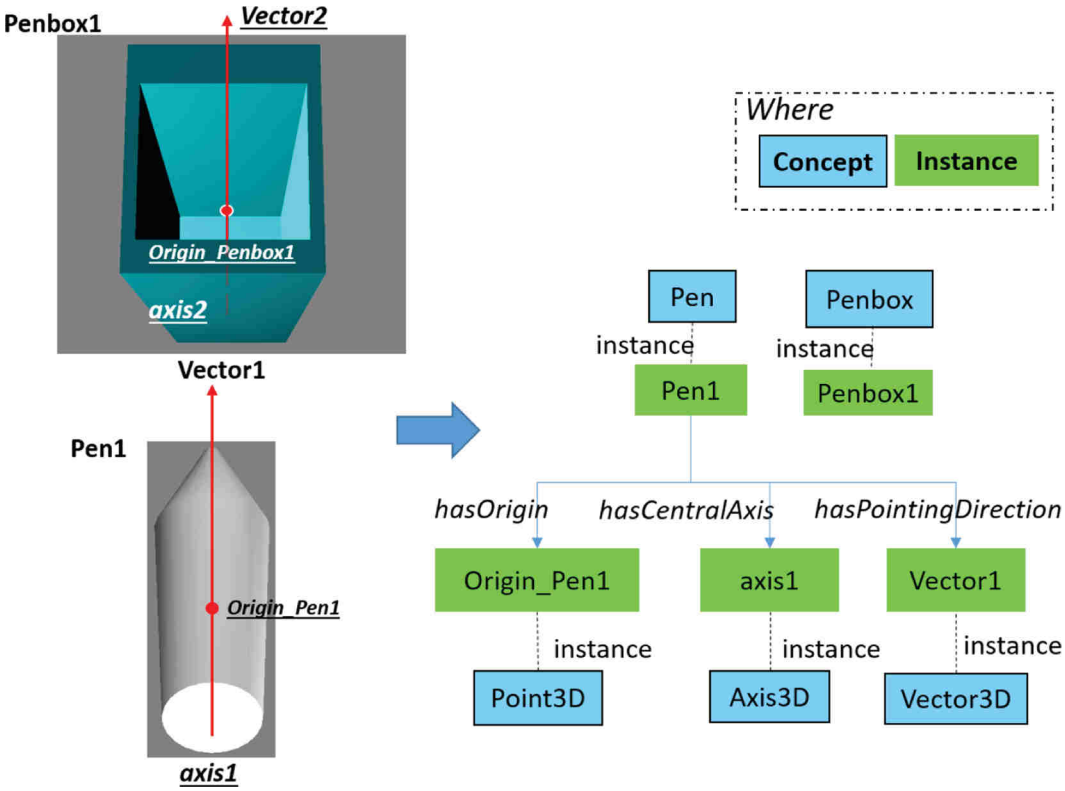

Fig. 9. Instantiation of ENVOn.

configuration of Pen1; *pen_goal*. Afterwards, the spatial constraint is mapped into the corresponding geometric constraint through the inference process presented at the end of Section 3.4: Vector1 *Against* Vector2 to control the trajectory definition when Pen1 is inserted into Penbox1, as explained at the end of Section 3.5.

4.2.2. Validation of path planning for a primitive action

In order to show the added value of using the ontology-driven two-step path planning with topological and geometric data (GTO) strategy, the different path planning strategies (G, GT, GTS, GO, and GTO strategies) presented in Section 4.1 have been tested and compared based on a parameterized path planning query generated for the primitive action "*Insert (Pen1, Penbox1)*". The obtained results are presented in Figs. 11 and 12. We conclude the following:

- The GT and the GTS strategies use almost half the computational time of the G strategy but with a bit more random samples. The result is due to the multi-level path planning process that reduces the search space of RRT to the place where a topological step crosses. Since two topological steps are involved in the narrow pen-penbox insertion use-case, the total number of random samples can be more than the one of the G strategy.
- There is no big difference between the results of the GT and the GTS strategies. Accordingly, the semantic information (i.e. complexity and geometric form) does not control the motions of how Pen1 should be inserted into P2.
- The GO strategy uses less than half the number of random samples and quarter the computational time compared to the G strategy. Such a result is caused by the inferred geometric constraints to reduce the search space of RRT.

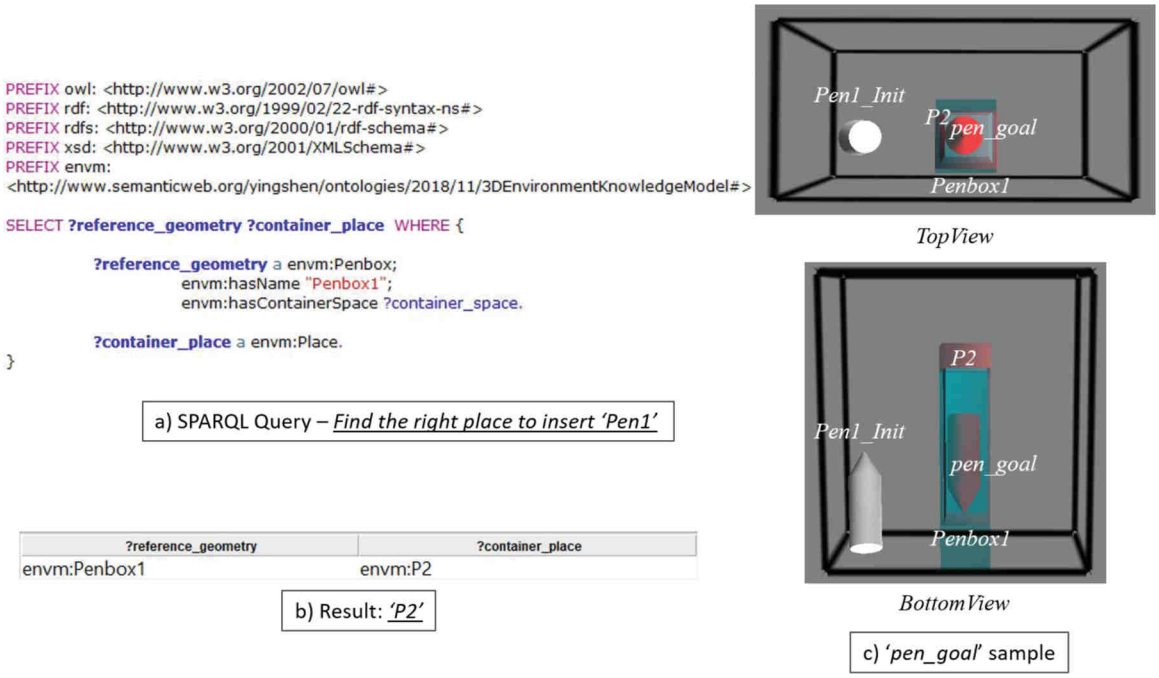

Fig. 10. The search of the goal to reach: 'pen_goal'.

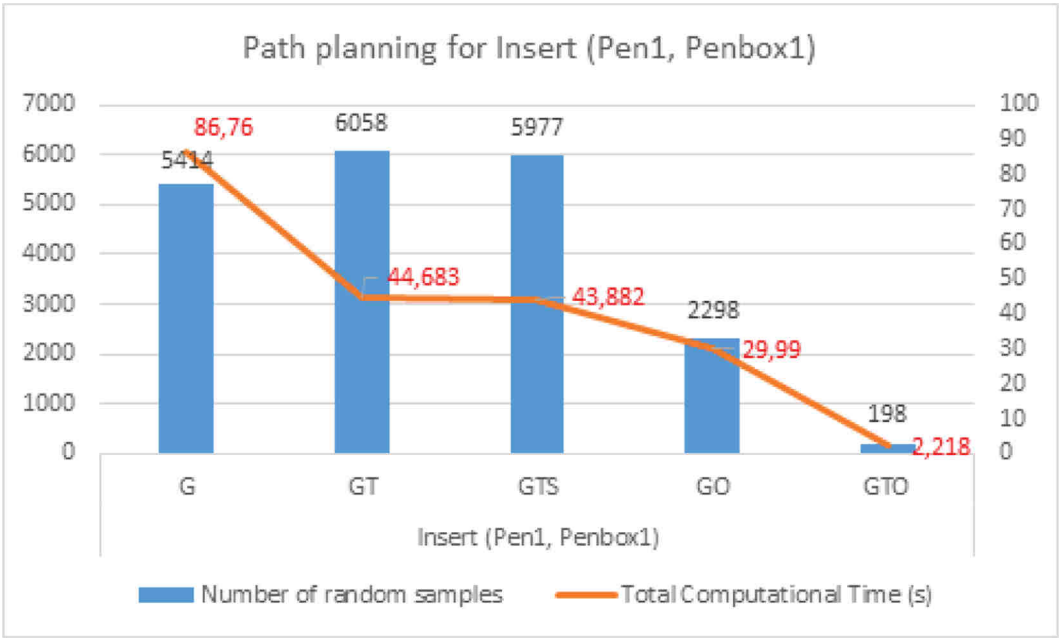

Fig. 11. Path planning results for 'Insert (Pen1, Penbox1)'.

- The result of the GTO strategy is much better than the GO strategy because the GTO strategy can determine the key milestone configuration at the border between P1 and P2. Once it is determined, the search space of RRT is reduced to the place where a topological step crosses.

As we can see in Fig. 11, our approach improves drastically the performances of the path planner used without semantic control (i.e. G strategy). The GO strategy reduces the number of samples randomly drawn by about sixty percent and the computational time by a factor of almost three. By using the GTO strategy, the number of samples is reduced by an order of magnitude of more than ninety percent, and the processing times are reduced by a factor of about forty.

4.3. *Validation of the proposed approach through the second use-case*

The second use-case concerns the "shape" attribute that makes the insertion much harder. Compared to the pen-penbox insertion use-case, this use-case requires additional study to match the shape of the hole and the manipulated object. Accordingly, the 3D environment is a cuboid workspace cluttered with five rigid bodies (O1 to O5) (see Fig. 13). O1 is fixed while 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

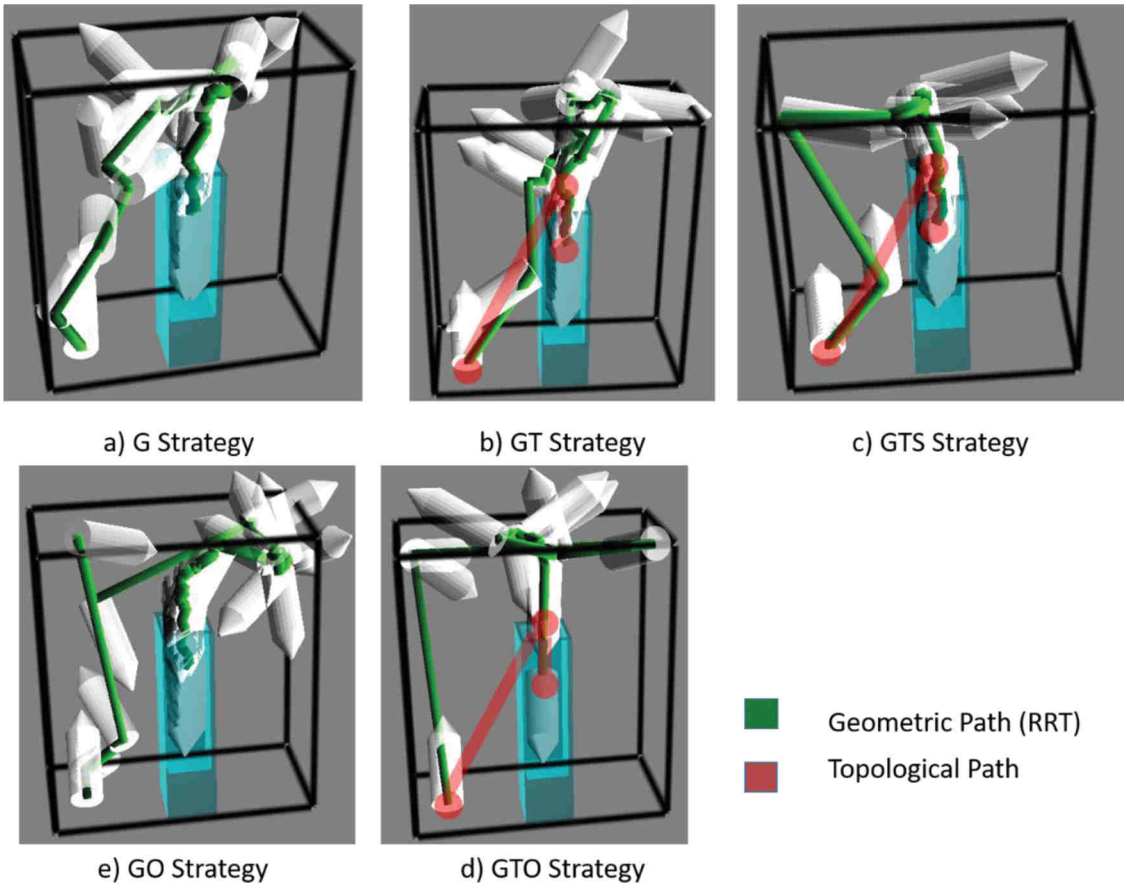

Fig. 12. The computed path using 'G, GT, GTO, GO, GTS' Strategies.

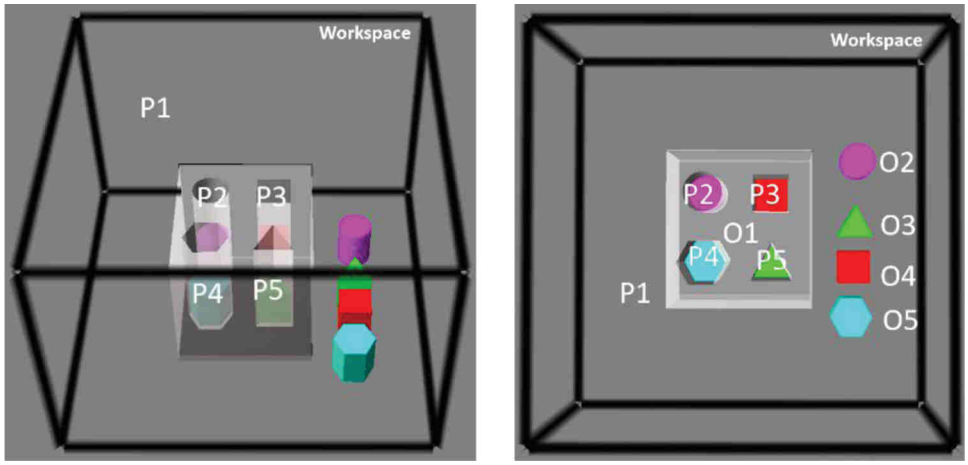

Fig. 13. Shape embedding game use-case.

respectively have the shapes *RSQ_Cylinder, RSQ_Cuboid, RSQ_PentagonPrism,* and *RSQ_Triangular Prism*. O2 to O5 respectively have the shapes *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).

4.3.1. Validation of the generation of the path planning query for a primitive action

One of the performed actions of the second use-case is to insert O3 into 01 along with the constraints that O3 and the destination hole should have the same shape (a triangular prism shape) and should be aligned. The corresponding path planning query is built from the specification of this primitive action.

First, the goal configuration of the path planning query is obtained by random sampling in the goal place. The sampling process must respect the geometric constraints associated with the goal place. Fig. 14 presents an example of the SPARQL query of finding the right place to insert O3 and the obtained result (P5 is the goal place). Fig. 14 (c) shows the goal configuration of O3 that is obtained using a random sampling within P5. Second, the spatial constraint is mapped into the corresponding geometric constraint (Vector1 *Against* Vector2 and Vector3

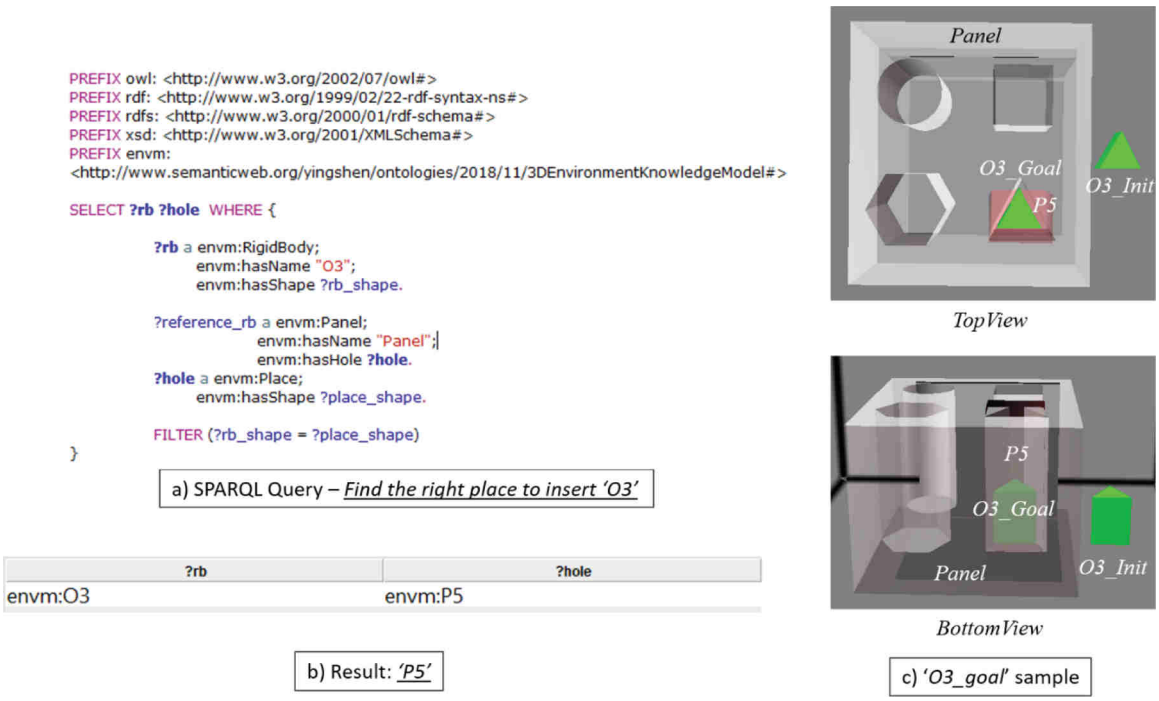

Fig. 14. The search of the goal to reach: 'O3_Goal'.

SameDirection Vector4) (see Fig. 15). It is done through the inference process using SWRL Rule 1: O3 ShapeAligned P5 (see Fig. 16).

4.3.2. *Validation of path planning for a primitive action*

To demonstrate how effective the task level information can contribute to path planning of a primitive action, the different path planning strategies (G, GT, GTS, GO, GTO) are tested and compared for

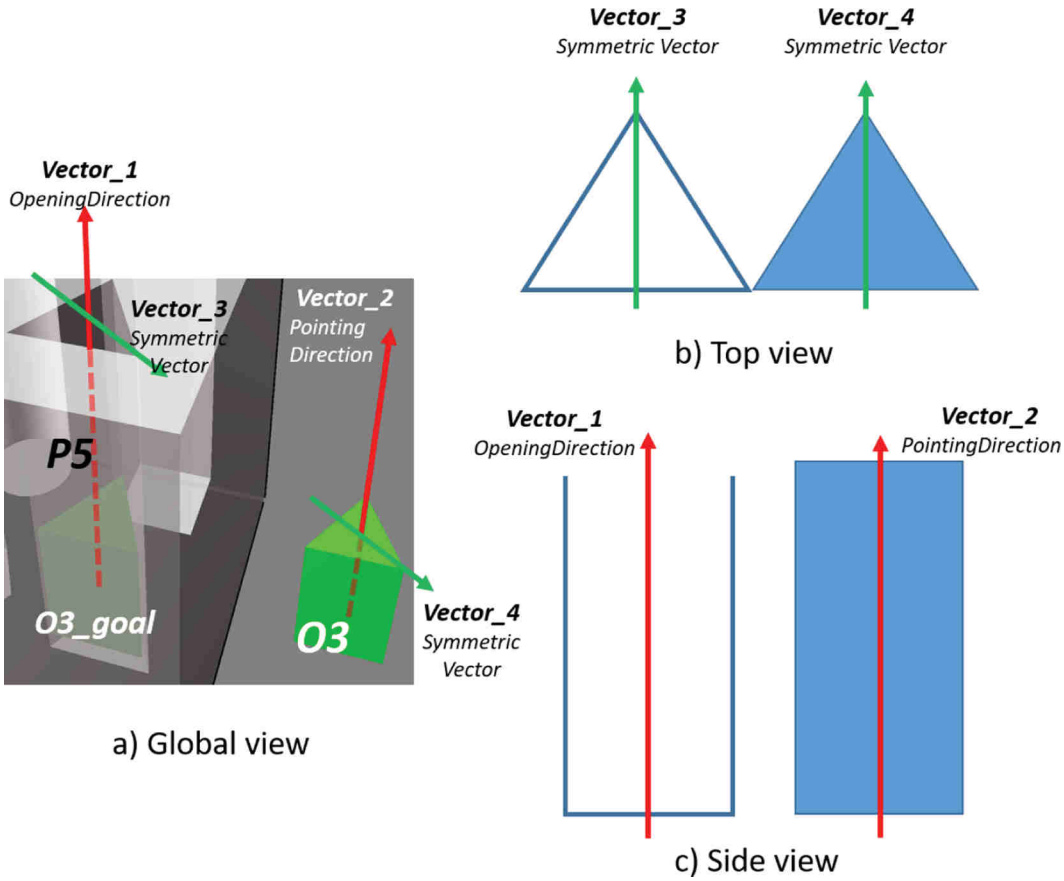

Fig. 15. The shape alignment for O3 and P5.

A SWRL Rule for spatial-geometric constraint mapping

```
1. IF {
2.  a instance of Place and a has3DShape RSQTriangular_Prism
3.  b instance of RigidBody and b has3DShape RSQTriangular_Prism
4.  vector_1, vector_2, vector_3, vector_4 instance of Vector and
5.  a hasPointingDirection vector_1 and a hasSymmetricVector vector_3 and
6.  b hasOpeningDirection  vector_2 and b hasSymmetricVector vector_4 and
7.  sconstraint instance of SpatialConstraint and
8.     sconstraint  hasRelation "ShapeAligned" and
9.     sconstraint  hasRefGeometry "b" and
10.    sconstraint  hasTargetGeometry "a" and
11. gconstraint1, gconstraint2 instance of GeometricConstraint and
12.    gconstraint1 hasRelatedSC "sconstraint"
13.    gconstraint2 hasRelatedSC "sconstraint"
14. }
15. THEN {
16.  gconstraint1 hasRefGeoElement "vector_2" and
17.  gconstraint1 hasTargetGeoElement "vector_1" and
18.  gconstraint1 hasRelation "Against"
19.  gconstraint2 hasRefGeoElement "vector_3" and
20.  gconstraint2 hasTargetGeoElement "vector_4" and
21.  gconstraint2 hasRelation "SameDirection"
22. }
```

Fig. 16. SWRL Rule 1: O3 ShapeAligned P5.

the manipulation of objects O2, O3, O4, and O5. The obtained results are presented in Fig. 17 (number of random samples) and Fig. 18 (total computational time). As an example, the different computed trajectories for O3 are shown in Fig. 19. We conclude the following:

- For all cases (O2 into P2, O3 into P5, O4 into P3, and O5 into P4), the G, GT, and GTS strategies cannot find collision-free trajectories even by reaching the maximum number of allowed random samples and take a huge amount of computational time (more than 5 hours). The reason is that it is difficult for RRT with uniform random sampling to find collision-free samples in narrow passages.
- The GO strategy can find a collision-free trajectory with a much less number of random samples needed. The reason is that the geometric constraints restrict the allowed pose of random samples so that the search space of random samples is then reduced. The path planning of inserting O5 into P4 uses less computational time because the margin between the 3D volumes of O5 and P4 is larger than the other cases (O2 and P2, O3 and P5, O4 and P3).
- The GO strategy still takes more than 50 minutes to find collision-free trajectories while the GTO strategy takes less than 19 seconds. The reason is that the GTO strategy can find the key collision-free geometric configurations at the borders between P1 and P2, P3, P4, P5. These geometric configurations provide guidance that leads the manipulated object into the holes.

Again here, the performances (number of random samples, computation times) are drastically improved with respect to the performances of the RRT algorithm used without semantic control.

5. Conclusion and future work

In this work, we presented an ontology-based approach that uses task level information to generate a path planning query for a primitive action of a task plan. The proposed 3D environment ontology (ENVOn) can give fast access to any geometric details of a given rigid body or a given place. It can also enhance the reasoning at the task level (e.g. to check whether a manipulated object can be inserted into a target place). ENVOn is an evolvable ontology that can be easily updated according to the task to be performed. To validate the proposed approach, two different use-cases with increasingly geometrically constrained environments are presented. First, we exploited the proposed ontology to automatically generate a path planning query associated with a target primitive action of a task plan. Through the reasoning process and the instantiation of the primitive actions in the ontology, we can infer the

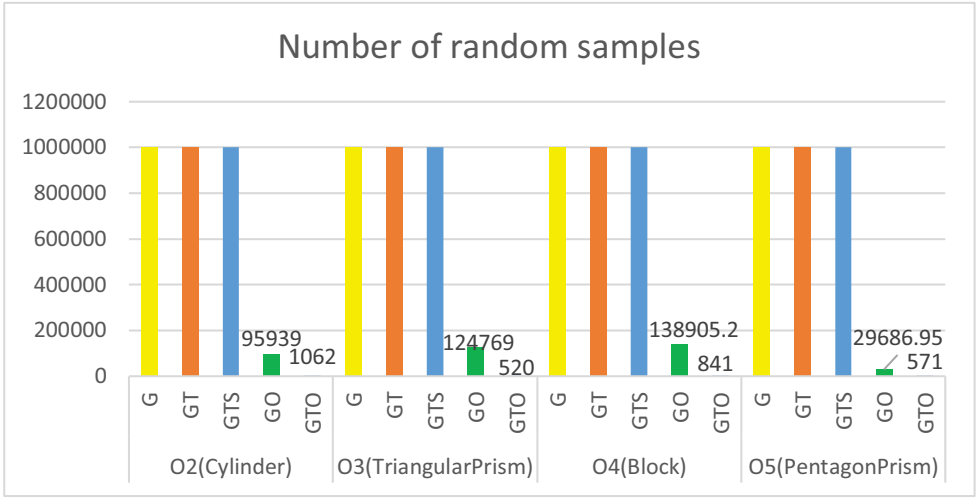

Fig. 17. Path planning results (G, GT, GTS, GO, GTO): Number of random samples used.

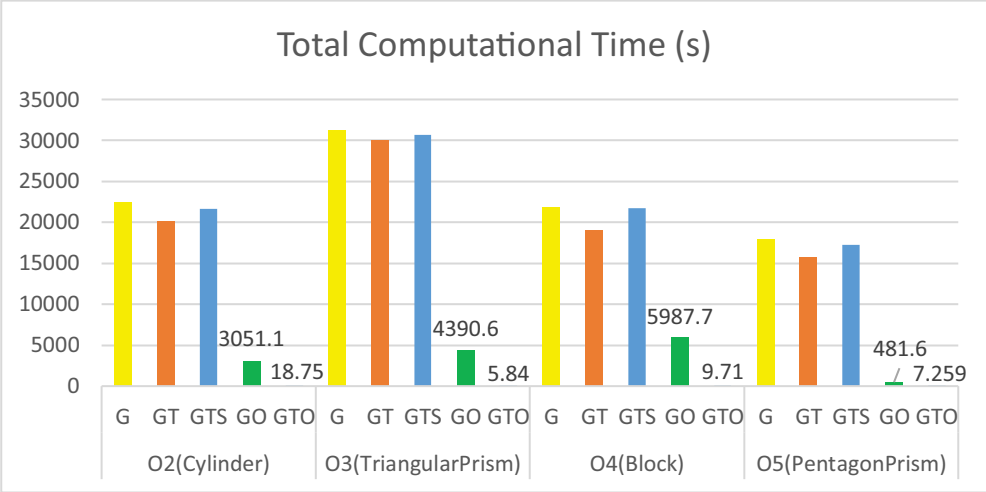

Fig. 18. Path planning results (G, GT, GTS, GO, GTO): total computational time.

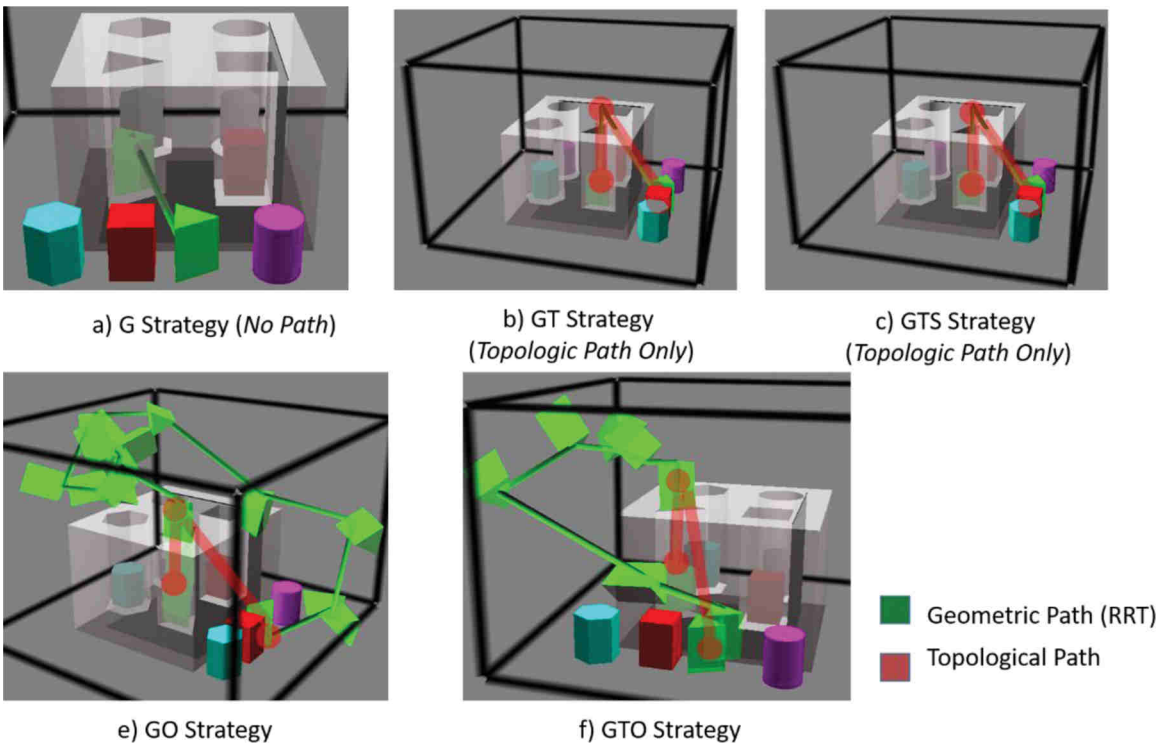

Fig. 19. The computed path using G, GT, GTO, GO, and GTS strategies.

start and the goal configurations, as well as task-related geometric constraints. Then, the results of the different path planning strategies are compared. The obtained results demonstrate that using task-related information allows better control on the RRT path planning algorithm involved to check the motion feasibility for the primitive actions of a task plan, leading to lower computational time and more relevant trajectories for primitive actions.

In the next steps of this work, we intend to explore the use of a wider range of motion planners at the geometric level of our multi-level motion planning architecture, and also to validate our action-specific knowledge ontology by defining and inferring spatial and geometric constraints of more various kinds. We will also further investigate ontology-based joint task and path planning by taking into account full task plans (and not the primitive tasks only), in an iterative approach where the task plan would allow to generate all associated path planning queries, and the path planning queries results would be used to question and help updating the task plan if non feasible or non-relevant. Studying the nature of the data exchanged between the task and path planning levels will be a challenging issue; for example, information provided by the path planning level to the task planning level should be richer than a mere binary information on the feasibility of motion for the considered task plan. An ontology-based model of the exchanged information should improve the cooperation between both planning levels involved.

CRediT authorship contribution statement

Yingshen Zhao: Conceptualization, Data curation, Investigation, Methodology, Visualization, Writing – original draft, Writing – review & editing. **Philippe Fillatreau:** Conceptualization, Data curation, Investigation, Methodology, Writing – original draft, Writing – review &

editing. **Linda Elmhadhbi:** Visualization, Writing – original draft, Writing – review & editing. **Mohamed Hedi Karray:** Conceptualization, Methodology, Supervision, Validation, Writing – original draft, Writing – review & editing. **Bernard Archimede:** Supervision, Validation.

Declaration of Competing Interest

None.

References

[1] A. Akbari, Muhayyuddin, J Rosell, Knowledge-oriented task and motion planning for multiple mobile robots, J. Exp. Theor. Artif. Intell. 31 (1) (2019) 137–162.

[2] A. Akbari, J. Rosell, Ontological physics-based motion planning for manipulation, in: 2015 IEEE 20th Conference on Emerging Technologies & Factory Automation (ETFA), IEEE, 2015, pp. 1–7.

[3] A. Aztiria, J.C. Augusto, A. Izaguirre, Spatial and temporal aspects for pattern representation and discovery in intelligent environments, in: Workshop on Spatial and Temporal Reasoning at 18th European Conference on Artificial Intelligence, 2008.

[4] Bidot, J., Karlsson, L., Lagriffoul, F., & Saffiotti, A. (2013). Geometric backtracking for combined task and path planning in robotic systems.

[5] D. Beßler, M. Pomarlan, M. Beetz, Owl-enabled assembly planning for robotic agents, in: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, 2018, pp. 1684–1692.

[6] W. Böhm, G. Farin, J. Kahmann, A survey of curve and surface methods in CAGD, Comput. Aided Geometr. Des. 1 (1) (1984) 1–60.

[7] A. Borrmann, J. Hyvärinen, E. Rank, Spatial constraints in collaborative design processes, in: Proceedings of the International Conference on Intelligent Computing in Engineering (ICE09), Berlin, Germany, 2009, pp. 1–8.

[8] Botsch, M., Pauly, M., Kobbelt, L., Alliez, P., Lévy, B., Bischoff, S., & Röossl, C. (2007). Geometric modeling based on polygonal meshes.

[9] S. Cailhol, P. Fillatreau, J.Y. Fourquet, Y. Zhao, A hierarchic approach for path planning in virtual reality, Int. J. Interact. Des. Manuf. 9 (4) (2015) 291–302.

[10] S. Cailhol, P. Fillatreau, Y. Zhao, J.Y. Fourquet, Multi-layer path planning control for the simulation of manipulation tasks: Involving semantics and topology, Rob. Comput. Integr. Manuf. 57 (2019) 17–28.

[11] O. Caldiran, K. Haspalamutgil, A. Ok, C. Palaz, E. Erdem, V. Patoglu, Bridging the gap between high-level reasoning and low-level control, in: International Conference on Logic Programming and Nonmonotonic Reasoning, Springer, Berlin, Heidelberg, 2009, pp. 342–354.

[12] S. Cambon, R. Alami, F. Gravot, A hybrid approach to intricate motion, manipulation and task planning, Int. J. Robot. Res. 28 (1) (2009) 104–126.

[13] B.A. Dang-Vu, O. Porges, M.A. Roa, Interpreting manipulation actions: From language to execution, in: Robot 2015: Second Iberian Robotics Conference, Springer, 2016, pp. 175–187. , Cham.

[14] De Silva, L., Pandey, A. K., Gharbi, M., & Alami, R. (2013). Towards combining HTN planning and geometric task planning. arXiv preprint arXiv:1307.1482.

[15] R. Dearden, C. Burbridge, An approach for efficient planning of robotic manipulation tasks, in: Twenty-Third International Conference on Automated Planning and Scheduling, 2013.

[16] M. Diab, A. Akbari, J. Rosell, An ontology framework for physics-based manipulation planning, in: Iberian Robotics conference, Springer, 2017, pp. 452–464. , Cham.

[17] M. Diab, A. Akbari, M. Ud Din, J. Rosell, Pmk—a knowledge processing framework for autonomous robotics perception and manipulation, Sensors 19 (5) (2019) 1166.

[18] C. Dornhege, M. Gissler, M. Teschner, B. Nebel, Integrating symbolic and geometric planning for mobile manipulation, in: 2009 IEEE International Workshop on Safety, Security & Rescue Robotics (SSRR 2009), IEEE, 2009, pp. 1–6.

[19] E. Erdem, K. Haspalamutgil, C. Palaz, V. Patoglu, T. Uras, Combining high-level causal reasoning with low-level geometric reasoning and motion planning for robotic manipulation, in: 2011 IEEE International Conference on Robotics and Automation, IEEE, 2011, pp. 4575–4581.

[20] P. Eyerich, T. Keller, B. Nebel, Combining action and motion planning via semantic attachments, in: Proc. of Workshop on Combining Action and Motion Planning at ICAPS, 2010.

[21] R.B. Fisher, T.P. Breckon, K. Dawson-Howe, A. Fitzgibbon, C. Robertson, E. Trucco, C.K. Williams, Dictionary of Computer Vision and Image Processing, John Wiley & Sons, 2013.

[22] C. Galindo, J.A. Fernández-Madrigal, J. González, A. Saffiotti, Robot task planning using semantic maps, Rob. Autom. Syst. 56 (11) (2008) 955–966.

[23] C.R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L.P. Kaelbling, T. Lozano-Pérez, Integrated task and motion planning, Ann. Rev. Control Robot. Autonom. Syst. 4 (2021) 265–293.

[24] R. Gayathri, V. Uma, Ontology based knowledge representation technique, domain modeling languages and planners for robotic path planning: a survey, ICT Express 4 (2) (2018) 69–74.

[25] M. Ghallab, D. Nau, P Traverso, Automated Planning: Theory and Practice, Elsevier - Morgan Kauffman, 2004.

[26] J. Guitton, J.L. Farges, Taking into account geometric constraints for task-oriented motion planning, in: Proc. Bridging the gap Between Task and Motion Planning, BTAMP 9, 2009, pp. 26–33.

[27] C.M. Hoffmann, Geometric and Solid Modeling, 1989.

[28] L. Kaelbling, T. Lozano-Perez, Hierarchical task and motion planning inthe now, in: IEEE International Conference on Robotics and Automation. An chorage, Alaska: Workshop on Mobile Manipulation, 2010.

[29] Z. Kootbally, C. Schlenoff, C. Lawler, T. Kramer, S.K. Gupta, Towards robust assembly with knowledge representation for the planning domain definition language (pddl), Rob. Comput. Integr. Manuf. 33 (2015) 42–55.

[30] J.J. Kuffner, S.M. LaValle, RRT-connect: An efficient approach to single-query path planning, in: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, 2000 vol.2, 2000, pp. 995–1001, https://doi.org/10.1109/ROBOT.2000.844730.

[31] F. Lagriffoul, D. Dimitrov, J. Bidot, A. Saffiotti, L. Karlsson, Efficiently combining task and motion planning using geometric constraints, Int. J. Robot. Res. 33 (14) (2014) 1726–1747.

[32] J.C. Latombe, Robot Motion Planning, 124, Springer Science & Business Media, 2012.

[33] S. LaValle, M.M.L. Steven, Rapidly-Exploring Random Trees: A New Tool for Path Planning, Iowa State University, 1998 technical report.

[34] T. Lozano-Perez, Spatial planning: A configuration space approach. Autonomous Robot Vehicles, Springer, New York, NY, 1990, pp. 259–271.

[35] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, P. Abbeel, Combined task and motion planning through an extensible planner-independent interface layer, in: International Conference on Robotics and Automation (ICRA), IEEE, 2014, pp. 639–646.

[36] A. Nüchter, J. Hertzberg, Towards semantic maps for mobile robots, Rob. Autom. Syst. 56 (11) (2008) 915–926.

[37] ... A. Olivares-Alarcos, D. Beßler, A. Khamis, P. Goncalves, M.K. Habib, J. Bermejo-Alonso, H. Li, A review and comparison of ontology-based approaches to robot autonomy Knowl. Eng. Rev. (2019) 34.

[38] A. Perzylo, N. Somani, M. Rickert, A. Knoll, An ontology for CAD data and geometric constraints as a link between product models and semantic robot task descriptions, in: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2015, pp. 4197–4203.

[39] M.J. Pratt, Introduction to ISO 10303—the STEP standard for product data exchange, J. Comput. Inf. Sci. Eng. 1 (1) (2001) 102–103.

[40] A. Pronobis, Semantic Mapping with Mobile Robots, KTH Royal Institute of Technology, 2011. Doctoral Dissertation.

[41] A.G. Requicha, Representations for rigid solids: theory, methods, and systems, ACM Comput. Surv. 12 (1980).

[42] R.B. Rusu, Semantic 3d object maps for everyday manipulation in human living environments, KI-Künstliche Intell. 24 (4) (2010) 345–348.

[43] A. Rodriguez, L. Basanez, E. Celaya, A relational positioning methodology for robot task specification and execution, IEEE Trans. Rob. 24 (3) (2008) 600–611.

[44] T. Siméon, J. Cortés, A. Sahbani, J.P. Laumond, A general manipulation task planner. Algorithmic Foundations of Robotics V, Springer, Berlin, Heidelberg, 2004, pp. 311–327.

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

[46] M. Tenorth, G. Bartels, M. Beetz, Knowledge-based specification of robot motions. ECAI 2014, IOS Press, 2014, pp. 873–878.

[47] M. Tenorth, M. Beetz, KnowRob—knowledge processing for autonomous personal robots, in: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2009, pp. 4261–4266.

[48] S.G. Tzafestas, Introduction to mobile robot control, Elsevier, 2013.

[49] E. Vassev, M. Hinchey, Knowledge representation for cognitive robotic systems, in: 2012 IEEE 15th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing Workshops, IEEE, 2012, pp. 156–163.

[50] J. Wolfe, B. Marthi, S. Russell, Combined task and motion planning for mobile manipulation, in: Twentieth International Conference on Automated Planning and Scheduling, 2010.

[51] F. Zacharias, C. Borst, Knowledge representations for high-level and low-level planning, in: Proceedings of the scheduling and planning applications workshop at the International Conference on Automated Planning and Scheduling, 2011.

[52] Y. Zhao, P. Fillatreau, M.H. Karray, B. Archimède, An ontology-based approach towards coupling task and path planning for the simulation of manipulation tasks, in: 2018 IEEE/ACS 15th International Conference on Computer Systems and Applications (AICCSA), IEEE, 2018, pp. 1–8.

[53] F. Basile, F. Caccavale, P. Chiacchio, J. Coppola, C. Curatella, Task-oriented motion planning for multi-arm robotic systems, Rob. Comput. Integr. Manuf. 28 (5) (2012) 569–582, https://doi.org/10.1016/j.rcim.2012.02.007.