Grounding Semantic Maps in Spatial Databases

Henning Deeken[a], Thomas Wiemann[a], Joachim Hertzberg[a,b]

[a]*Knowledge-Based Systems Group, Osnabrüeck University, Wachsbleiche 27, Osnabrüeck, Germany*
[b]*DFKI Robotics Innovation Center, Osnabrück Branch, Albert-Einstein-Strae 1, Osnabrüeck, Germany*

Abstract

Semantic maps add to classic robot maps spatially grounded object instances anchored in a suitable way for knowledge representation and reasoning. They enable a robot to solve reasoning problems of geometrical, topological, ontological and logical nature in addition to localization and path planning. Recent literature on semantic mapping lacks effective and efficient approaches for grounding qualitative spatial relations through analysis of the quantitative geometric data of the mapped entities. Yet, such qualitative relations are essential to perform spatial and ontological reasoning about objects in the robot's surroundings.

This article contributes a framework for semantic map representation, called SEMAP, to overcome this missing aspect. It is able to manage full 3D maps with geometric object models and the corresponding semantic annotations as well as their relative spatial relations. For that, spatial database technology is used to solve the representational and querying problems efficiently. This article describes the extensions necessary to make a spatial database suitable for robotic applications. Especially, we add 3D spatial operators and a tree of transformations to represent relative position information. We evaluate the implemented capabilities and present real life use cases of SEMAP in different application domains.

Keywords: Semantic Mapping, Spatial Analysis, Knowledge Representation

Email addresses: `henning.deeken@uni-osnabrueck.de` (Henning Deeken), `thomas.wiemann@uni-osnabrueck.de` (Thomas Wiemann), `joachim.hertzberg@uni-osnabrueck.de`, `joachim.hertzberg@dfki.de` (Joachim Hertzberg)

1. Introduction

A semantic map for a mobile robot has to combine semantic, topological and geometric information in a compact representation. These different types of information are required to solve relevant problems like localization, path planning, 3D trajectory planning, task execution, object search, and more. Hence, semantic maps have to evolve from specially tailored task-specific representations towards multi-purpose environment models that can be re-used in different applications and updated dynamically. Such generalized models should be able to fuse information from different data layers via a query interface that allows to extract task-specific environment data on-demand.

Current approaches in semantic mapping already exhibit features of more generalized environment models. There has been significant progress in describing the semantics of environments using ontological approaches to model a-priori background knowledge and to capture facts about an environment's current state. Similarly, large-scale spatial mapping, scene segmentation, and object recognition are well understood and can be used to gather spatio-semantic data of real-world environments. The study of the anchoring problem [1] has lead to effective strategies to derive environment knowledge from sensor data and to track entities and their features over time. To that end, it is crucial to link semantic knowledge with geometric data and perform data analysis across both domains dynamically with the acquisition of updated information. However, the representational frameworks underlying semantic maps are still unable to ground spatial relations between entities. If grounding spatial relations is addressed, it is usually done during semantic map building. Appropriate tools on a representational level are rarely seen, although the benefit of spatial analysis for enriching semantic knowledge – especially for anchoring physical objects in large-scale semantic maps – is obvious.

This article presents how to derive and manage qualitative spatial relations between objects from quantitative geometric environment data captured by some kind of mapping approach. It shows how to realize efficient spatio-semantic querying on semantic maps by integrating a spatial database into a semantic mapping framework. The close integration of a spatial database provides a dedicated storage and processing module for the spatial environment data as a suitable complement to a classical knowledge-based system. By correctly anchoring spatial records to their respective semantic counterparts, the database's spatial operators provide the ability to derive qualitative information about the spatial relations between stored entities that is otherwise covert. This adds an essential feature to semantic map representations, since grounding spatial relations uncovers important information about the robot's environment. In our approach, the current semantic world model stored in a dedicated knowledge base can be updated accordingly whenever an object is inserted or modified in the semantic map. It also allows to query for environment data on demand using spatial and semantic constraints simultaneously, which allows to answer typical questions about the environment, as presented in Figure 1.

We have cast this approach to combine semantic and spatial data into the *Semantic Environment Mapping Framework* (SEMAP). In this paper, we describe the basic concepts of SEMAP's architecture, with special focus on the integration of the spatial database into the semantic mapping framework. We discuss the extensions added to an existing geometric database system that are necessary to achieve the desired functionality. We present and evaluate the new features of this semantic mapping framework that arise from the novel combination of the geometric database with a classical knowledge-based system, especially the feature of grounding qualitative spatial relations through the quantitative analysis of spatial data. We show that the presented approach generalizes well into different application domains by presenting real world examples of applying SEMAP.

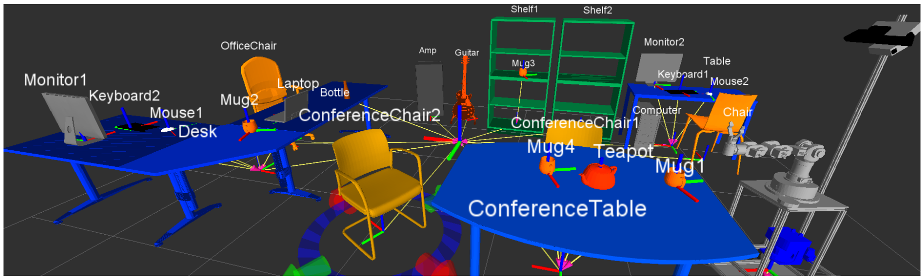


Figure 1: Artificial rendering of an office environment modeled in SEMAP. The labels denote object instances that are present in the knowledge base at their locations in 3D space and in relation to other objects, as represented in the semantic map.

- **Q1** Which objects are in this `Room`?
- **Q2** How many `Chair`s exist?
- **Q3** Where is `Mug2`?
- **Q4** Which `Mug` is closest to the robot?
- **Q5** Is there a `Computer` in this `Room`?
- **Q6** Is the `Monitor1` on the `Desk`?

2. Related Work

Over the last decade, the discipline of semantic mapping has become increasingly popular and successful. A recent survey by Kostavelis and Gasteratos [2] reviewed more than 120 different approaches. It summarizes the significant progress made on a broad range of mapping approaches and applications for semantic maps, including task planning [3], localization [4, 5], navigation [6, 7] and human-robot-interaction [8].

This review also revealed a significant heterogeneity in the processes of semantic map building, as well as in the underlying semantic map representations, because access to spatio-semantic environment data is beneficial in a multitude of applications. But the level of detail or selection of appropriate data types and information sources varies significantly, depending on the application. Therefore different semantic maps use different underlying spatial representations and semantic annotations.

Bastianelli et al., for example, presented a hybrid semantic map consisting of annotated 2D occupancy grids, whose labels were given by a human instructor, and topological graphs [8]. It was used for topological navigation, object search and object manipulation. Nüchter and Hertzberg demonstrated how 3D point clouds can be automatically segmented into categories like walls, floor and ceiling [9]. Pronobis and Jensfelt presented a vision-based system that allows to identify objects and rooms by analyzing features on position-tagged images and the geometric attributes, like area and shape, of occupancy grid maps [10].

This heterogeneity is also reflected in the definition of semantics maps, which either intentionally make no particular assumption about the mapping process or the underlying representations [9, 11] or rely on the concept of hybrid maps [12, 13]. Yet, a common agreement is that semantic maps have to be paired with formal knowledge representations and reasoning, to unfold their full potential. Recent literature provides several examples of how knowledge base components can be beneficial in semantic mapping [14, 15], for reasoning about the environment. These approaches usually use ontological and graph-based knowledge representations, based on description logics [16].

One example for such a system is *KnowRob* [17], which combines a knowledge representation in the Web Ontology Language (OWL) and Prolog-based reasoning with an interface to the robot's control architecture. The goal of KnowRob is to provide a system that is fully integrated with the robot to generate new knowledge from sensor perception and effectively guide the robot's behavior through semantic inference. In the context of semantic mapping it has been used to answer queries about a semantic object map [18]. It has been used in various projects. One is *RoboHow* that explored possibilities to use the World Wide Web as resource to find instructions for solving everyday manipulation tasks [19]. Another example is *RoboSherlock*, which defines a generic interface for perception algorithms and a knowledge base to plan which perception modules to use and to consistently feed perceptions into the knowledge base [20]. *OpenEASE* aims at creating a knowledge base for manipulation episodes that can be queried by multiple robots to share their experiences in order to learn manipulation tasks and to improve their performance [21].

It is apparent that semantic maps are intertwined with knowledge representations and reasoning capabilities. It is, however, somewhat unclear where to draw the line between a semantic map and the associated knowledge representation and reasoning systems. Similarly, managing their inter-dependencies remains an open issue. In [2] Kostavelis and Gasteratos regarded the question of *How semantic maps aid knowledge representation and vice versa?*, as one of the open topics in semantic mapping. They pointed out that ontologies and other formal knowledge representation schemata can yield additional insights into a model of the robots surroundings by encoding and revealing attributes even when these are not perceivable. However, the authors stressed that proper semantic mapping fuels the knowledge representation by recognizing and anchoring entities in the environment to connect spatial and semantic knowledge. For that, they considered creating a *spatially ordered hierarchy* important. This assessment directly points to the challenge of continuously grounding the spatial relations of objects within an environment.

The set of qualitative spatial relations holding in the environment's current state, such as "`Mug2` rests **on** `Desk`" or "`ConferenceTable` is **in front of** the `Robot`", has to be uncovered by inspecting the environment's spatial aspects. To logically reason about the spatial relations between entities by using qualitative spatial reasoning (QSR), they need to be explicitly stored as symbolic knowledge. Qualitative constraint calculi, like the interval calculus [22] or the Region Connection Calculus (RCC) [23], can effectively reason about sets of qualitative spatial relations. Suitable software solutions like the SparQ toolbox [24] exist, but are rarely integrated into semantic mapping approaches. According to Wolter and Wallgrün, this is due to a lack of explicitly available qualitative spatial relations, since the important step called qualification is often missing and remains largely unsolved in practice. The lack of effective tools for grounding spatial relations in sensor data captured from the real physical environment inhibits a wide-spread use of QSR in robotics.

Uncovering spatial relations can be part of the map building and anchoring process. Sjöö et al. presented a combination of an axiomatic system and probabilistic inference to interpret topological spatial relations such is-on or is-in during the mapping process [25]. For additional examples of reasoning with spatial relations in the context of real-world robotics applications, we refer to the comprehensive review by Landsiedel et al. [26].

Grounding spatial relations during the map building pipeline is generally a good approach, but is restricted to processing incoming sensor data and limited to the current excerpt of the environment that is under the robot's scrutiny. Hence, it usually does not scale over the entire environment model, nor does it allow to make spatial queries for objects, whose spatial relations are not yet grounded. Especially, when environment dynamics are considered and a large volume of spatial and semantic data has to be integrated into the semantic map on a continuous basis, maintaining a set of geometrically grounded spatial relations in the knowledge base becomes a tedious task. Hence, effective tools to map from quantitative metric data to qualitative symbolic facts are necessary in the context of semantic mapping, in order to enable the usage of qualitative spatial calculi or other types of formal reasoning over spatio-semantic environment data. It is therefore desirable to provide the capability of grounding spatial relations as a feature of the semantic map, since this complements the handling of spatial relations during map building. In this article, we propose to use a spatial database as a tool to map from geometric data to symbolic spatial relations.

Spatial databases extend relational databases to store, query and analyze geometric data. They enable spatial lookup to search for geometries within a certain region or volume and provide spatial analysis to test if two geometries overlap or intersect. To reduce the evaluation time of spatial relations, spatial indexing techniques are used. Spatial indexing abstracts complex geometries to primitive bounding geometries (2D rectangles or 3D boxes), whose relations can be evaluated efficiently even in large data sets. Most indexing techniques rely on height-balanced search trees of bounding geometries, so called R-trees [27].

Spatial operators determine whether a spatial relation holds between two geometries and map from quantitative geometric data to symbolic spatial predicates. Evaluating distances in 2D and 3D is straightforward, but the analysis of topological and directional relations is subject to extensive research, especially in 3D. Topological relations in 2D have been extensively studied. The DE-9IM model [28, 29] is the standard for spatial databases proposed by the Open Geospatial Consortium (OGC) [30]. An overview of approaches to address 3D topological analysis is given in [31] based on the geometric decomposition scheme presented in [32, 33], to realize the evaluation of 3D intersection, touch and containment. The research on qualitative spatial reasoning (QSR) has proposed various calculi to define and work with directional relations, varying frames of reference and cardinal directions. A comprehensive overview of one and two dimensional solutions is given in [34]. For 3D, Borrmann and Rank describe two approximate approaches using *projection-based* and *half-space* models to analyze directional relations [35].

Spatial databases are commonly used as back-ends for geographic information systems (GIS) in geography, climatology and governmental administration, to store and analyze geographic and cartographic data. GISs primarily offer processing for 2D data, but 3D is actively studied ([36, 37]) and modern solutions provide at least storage for 3D data. However, a full tool set of spatial operators in 3D is still missing.

Since spatial databases already integrate means for spatial analysis on top of storing geometric representations, they are apt candidates for determining qualitative spatial relations in the context of semantic mapping. Therefore, the main contribution of this article is to solve the open problem of grounding qualitative spatial relations in semantic maps by integrating a spatial database into a semantic mapping framework.

We analyzed existing spatial databases and identified the extensions that are needed to make spatial relations qualitative for 3D objects. Besides extending a spatial database with new operators, we present the corresponding schemas and table layouts that are required to support articulated objects and dynamic update of spatial relations when objects are inserted or deleted. Our implementation focuses on making spatial relations qualitative to update the current state of the environment. It serves as a means to generate symbolic knowledge about known facts and spatial relations about the most likely world model. Although probabilistic mapping approaches can be used to determine the current world state modeled in SEMAP, they are not yet considered explicitly in the current implementation.

If the robot's perception provides information on changes in the environment, SEMAP's model can account for these dynamics by adding, deleting or updating its entities. The framework currently does not account for a history of the environment's past states, nor does it provide a set of alternative environment models or a probability distribution over models, to account for uncertainties during the map building process. From a probabilistic perspective, SEMAP represents a maximum likelihood model that is maintained over time.

We illustrate the steps necessary for this integration, based on our proof-of-concept implementation and an exemplary office domain. More domains are presented and discussed in the application examples and in the final discussion.

3. The SEMAP Framework

SEMAP was designed as a representation and reasoning system for environment modeling in robotics. It is based on an object-based environment model in which every entity in the environment belongs to a known concept class, contributes to a set of asserted facts and consists of a spatial model, which can be either a single volumetric body or an articulated kinematic chain of those. To account for the different nature of symbolic and geometric data, SEMAP stores the different kinds of information in dedicated storages. A close connection between the spatial and semantic aspects of an environment is maintained by the framework's spatio-semantic data maintenance layer and querying interface as shown in Fig 2.

3.1. Architectural Concept

All geometric aspects are stored in a PostGIS database and describe the shapes and poses of the individual objects in the environment. For articulated

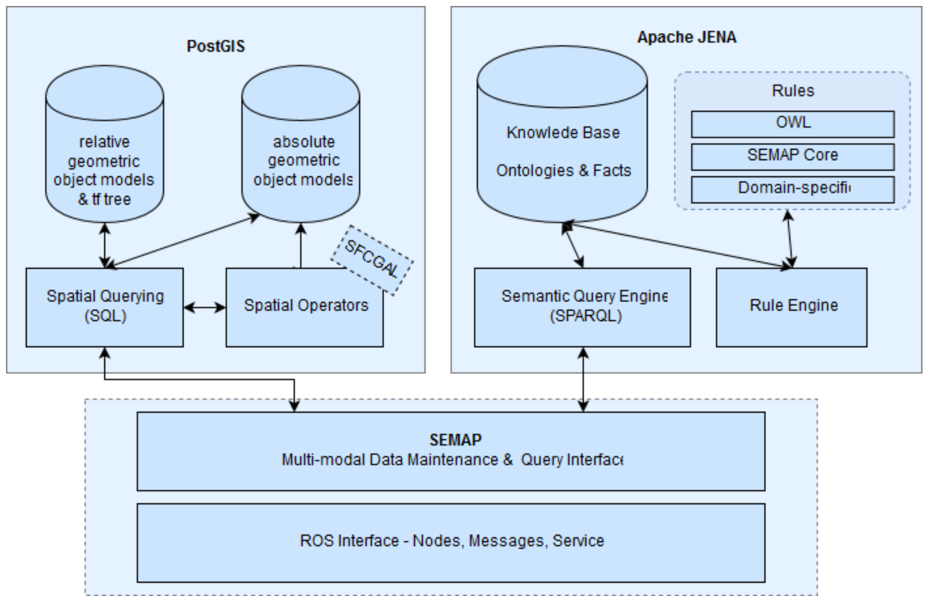


Figure 2: The SEMAP framework consists of a PostGIS database which provides spatial data storage and querying capabilities and an Apache JENA triplet store to maintain the ontological background knowledge and actual facts. Both data domains are coupled via a query interface that can be accessed by robot control systems like ROS.

objects, their kinematic chains and current joint configurations are represented as well. Additionally, the database maintains relational links that connect geometric data sets to their complementary semantic descriptions in a separate knowledge base with factual and conceptual environment information.

The knowledge base uses description logics (DL) [16], featuring the classical separation into a T-box for storing concept definitions including the taxonomy and an A-box for asserted facts. We use a DL-based approach because the underlying ontological models can be constructed to separate domain-independent and domain-dependent knowledge. This helps make the core components application-independent and extensible to different application domains. For that, the T-box maintains a set of domain-independent ontologies that provide a semantic model of the supported geometric types, how they can be combined to form objects and how objects constitute an environment, whereas a domain-dependent ontology provides the necessary vocabulary to describe knowledge about a certain application. Within the A-Box the combined ontological descriptions are used to store facts on individual instance in the environment. Such a system can easily be paired with reasoning modules to enable rule-based inference on the stored environment knowledge.

To communicate with robot control frameworks, we use an intermediate layer between the robot's control architecture and the semantic map representation. This layer provides interfaces to insert information about environment entities from different data sources and handles updating the model. It links the spatial database to the knowledge base by adding URIs to the geometric entities stored in the relational data base that point to the respective instances in the knowledge base. This interface layer handles the incoming queries to retrieve target-specific data and convert it into the required representation.

3.2. Software Components

To represent geometries we chose to use PostGIS as it supports 3D geometries best among the various open source spatial data base implementations available, as shown in Table 1. PostGIS is an open source GIS, based on the relational database PostgreSQL [38], that is compliant with the standards of the Open Geospatial Consortium (OGC). PostGIS provides representations for a number of geometric primitives. These include points, lines, polygons, and collections of geometries, as defined in the "Simple Feature Access" specification [30]. Even though the standard is specified for 2D geometries only, PostGIS also supports three dimensional primitives and includes data types for meshed surface structures based on triangular or polygonal primitives. PostGIS's analytic functions can interpret the spatial information as geographic data in a geodetic reference system or as geometric data in Cartesian space. For spatial querying, PostGIS combines regular R-trees with Generalized Search Tree indices (GiST) to speed up mixed queries with spatial and relational constraints. To analyze 2D simple feature geometries, PostGIS uses the GEOS library [39], which provides an extensive tool set of spatial operators. Native PostGIS only supports few operations on 3D data, but can be extended with custom operators using the SFCGAL plugin [40]. The SFCGAL project defines an interface to the Computational Geometry Algorithms Library (CGAL) [41], which provides an extensive set of geometric algorithms. These algorithms can then be used to define additional 3D spatial operators for PostGIS. PostGIS in combination with the SFCGAL extension realizes the storage of spatial environment data consisting of both 2D and 3D geometric primitives. For spatial analysis, the close integration of CGAL allows the missing spatial operators to be implemented for 3D geometries.

In addition to using PostGIS, we have implemented a prototypical integration of the semantic web framework Apache JENA to support query languages like SPARQL. We will not dive into the details of this approach – as it is work in progress – but present a preliminary example of the ontology that will be used to link the spatial database to the knowledge base.

To demonstrate the use of SEMAP on a real robot, we implemented an interface to the Robot Operating System (ROS). This will be made public as an addition to SEMAP together with the ROS bindings and the reference data set presented in this article.

3.3. Ontological Model

The ontological model underlying a environment representation in SEMAP is comprised of two parts: SEMAP's core ontology, which is independent of any domain specific application and a domain-specific ontology, which may be changed depending on the application.

SEMAP's core ontology gives the conceptual background for representing the spatial elements within an environment model as presented in Figure 3. These concepts are closely related to the data base layout of the PostGIS back end, as will be discussed below. The ontology uses standards from the Open Geospatial Consortium (OGC), because these well-defined models of geo-spatial data are in alignment with PostGIS's data types, which were also defined by the

Name	**PostGIS**
Description	spatial database extension for the PostgreSQL database
Authority Website	`http://postgis.net`
Spatial Types	Points, LineStrings, Polygons, MultiPoints, MultiLineStrings, MultiPolygons, GeometryCollections, Triangle Irregular Networks, Polyhedreal Surface
Spatial Index	R-tree-over-GiST spatial indexing for high-speed spatial querying
Spatial Functions	Over 300 functions and operators, no geodetic support except for point-2-point non-indexed distance functions, custom PostGIs for 2D and some 3D, some MM support of circular strings and compound curves
Name	**MySQL**
Description	Includes a limited set of spatial representations and queries natively.
Authority Website	`http://www.mysql.com`
Spatial Types	Geometry, Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon, GeometryCollection
Spatial Index	R-Tree quadratic splitting-indexes only exist for MyISAM
Spatial Functions	OGC mostly only MBR (bounding box functions) few true spatial relation functions, 2D only
Name	**Spatial Lite**
Description	SQLite with spatial datatypes, functions, and utilities
Authority Website	`https://www.gaia-gis.it/fossil/libspatialite/home`
Spatial Types	Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon
Spatial Index	R-Tree variants
Spatial Functions	Basic functions for Point, LineString and Polygon

Table 1: Comparison of open source spatial database implementations regarding their spatial types, spatial indexing technique, and available spatial operators. Adapted from `http://infolab.usc.edu/csci587/Fall2016/`

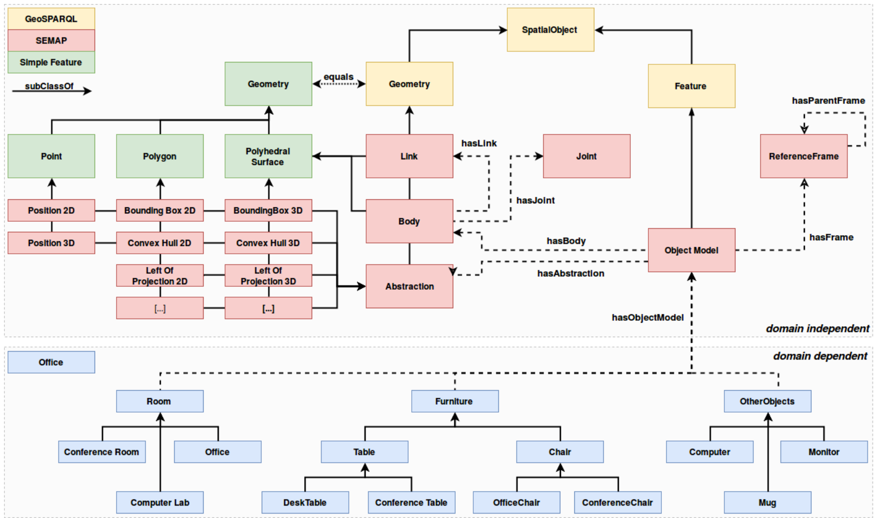


Figure 3: An excerpt from the ontology that is used to fuse the labels of objects stored in the spatial data base with semantic concepts in different domains.

OGC. GeoSPARQL's `SpatialObject` and the fundamental distinction between geometries and features are integrated in SEMAP's upper ontology. Here, the concept `Geometry` describes any kind of spatial primitive and provides a semantic wrapper for all OGC data types and serves as a bridge to the well known Simple Feature Ontology. SEMAP's KB contains a corresponding instance of a `Geometry` sub-concept, for every geometric primitives stored in SEMAP's DB. The property `semap:hasDbId` is used to create an associative link between the geometric primitive and its semantic wrapper. SEMAP internally uses these associations to join spatial and semantic data, in its query interface.

The super-concept `Feature` is used for all things that can be described spatially like SEMAP's `ObjectModel`, which aggregates sets of semantically wrapped geometries to represent an object. For this, it uses the `geo:hasGeometry` property and its two specializations: `semap:hasBody` composes a set of geometries that constitute the object's actual body. In case of articulated objects, the `Link` and `Joint` concepts are used to describe the object's kinematics. `semap:hasAbstraction` provides a set of coarser representations, like oriented

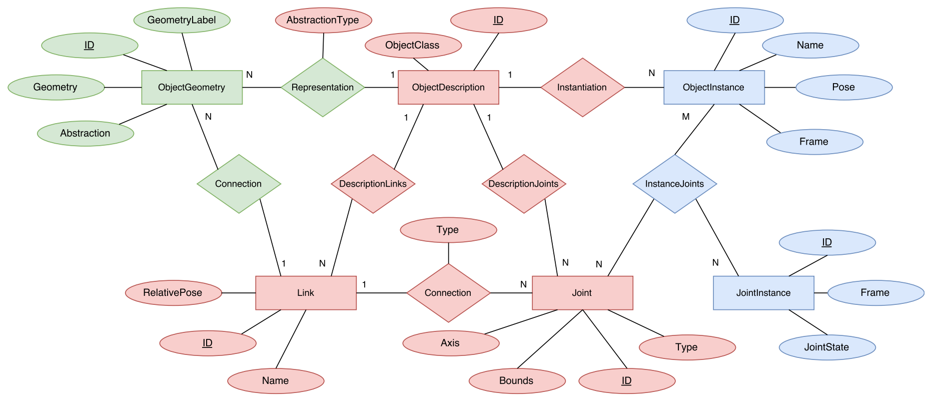


Figure 4: The database schema for representing the environment model. Explanations provided in the text.

and axis-aligned bounding boxes and convex hulls. These abstractions are used for accelerated spatial processing and enable the analysis of directional relations like `left-of` or `above-of`, based on projection and halfspace geometries [42].

To create a spatio-semantic environment model for a particular application, domain-specific ontologies, knowledge bases and rule-sets can be imported into SEMAP's knowledge base component. To describe domain-specific concepts spatially and reason about them as part of SEMAP's environment model, the respective entities can be associated with an `ObjectModel` via the `hasObjectModel` relation. Figure 3 shows this by connecting objects from a simple ontology describing objects and rooms in an office environment to the SEMAP core ontology. The used ontology is in partial alignment with the indoor furniture classification ontology used in our previous works on semantic mapping [43].

3.4. Database Schema

Figure 4 displays the database schema for storing semantically annotated objects in the spatial database. This schema is roughly divided into three parts: the representation of object *classes* (red), individual object *instances* (blue), and their different *geometric* representations at different abstraction levels (green).

To connect the geometric models in the database and the conceptual representation in the ontology, the entity ObjectDescription has an attribute ObjectClass that maps the description in the database to one of the concepts in the ontology (ie. to the concept `office:Mug`. To represent articulation, each object class can consist of several Links and Joints that are connected in a kinematic chain. The individual ObjectInstances have individual Names to have a readable label besides the internal ID, which is aligned with the semantic wrapper's `hasDbID` property. To model articulation, each object instance can represent individual JointInstancess that are linked to connecting ObjectGeometries via JointConnections and LinkConnections that refer to the object descriptions links and joints.

In our modeling, the ObjectDescription entity represents the generic spatial model of an object class that can be instantiated via the Instantiation relation. Since the individual attributes are stored in the blue instantiation relations, the geometries associated with the object descriptions can be re-used to prevent storing identical geometries multiple times. SEMAP supports 3D polyhedral mesh data to describe the body geometries of each individual part of an object. The individual configurations of the partial geometries are transformed according to the instances' poses and joint states. Since geometric queries in 3D can be computationally expensive, we can store object geometries at different *abstraction* levels. For example, the precise polyhedral mesh representation of a CAD model can be abstracted by its bounding box or convex hull, which can be used for efficient but less precise qualification. These abstractions are initialized when the objects are inserted into the database and updated dynamically. Examples of the computed abstractions are shown in Figure 5. SEMAP's default abstractions are 2D and 3D axis-aligned and oriented bounding boxes, and convex hulls. Additionally, point-based abstractions are also computed. These auxiliary geometries are created with functions from PostGIS and SFCGAL for the entire object as well as for each individual link. The level of abstraction is stored in the attribute AbstractionType in the Representation relation. By convention, all object geometries are defined in a right-handed coordinate system and the base link of an object is placed at the object's bottom, as it

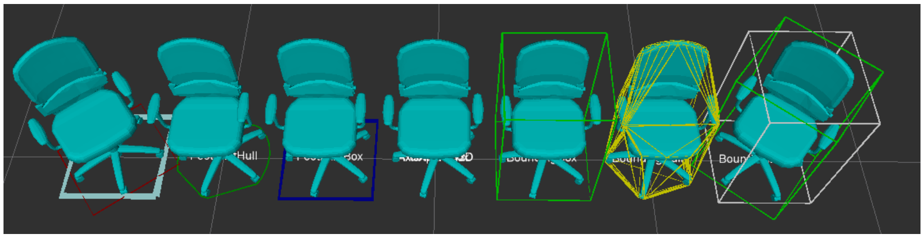


Figure 5: SEMAP provides a set of geometric abstractions to enable accelerated spatial queries. From left to right: 2D axis-aligned bounding box, 2D convex hull, 2D bounding box, 3D bounding box, 3D convex hull, and 3D axis-aligned bounding box. The axis-aligned bounding boxes (in gray) are overlaid with the oriented bounding boxes for comparison.

is often done when using the Unified Robot Description Format (URDF). For convenience, SEMAP supports the direct import of URDF files.

Semantic information about a geometry is stored in a GeometryLabel string that labels the sub-part of the object. These refer to a semantic description that is maintained separately in the dedicated knowledge database and linked to the spatial database table via this label. This way, the semantic description of the object is directly integrated into the PostGIS database, so that we can use relational queries on these labels to emulate data retrieval based on object semantics. We use this feature to perform spatial queries in PostGIS to ground certain spatial predicates, which are then asserted to the knowledge base as facts.

As described so far, the object descriptions are only the blueprints from which instances are created to model the actual environment. To build an actual environment model, the ObjectInstance table combines a reference to an object description with the position and joint states of an actual instance. To manage positional information within the environment model, SEMAP implements a relative positioning system using a transformation tree. Frames in the transformation tree span local coordinate systems, in which the relative positional information is expressed. These frames are defined with respect to each other and create a directed tree. At the root of the tree, the global root frame defines the global coordinate system. The relative frames of all objects can be transformed into this system by traversing the tree. With this transformation tree, SEMAP also supports a common practice in many robotic systems (in analogy to ROS's TF library), but in persistent storage. That allows to preserve the environment's state during robot downtime, which is required in long-term applications.

The implementation of the transformation tree is realized in the Frame table, which Figure 4 does not show for sake of readability. This relation connects the Pose of an object part's instance to the frame to which it is related, via a reference to the frame of the parent object. Each object instance has a pose, which is the anchor for the object's base link. Additionally, each joint instance has a frame to allow for a frame-based view on the object's entire kinematic chain. Another important function of SEMAP's transformation tree is to build a bridge between two different views of an instance's spatial representation.

Up to this point, we have described the relative view, which is taken in the context of a frame-based positioning system. However, once an object instance is subject to SEMAP's spatial query system, there is also the demand for an absolute view on the object's geometry, because relative geometric information can not be processed by PostGIS's R-tree implementation. In PostGIS, all geometries have to stem from the same global reference frame. In order to obtain reasonable results in the spatial analysis, SEMAP maintains a second object description for each object instance that provides a copy of the relative description's geometries and abstractions in absolute coordinates. To create this view, the transformation from the root frame to the instance's frame is applied to all the geometries stored in the relative description. Since this is a potentially expensive operation, SEMAP creates full absolute representations only on demand. By default, only the description's abstractions are transformed. All absolute representations are cached and reused, until they expire, which happens every time the object changes in pose or configuration. Since an instance's

Operator	2D Geometries	3D Geometries	3D TIN	3D Polyhedron
MinDistance	✓/ ❍	✓/ ❍	❍	✓/ ❍
MaxDistance	✓	✓	✗	✓
WithinRange	✓	✓	✗	✓
FullyWithinRange	✓	✓	✗	✓
ShortestLine	✓	✓	✗	✓

✓: native PostGIS ❍: SFCGAL plugin ✗: currently not implemented

Table 2: List of PostGIS's metric spatial operators and the geometric primitives supported.

frame can be the reference frame for other objects, any change affects all objects that descend from it.

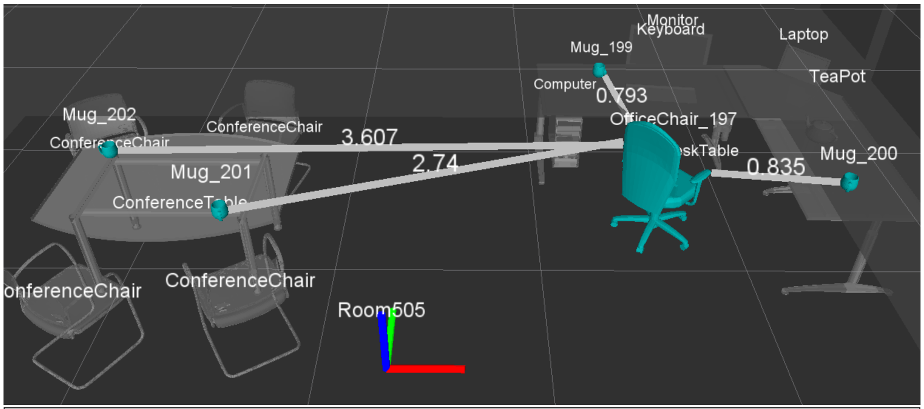
3.5. Spatial Operators

Next, we will review the spatial operators available in PostGIS and discuss their usage in robotic applications. We will distinguish them by their support for the following datatypes: basic 2D and 3D geometric primitives (points, lines, polygons) and 3D triangle and polygonal mesh data. We also discuss how to construct custom operators using the SFCGAL plugin for operators that are missing in native PostGIS, but are required for robotic applications.

Metric Operators

PostGIS offers a number of metrical operators to measure the minimal and maximal distance between geometries, to test whether a geometry is (fully) within a parametrized range of another geometry or not and to return the shortest or longest line between two geometries. These operators are available for most 2D and 3D geometries, except for the TIN type, which is implemented as SFCGAL extension that offers minimal distance measures. A list of all operators available in SEMAP is presented in Table 2.

In robotic applications, metric operators are a valuable tool to look up objects within a certain range around a query location, such as the robot's posi-




```
rosservice call /get_distance_between_objects
"reference_object_types: ['OfficeChair'] reference_object_geometry_type: 'BoundingHull'
 target_object_types: ['Mug'] target_object_geometry_type: 'BoundingHull'
 sort_descending: false max_distance: false"
```

Figure 6: Example of using SEMAP's ROS interface to measure the minimal distance between all `OfficeChairs` and all `Mugs` using the 3D convex hulls.

tion. SEMAP utilizes the different operators to provide distance-based queries across the various geometric representations of object models. Figure 9 shows the implementation of SEMAP's operator to measure the minimal or maximal distance between objects. It allows to sort results in ascending or descending order, which is beneficial in prioritizing objects. Figure 6 illustrates the usage of the operators using 3D convex hulls.

Topological Operators

PostGIS implements the DE-9IM calculus [29]. Most common topological relations, such as equals, intersects, covers and touches, can be evaluated for 2D geometries. Similarly, equality, intersection and containment tests for the Box2D type and an intersection test for the Box3D type are available. Additionally, SFCGAL provides intersection tests for all 3D geometries as shown in Table 3.

Among these operators, those evaluating containment and intersection relations are most valuable and versatile for robotic applications. On the one hand,

Operator	2D Box	2D Geo.	3D Box	3D Geom.	3D TIN	3D Polyh.
Containment	✓	✓	✚	✗	✗	✚
Intersection	✓	✓	✓	❍	❍	❍
Touch	✗	✓	✗	✗	✗	✗
Equality	✓	✓	✗	✗	✗	✗

✓: native PostGIS ❍: via SFCGAL plugin ✚: custom extension ✗: currently not implemented.

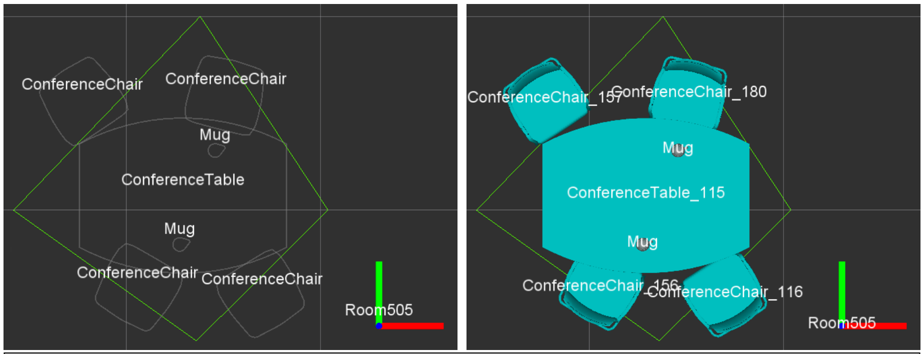
Table 3: List of available topological spatial operators and the geometric primitives supported.

they allow for spatial look-up by identifying if an object's geometry lies within (or at least intersects with) another geometry. In this aspect, containment operators work similar to metric operators, but exceed them in flexibility, since potentially arbitrary areas or volumes can be queried. On the other hand, they allow to ground the spatial predicates that hold between objects, which makes topological operators highly relevant for applications in semantic mapping. By applying topological operators on SEMAP's environment model, all objects that are `in` a certain area can be queried to create the respective semantic knowledge, which in turn can then be processed by qualitative spatial reasoning techniques separately from the geometry with justification from a geometric evaluation.

Unfortunately, both PostGIS and SFCGAL offer no 3D containment tests. Hence, we extended SFCGAL with such operators by using existing CGAL algorithms. The current implementation is limited to detect containment for a set of target points or a polyhedral body within a reference polyhedron. It enables SEMAP to evaluate 3D containment on all 3D bounding volumes, which are represented by polyhedral mesh data. Examples for SEMAP's 2D and 3D containment tests are presented in Figure 7 and 8.

Directional Operators

PostGIS natively provides a set of *directional operators* to identify if a geometry is `left-of`, `right-of`, `above-of` or `below-of` another geometry. They operate on the 2D axis-aligned bounding boxes of the query geometries only.




```
rosservice call /get_objects_within_area
"reference_polygon: ...  target_object_types: ['Chair','Table']
target_object_geometry_type: 'FootprintHull'  fully_within: false"

target_ids: [115,156,116,157,180]
```

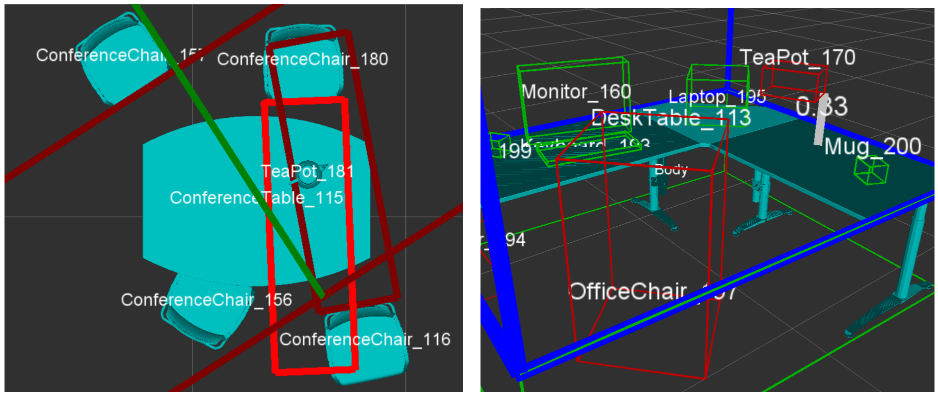
Figure 7: The presented operator evaluates the object's 2D convex hulls against a reference polygon. Since the operator's relaxed interpretation was chosen, intersecting objects were included in the query result, as well as those fully within the reference polygon.

Thus, they identify the directional relations with respect to the *extrinsic* global reference frame, but not based on the object's *intrinsic* reference frames, which limits their utility for robotic applications. Figure 8 (a) demonstrates this problem: the native operators can not infer that the *ConferenceTable* is `in-front-of` most of the *ConferenceChair*s, but `behind-of` *ConferenceChair116* as to the chair is facing away from the table. Another issue is that the operators neglect the third dimension, which makes it impossible to determine that the *TeaPot* in the depicted scene is `above-of` the *ConferenceTable* in a three-dimensional sense. To overcome these shortcomings and allow for spatial analysis using 3D directional relations and intrinsic reference frames, we implemented the projection-based and halfspace-based model, as proposed by Borrmann [42].

In the projection model, the faces of each object's bounding box are extruded to create six box geometries on top of every face. The extrusion's distance is determined by multiplying the object's extent in the respective dimension by a scaling factor. In the half-space model, six additional box geometries are created by first extending the bounding box faces along the two secondary axes before extruding along the primary axis. The extrusion's direction follows the conventions for object descriptions and both models are stored within an object description's set of abstraction models and transformed accordingly for each object instance. These additional box geometries can be used to evaluate 2D and 3D directional relations from the object's intrinsic point of view. The containment operator is used for a *strict* interpretation of directional relations, labeling only those object to be in the tested relation if they are completely within the projection space. For a *relaxed* interpretation, the intersection operator is used, which allows for partially included objects, too. The presented directional models are quite basic and could, if required, be exchanged with more elaborated models.

Figure 8 presents examples of these auxiliary geometries. In (a), the light red 2D box extending *ConferenceChair180*'s front, as well as the dark red box extending from *ConferenceChair116*'s back constitute projection geometries. These geometries now properly reflect the reference object's intrinsic viewpoint, e.g., *ConferenceChair180* is `behind-of` *ConferenceChair116*, whereas *ConferenceChair116* is `in-front-of` *ConferenceChair180*. (b) presents a 3D example for detecting directional relations. The blue box above the desk's top is used to detect objects that are above it, such as monitor, laptop, and mug.

By combining directional and topological operators, additional spatial relations can be identified. To evaluate the `is-on` relation, SEMAP pairs the strict 3D `above-of` operator with an additional distance constraint that rejects all target objects beyond a certain threshold distance, such that it can find all objects that are on another object. In Figure 8 (b), for example, the chair's bounding box violates the strict `above-of` relation relative to the desk's top projection, and the teapot's bounding box exceeds the distance threshold, because it was artificially placed way above the desk's surface. All other objects are correctly classified as being on the desk.



(a) 2D Directional Models (b) 3D Directional Models

Figure 8: An application example of SEMAP's custom directional operators in 2D and 3D.

3.6. Query System

Querying SEMAP for environment knowledge is done by using the methods defined in SEMAP interface layer. This interface layer also handles the synchronization between SEMAP's spatial data base and the knowledge base.

The PostGIS back-end is queried using SQL, whereas the Apache JENA triplet store provides a SPARQL endpoint. Currently, there is no automatic synchronization between the two querying interfaces, such that the synchronization has to be explicitly triggered in the correct order.

First, the spatial database is triggered to evaluate binary spatial operators that identify a relation between two geometries, e.g., determine the distance between two objects. For this, a set of reference and target geometries must be assigned. To restrict the set of geometries in terms of their object classes, SEMAP relies on the semantic labels assigned in the PostGIS data base. This allows to impose semantic constraints during the spatial querying process. It is also possible to refer to specific objects by using their IDs directly. The type of spatial representation can be constrained, as well. SEMAP allows to use both complete body geometries, as well as the given abstractions in 2D and

```
def get_distance_between_objects( call ):

if call.max_distance:
        distance = ST_3DMaxDistance(ref_geo.geometry, tar_geo.geometry)
else:
        distance = ST_3DDistance(ref_geo.geometry, tar_geo.geometry)

pairs = db().query( ref_obj.id, tar_obj.id, distance ).\
        filter(
                ref_geo.id.in_(get_geo_ids(ref_obj, obj_const, geo_const)),
                tar_geo.id.in_(get_geo_ids(tar_obj, obj_const, geo_const)),
                ref_obj.id != tar_obj.id )

if call.sort_descending:
        result = pairs.order_by( desc( distance ) ).all()
else:
        result = pairs.order_by( distance ).all()
```

Figure 9: A code excerpt of SEMAP's distance measurement operator.

3D. All geometries must obviously be drawn from the absolute view on the object instances. The semantic and geometric constraints are evaluated prior to filtering. An example is presented in Figure 9.

Once executed, the spatial query returns pairs of object IDs that satisfy the spatial relation tested for. Depending on the operator, additional information is returned as well, i.e., the respective distance between the objects. The spatial relations are now grounded in terms of a quantitative geometric analysis. Next, as they represent facts about spatial predicates holding between entities, they are accordingly asserted in the OWL-based A-box in JENA's triplet store. After the insertion it is possible to semantically query for spatial relations. In this case, the SEMAP ontology provides additional information about the entities and relations encountered in the environment's domain through ontological reasoning about the conceptual background knowledge in the T-box. Other types of inference, for example, using rule-based reasoning, can be used from here.

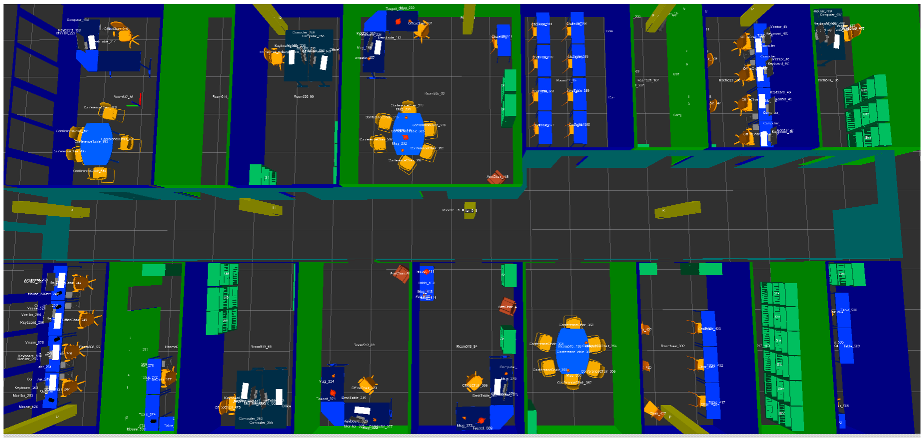


Figure 10: The created data set remodels an office environment at Osnabrück University.

3.7. Performance Evaluation

To evaluate the performance characteristics of spatial queries, we conducted a set of experiments. Since SEMAP makes no assumptions about the environment model's data source, it allows to build environments from sensor data, as well as from provided CAD models. For the following evaluation we constructed a reference data set from CAD data modeling the building of the Computer Science department at the University of Osnabrck depicted in Figure 10. The data set contains a total of 300 individual object instances that were created from 35 different reference models used as gemetric object descriptions. The objects are spread across 18 different rooms, which resemble real offices, computer labs and seminar rooms. Examples of how to generate similar maps from real sensor data are presented in Section 4.1.

Based on this data set, we conducted a set of test queries, to exhibit the performance of SEMAP's spatial querying and to differentiate different strategies for using spatial operators. All test where conducted on a Lenovo ThinkPad W530 with Intel Core i7-3940XM (4x 3.0 GHz, 8MB cache) and 8GB RAM.

First, we tested the execution of the containment operator using two dif-

Dim.	Reference			Target			Num. Tests	Time/Test [s]	Total Time [s]	Num. Pairs
2D	Room	BB2D	18	All	POS2D	300	5400	0.000007	0.039337873	282
	Room	CH2D	18	All	POS2D	300	5400	0.000012	0.064801216	279
	Room	BB2D	18	All	BB2D	300	5400	0.000011	0.061866045	278
	Room	CH2D	18	All	CH2D	300	5400	0.000014	0.074759007	275
	All	BB2D	300	All	POS2D	300	90000	0.000003	0.269836902	439
	All	CH2D	300	All	POS2D	300	90000	0.000004	0.325406074	430
	All	BB2D	300	All	BB2D	300	90000	0.000002	0.203353166	363
	All	CH2D	300	All	CH2D	300	90000	0.000003	0.265438796	360
3D	Room	BB3D	18	All	BB3D	300	5400	0.033033	177.784672022	268
	Room	BB3D	18	All	POS3D	300	5400	0.016201	90.110987186	274
	All	BB3D	300	All	BB3D	300	90000	0.036631	3274.874104981	278

Table 4: Performance evaluation of the strict 3D containment operator.

ferent query types. The first type creates an inventory list for all rooms by performing a many-to-many strategy with instances of `Room` as reference set and an unrestricted target set. The second query type provides a full enumeration of all object pairs matching the `is-in` relation, by using a completely unconstrained many-to-many strategy. Both queries were performed using the strict containment operator in 2D and 3D and on different abstraction levels.

The results shown in Table 4 provide two insights. First, increasing the geometric abstraction level decreases query selectivity and vice versa. Testing, for example, 2D positions against the 2D bounding boxes returns more results than testing bounding boxes against each other. This is expected, since the latter is more restrictive. Evaluating against convex hulls is even more selective. The same holds for the comparison between queries executed in 2D and 3D, here evaluating in three dimensions is obviously more selective.

Secondly, increased accuracy comes at computational cost and vice versa. Comparing the 2D data sets reveals that testing positions or bounding boxes against bounding boxes is considerably faster than testing against convex hulls. This is due to the fact that the necessary tests can be performed in constant time, since both geometries are of fixed size, whereas the geometric complexity of the convex hulls is usually higher and also varies among objects. In the 2D case, these differences are negligible, since each test only takes a couple of microseconds, so that even a large number of tests can be performed in reasonable time. The full enumeration of containment relations on 2D convex hulls was executed in 0.26 s for total of 90.000 tests.

For 3D spatial queries, however, the situation is different. Testing a single pair of 3D bounding boxes takes about 35 ms, which is reasonably fast for a small number of queries, but with an increased number of tests, the query time accumulates to minutes or more. The full enumeration of all containment relations using 3D bounding boxes, for example, took over 54 min. This tendency was expected, but PostGIS's performance on 3D geometries seems to leave room for optimization. Currently, the poor scaling of the 3D operators renders the direct evaluation of 3D spatial relation useless, especially in robotic applications that need near-realtime response. It is, however, possible, to narrow down the set of geometric tests, which addresses these performance problems, as described next.

3.8. Increasing Performance

The first strategy is to successively apply spatial operators, with an increasing level of selectivity and computational complexity. By applying coarser but quick to compute spatial tests, we narrow down the object pairs that need to be tested with computational expensive operators.

Figure 11 illustrates this strategy on a simple scene. Here, we want to test which objects shown in (a) are on the table. We could test for the 3D relations holding between all objects in this scene, directly or apply a 2D query as a filter query before. In (b), the convex hulls of the objects are shown. Querying for strict 2D containment reveals that mugs and tea pot are fully contained in the table's convex hull, while the chairs are not. Therefore, we can immediately rule out that the chairs may be on the table. The given 2D containment, however, may indicate that the target objects are either `in` or `on` the table or that 3D directional relations, such as `above-of` and `below-of`, may hold as well. We can therefore continue with testing for `is-on` based on the 3D bounding boxes.

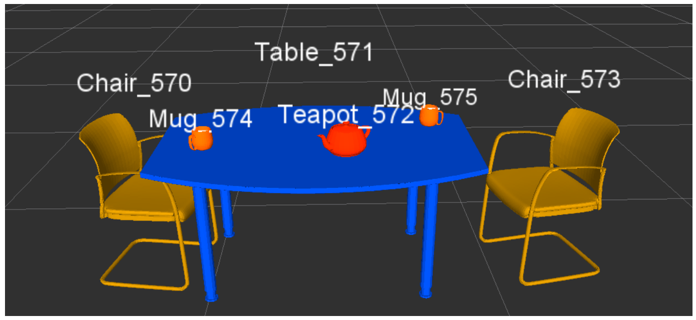
Testing for a 2D relation before applying the more complex 3D spatial operators, can effectively reduce the number of tests. To exemplify the advantages of this approach, we conducted the previously described experiment again, but used a 2D containment before testing for 3D containment. This significantly reduced the number of 3D tests from 90000 to 359, as well as the total runtime for testing from 54 min to 14 s. The results are shown in Table 5.

OP	Reference				Target				Num. Tests		Time [s]	Num. Pairs
2D/3D	Room	BB2D	BB3D	18	All	BB2D	BB3D	300	5400	277	9.9509649270	268
2D/3D	Room	BB2D	BB3D	18	All	BB2D	PT3D	300	5400	277	4.837368965	277
2D/3D	All	BB2D	BB3D	300	All	BB2D	BB3D	300	90000	359	14.018936157	278
2D/3D	All	BB2D	BB3D	300	All	BB2D	PT3D	300	90000	359	6.91635704	281

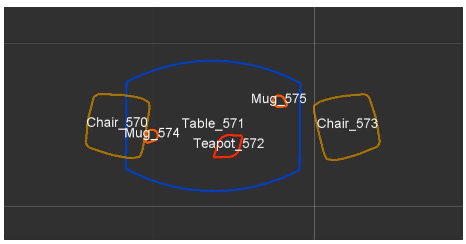
Table 5: Performance evaluation of the strict 3D containment operator using a 2D pre-query.

Similar performance increases can be produced for all other spatial operators when testing for intersections or directional relations. The actual run times vary from operator to operator and are dependent on the number and the complexity of the involved geometric tests. Using pre-queries to accelerate the query process is a strategy that can also be used across all spatial relations and on the different geometric representations and abstractions of an object. The choice is usually dependent on the application and always a trade-off between computational complexity and spatial accuracy. By default SEMAP's query interface already applies suitable 2D pre-queries, before executing 3D spatial queries, to allow online robotic applications.

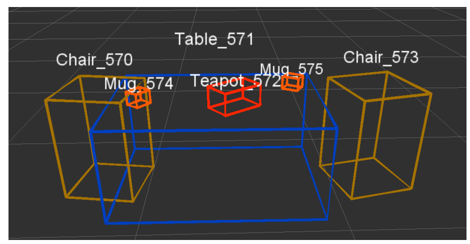
Another strategy to optimize query performance while keeping geometric accuracy, is decomposing objects into their individual parts before testing. By default, SEMAP's spatial query system performs object-to-object tests, using either the object's body geometry or a geometric abstraction that covers the entire object. Figure 11 shows two problems that arise: While geometric analysis on the actual object's body returns the most accurate evaluation, it is very costly, especially when the models are as detailed as the chairs in (a). An evaluation on the entire object's 3D orientated bounding boxes is faster, but may not provide the required level of detail. In (c), for example, an intersection would



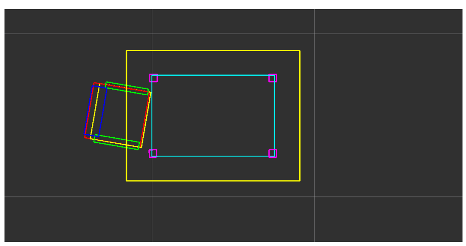
(a) 3D Scene



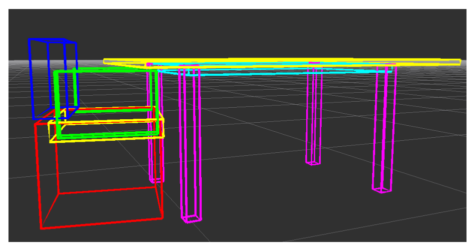
(b) 2D Convex Hulls



(c) 3D Bounding Boxes



(d) 2D Decomposition



(e) 3D Decomposition

Spatial Relation	**Geometry**	**Target**	**Reference**	**Result**
`is-in`	CH2D	Teapot572	Table571	True
`is-in`	BB3D	Teapot572	Table571	False
`is-on`	BB3D	Teapot572	Table571	True
`intersects`	GEO3D$_{body}$	Chair570	Table571	False
`intersects`	BB3D$_{body}$	Chair570	Table571	True
`intersects`	BB3D$_{links}$	Chair570	Table571	False

(f) Spatial Queries.

Figure 11: (a) shows a simple 3D environment, (b) the 2D convex hulls and (c) the 3D oriented bounding box of the entire object, whereas (d) and (e) show the oriented bounding boxes of the individual parts of the objects in 2D and 3D, respectively. (f) shows the evaluation of different spatial relations and their results.

be found for *Chair570* against *Table571*, even though there is none between the object bodies. Since SEMAP supports composite objects and kinematic chains, it is further possible to query for spatial relations by considering the individual links of objects. By decomposing the objects into parts as shown in (e), we can significantly reduce the computational complexity and still determine the desired results, as (f) shows. To find cases where this object decomposition scheme is valuable, again a 2D query for intersecting object footprints can be used, before applying more complex operators. In (d), for example, the intersection of the entire object's 2D convex hulls can be used as an indicator that the links may need to be evaluated individually. Increasing the level of detail, e.g., by checking (d) the link's individual 2D bounding boxes, may then provide the information that a 3D test must be executed only for the table's surface against the chair's seat, legs and arm rests. Testing these links against each other using 3D bounding boxes, finally reveals that no links intersect and applying the directional operator shows that these parts of the chair are indeed below the table surface, as shown in (e).

We manually segmented the chair's CAD model and imported it into SEMAP via an URDF description. One could, however, add automatic object decomposition functionality to the framework, to use this strategy without additional manual effort. See [44] for a suitable approach.

4. Practical Applications

This section demonstrates how to perform combined spatial and semantic queries with the SEMAP framework in order to support various applications that benefit from semantic environment data.

4.1. Map Generation from Sensor Data

For practical applications it is crucial that SEMAP is able to handle semantic information from a real mapping process on a mobile robot. Since the framework can handle all geometric data types supported by PostGIS, it is possible to add semantically annotated objects to SEMAP from different mapping approaches. Figure 12 shows results from an approach that uses CAD reference models for semantic classification in RGB-D data [43]. If CAD models are not available, surface reconstruction methods in combination with semantic classification based on planar constraints can be used to create a semantically annotated polygonal representation from incoming sensor data, as shown in Figure 13.

This is to illustrate that arbitrary annotated polygonal data – including appropriately converted octree representations – from actual robotic data can be fed into SEMAP instances. For this article, we tested SEMAP with data from the approaches presented above, but the integration of other reference data sets like NYU [46], Robo@Home [47] or others is clearly feasible after the implementation of suitable converters.

4.2. Topological Structuring

Performing spatial and semantic analysis on the environment model can make information explicit that is otherwise only implicitly encoded in the data. The topology of an environment, for example, is covert in the spatial relations that hold between objects and can be revealed by applying spatial operators. The extraction of topological knowledge is a key feature of our semantic mapping framework and is of great benefit for path planning and exploration, especially during the initial map building process, when many spatial relations need to be

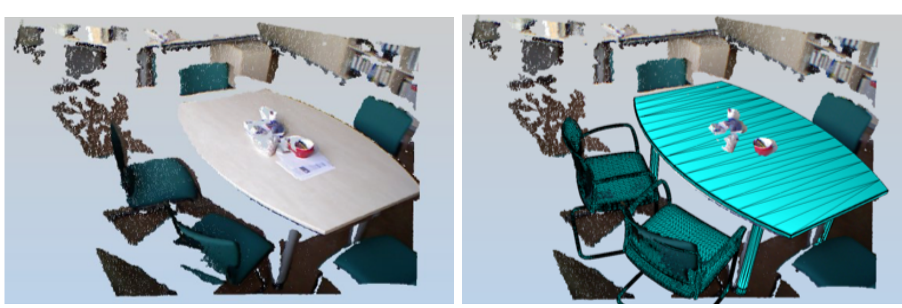


Figure 12: Detection of furniture instances from RGB-D data using CAD reference models as presented in [43].

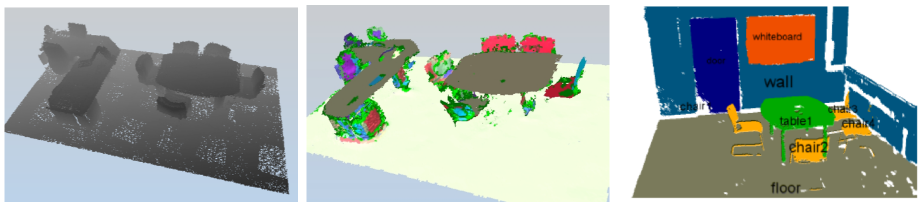


Figure 13: Semantic labeling of polygonal reconstructions from point cloud data (left) based on normal orientations and planar relations [45, 9] (center) to a SEMAP model (right).

grounded at once.

To bootstrap assertions on topology underlying our test data set, we used queries like the ones evaluated in Section 3.7, to create an inventory of all rooms and used it to insert the objects into SEMAP's knowledge base afterwards. The obtained spatial predicates were then used to restructure the environment's transformation tree to reflect the topological relations between the objects. We use the containment relation, to bind the objects found in each room to the reference frames of the respective room. See Figure 14 for an example.

A subsequent query identified all objects that are `on` objects of type `Table`. The results were also used to bind the target objects to their parent's frame. This step is illustrated in Figure 15: (a) shows a the transformation tree of a single room before and (b) after the objects are bound to their supporting tables. (c) shows a close-up of single table. Since the redirection of a reference frame is negligible, the run time of a batch-wise topological restructuring compares to the performance of the strict containment operator as shown in Table 5. Of course the same procedure can be applied for other common objects with surfaces, as well as parts of objects, e.g., the boards of a shelf.

Applying topological restructuring of the relative transformations brings several benefits: First, objects move together with their topological parent, e.g., a mug bound to a table moves if the table is moved. Second, the explicitly encoded topology can be queried directly from the transformation tree, which is considerably faster. An example: a spatial containment query to evaluate the

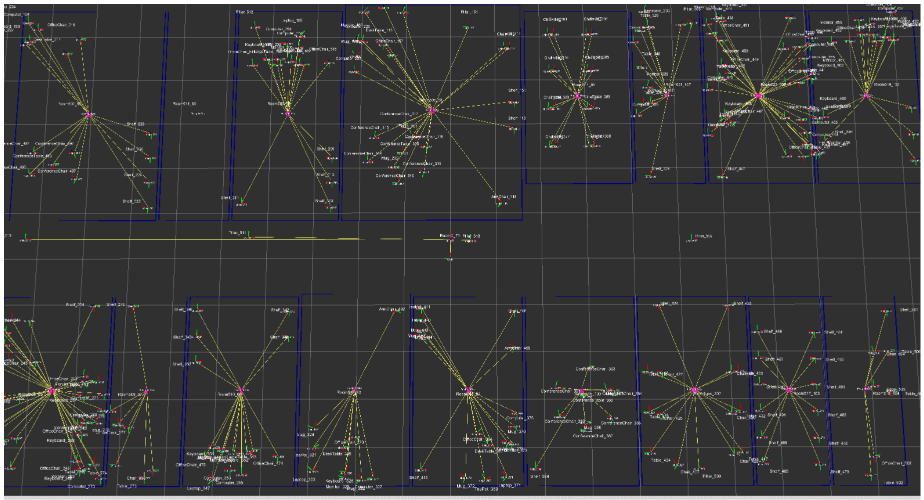


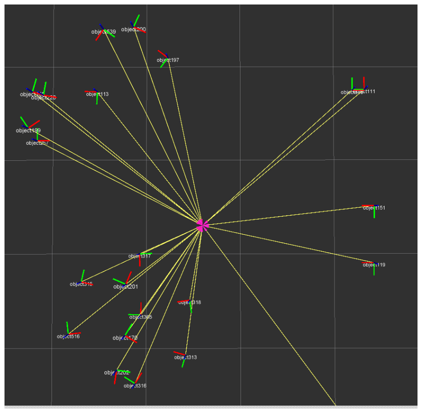
Figure 14: Applying containment queries on a global scale allows to structure environments topologically. Here, the 2D bounding boxes of all `Room`s were used to structure the environment displayed in Figure 10.

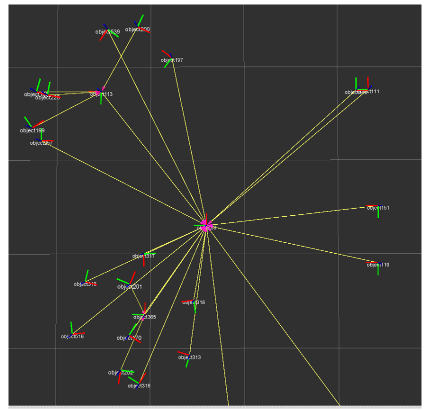
objects within `Room505` took about 0.94 s, whereas retrieving the same inventory list from the transformation tree after the environment was topologically restructured took merely 0.0025 s. This significant drop in retrieval time is owed to the fact that a relational database lookup is considerably faster than spatial queries, as no geometric analysis is involved. Third, all explicit relations can be returned as a topological graph that can serve as input for topological navigation, without taking the detour over the knowledge base.

4.3. Object and Scene Classification

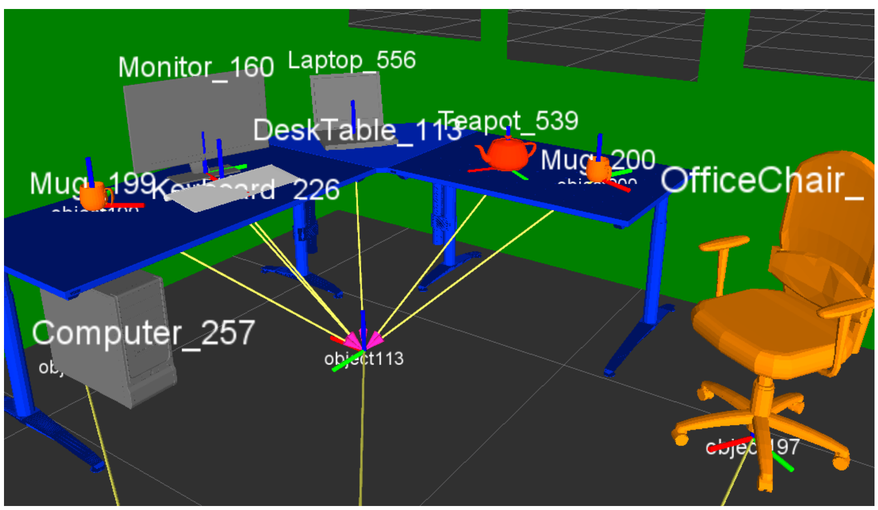
Topological analysis can also be the source of further insight into the environment's semantics. For instance it is possible to classify groups of objects into high-level aggregates or discriminate between type of rooms depending on their inventory. This requires suitable background knowledge and rules that discriminate object properties, assign additional attributes or create new entities.

Such rules can be implemented, for example, by adding an SRWL rule inter-





(a) Topologically Structured By `in` (b) Topologically Structured By `in` and `on`



(c) 3D Close-Up

Figure 15: (a) shows the transformation tree of a single room structured by evaluating the spatial relation `is-in`. (b) shows the same scene structured by additionally evaluating the `is-on` relation. (c) shows a table in close-up to illustrate how objects on the table are bound to the table's reference frame.

```
?room rdf:type office:Room
?room semap:hasObjectModel ?room_obj
?room_obj semap:hasConvexHull2D ?room_abstr_ch2D
?desk rdf:type office:DeskTable
?desk semap:hasObjectModel ?desk_obj
?desk semap:hasConvexHull2D ?desk_abstr_ch2D
?desk_abstr_ch2D semap:isIn2D ?room_abstr_l2D
==>
?room rdf:type office:Office
```

Figure 16: A rule classifying a room as an office, due to a particular type of table in it.

preter to the Apache JENA back end. Figure 16 shows a simple classification rule that uses the concepts defined in our office domain, as well as the grounded spatial relations, to specify that an instance of type `Room` is actually of type `Office`, due to the fact that it contains a specific type of table, namely a `DeskTable`. The scene in Figure 15c would qualify for this rule-based classification. This type of reasoning is used as a key component in [43].

In a similar style, other room types could be distinguished from each other based on their contents. For the scene shown in Figure 17, one could identify instances of `DesktopComputer` and `Monitor` within a narrow search radius around an instance of `Table` and group those into a new object entity of type `Workplace`. Afterwards aditional queries over the number of workspaces contained in a room could be used to differentiate between `ComputerLabs` and `Offices`.

4.4. Object Retrieval

To search for objects based on spatial and semantic criteria is an asset in many robotic activities, ranging from task planning and object manipulation to human-robot interaction. SEMAP's query system can be of help in all such applications.

Imagine the robot shown in Figure 18 (a) is asked to perform fetch-delivery tasks, e.g., to bring the operator *his* coffee mug. To solve this task, the robot is challenged to find out possible target mugs within the environment and identify

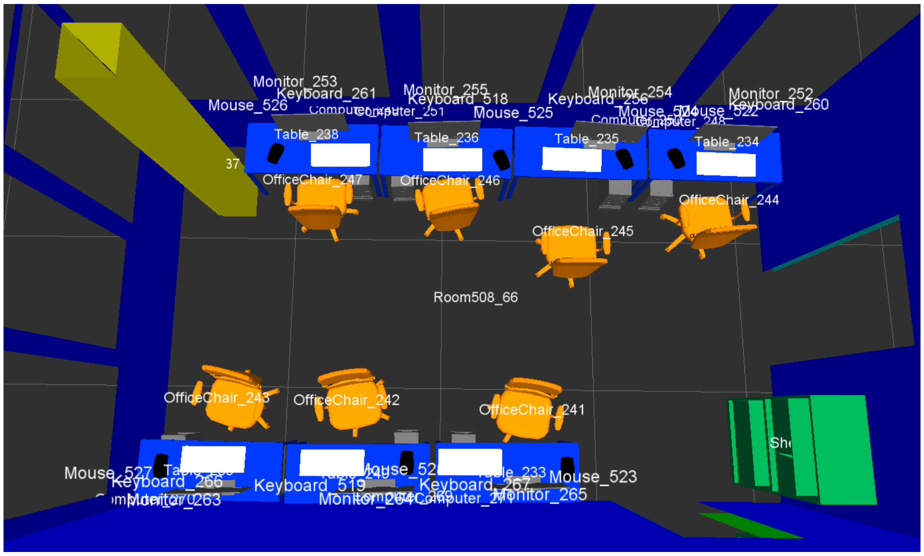
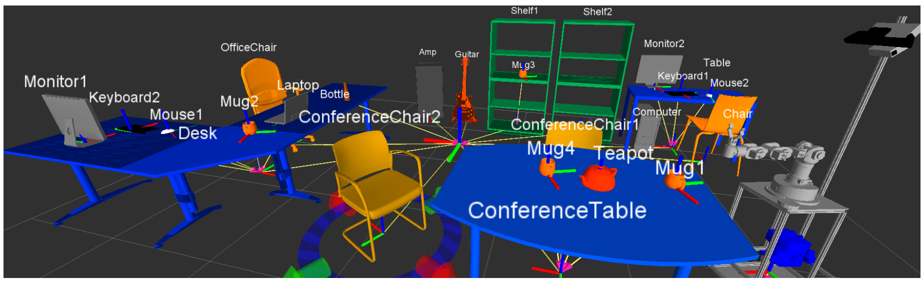


Figure 17: A SEMAP scene of a computer lab room with multiple workspaces, each consisting of a table, desktop computer and monitor.

the correct instance. Hence it has to answer various basic queries about the environment. A query like *Q1* in Figure 18 (b) will provide a set of potential targets and the rooms they are in. To narrow down this selection to the actual target, additional information is needed. However, a query about topological relations, such as *Q2*, may enable the robot to formulate natural questions, e.g., "Do you mean the mug *on* the desk, *right of* the laptop?". A likely response could be: "No, mine is *on* the `Shelf`.". This additional information allows to filter the results of *Q1* down to a single instance, namely `Mug3` and thus yields a distinct target for the robot. If the robot's next task is to serve tea, it can issue a query like *Q3* that directly identifies the most suitable target, the `Teapot`, and immediately retrieves its pose and relative geometries to guide the navigation, grasp planning, and object manipulation. Note that the latter query can be enriched with robot-dependent information, such as the maximal viable bounding volume to fit the robot's gripper, in order to extract only suitable matches.

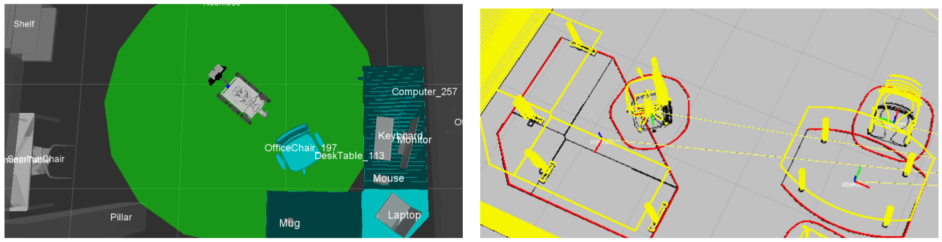


(a) An office environment modelled with the SEMAP framework.

Query / Result
Q1 Return all mugs and the objects (parts) they are on.
R1 `Mug2` on `Desk`; `Mug1` and `Mug4` on `ConferenceTable`, `Mug3` on `Shelf1-Board3`
Q2 Which relations holds for `Mug200` with respect to desk and laptop?
R2 `Mug2` *is-on* `Desk`, *right-of* `Laptop`
Q3 Return pose and geometry of a (graspable) teapot.
R3 `Teapot`, $Pose_{27}$, $RelativeGeometry_{27}$

(b) Several query types that let SEMAP support different robotic applications.

Figure 18: An exemplary office environment and questions referring to objects in it that may come up in fetch-and-delivery tasks for a service robot in such a place.



(a) Environment Awareness

(b) Robot Navigation Extraction

Figure 19: Two applications that make use of dynamic data extraction from SEMAP's environment model: (a) filters the robot's environment for relevant objects, whereas (b) extracts a map for navigation.

4.5. Environment Awareness & Dynamic Map Updates

Obviously the just-mentioned object retrieval queries are only useful if dynamic changes in the environment are continuously detected and incorporated into SEMAP's representation.

To detect dynamic changes in the environment, we proceeded as follows. We first created a module that implements environment awareness for our mobile robot and informs its object recognition module about entities that can and should be tracked. It identifies these objects of interest (e.g., tables and chairs) within a search radius around the robot, as depicted in Figure 19 (a). It uses a continuously executed range query using SEMAP's distance operator on a parameterizable set of objects. This informs the robot about the it need to track and check whether they are still present or not.

To this end, we use the currently stored object locations to navigate to the nearest object. Then we use parts of our map-building pipeline for object recognition. We hereby rely on a CAD matching approach, similar to the one presented in [43]. The necessary CAD models are provided by SEMAP directly. If the object is recognized at the given location, we use the returned estimate on the object pose, as an update to SEMAP's environment model. Once the object is updated, a spatial relation extraction query is automatically triggered to inform the knowledge base about potential changes in the environment topology. If the object is not found at the given location, we retract the entity from the environment model.

We also use the strategies presented above, to topologically re-structure the environments model after every map update, ie., when an object has moved or a new object was created. To correctly insert a new object, the run time is around 0.3 s on average, which indicates that the environment topology can be maintained with every change on our mobile robot.

Currently, we can not track the articulation of environment entities, since our perception pipeline is limited to detecting rigid objects, yet we were able to test SEMAP's ability to represent the dynamics of articulated objects by using our mobile robot itself as a test sample. We imported the robot's URDF model to create an articulated object entity within SEMAP. Next, we continuously fed the robot's pose estimate and joint states into the environment model, to align the robot's SEMAP model with the current world state. In doing so, we are able to query for spatial relations between the robot's links and the environment, i.e., we were able to infer that the robot's gripper is over a desk during the execution of an object manipulation task, as shown in Figure 20.

4.6. Navigation Map Extraction

SEMAP represents a model of the robot's environment, from which multiple applications can retrieve task-specific environment data on demand, rather then maintaining several different semantic map representations simultaneously. In this sense, SEMAP is a hybrid map, but with the additional freedom of deciding at run time which set of map representations suits the given applications best.

As an example for the extraction of task-specific maps, we implemented a module that extracts grid maps for localization and navigation from SEMAP's database. It queries the environment's absolute geometries and dissects these into multiple horizontal slices, which are then used to create a 2D projection of the environment's 3D geometry. Converted into an occupancy grid map, this projection is made compliant with the standard algorithms for robot naviga-

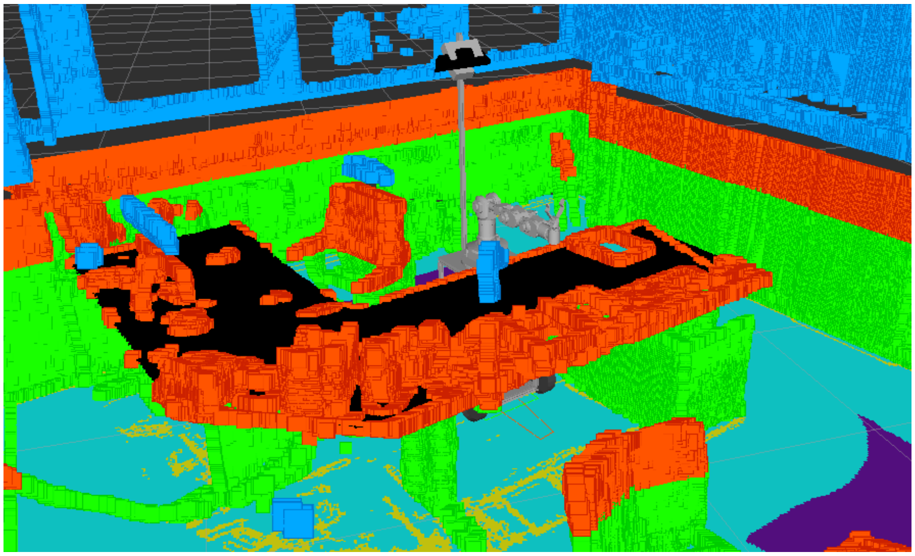


Figure 20: To query for the spatial relations between the robot and environment entities, the SEMAP model of our mobile robot is continually updated using its pose estimate and its joint states to describe its articulation state.

tion in ROS. A parameterizable set of rooms and objects is used to tailor the extraction processes to the robot-dependent demands of the application. By default, we create the navigation maps for the entire floor on which a robot operates, including all geometries along the robot's height. We also augment the grid maps to restrict the robot from areas in which it may disturb humans. For this we use SEMAP's semantic knowledge to identify all chairs and then add safety buffers to their geometries using additional PostGIS operators for applying a spatial padding. Figure 19 (b) shows the map extraction process. The horizontal slices through the environment geometry are shown in yellow, the convex hulls of the blocked objects in red, the resulting occupancy grid is shown in black. Note how the projected boundary around the chairs is larger than their spatial footprint, due to the semantically-augmented spatial padding.

By registering to dynamic changes in SEMAP's database, we avoid the inconsistencies that may result in robotic systems when 2D navigation is decoupled

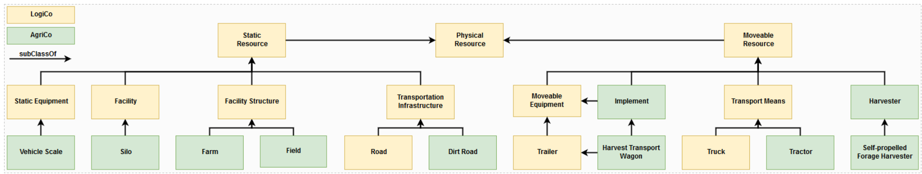


Figure 21: Excerpts of an agricultural-specific domain model added to SEMAP.

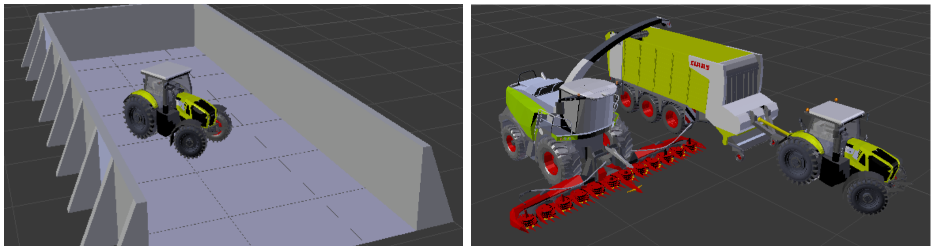
from 3D data processing. It is further possible to work with multiple instances of the map extraction module on a single SEMAP model. These can either provide multiple robots with customized maps or a support a single robot's 3D navigation using a stack of 2D grid maps, like in Figure 20. A detailed description of the map extraction module used there is presented in [48].

4.7. Changing the Application Domain

So far, all examples where confined to service robotic tasks in an office domain. To clarify that SEMAP defines a domain-independent framework for constructing environment models, we applied it in an entirely different application domain. To achieve this, SEMAP's core ontology has to be paired with a suitable ontology for the new application, such that domain-specific knowledge can be represented. The underlying representations and reasoning mechanisms remain the same.

In recent work, we applied SEMAP in an agricultural context [49]. For this, we simply replaced the office ontology used throughout, with a new domain model. Figure 21 shows this ontology, which describes entities in agricultural environments, such as fields, farms and tractors.

We generated an environment representation based on this model, by importing URDF models of agricultural machines, as well as a set of fields, represented by using polygonal boundaries and silo facilities, using 3D CAD models. Next, we use recorded telemetry data from a real harvesting campaign, to replay real machine movements between a field and a silo facility in our SEMAP model. We then used SEMAP's spatial and semantic reasoning capabilities, to detect



(a) Localization of a tractor in a silo (b) Detecting an overloading situation.

Figure 22: We used telemetry data from an actual agricultural machines to dynamically synchronize an environment model in SEMAP. Using the spatio-semantic query interface, we were able to topologically localize machines within agricultural facilites (fields, farms and silos) and to identify the correct positioning of two machines for overloading harvested goods.

spatial relations between the agricultural machines and their environment, to gain insight into the agricultural process underlying the machine activities.

For example, we continuously identified the topological relations that hold between a movable entity, such as a tractor, and the set of agricultural facilities, namely the fields, farms and silo facilities. We used the 2D position abstraction of each tractor and harvester to test for containment against the facilities 2D polygonal boundaries. We use the positive results for grounding a generic predicate `isAt`, as well as specific predicates defined in the agricultural ontology, such as `inField`, `onFarm` and `atSilo`.

We used SEMAP to reason about more complex spatial relations, too. For example, we combined several basic directional relations about a harvester and a transport vehicle, to construct the domain-specific relation of describing that both vehicles are correctly positioned for overloading harvested goods. In a situation like the one shown in Figure 22 (b), we started with a range query to detect if the transport vehicle is close enough to the harvester. If so, we tested whether the trailer is left of the harvester (or right – depending on the orientation of the overloading boom) and if the harvester's overloading boom is over the trailer. If so, the relation `positionedForOverloading` is inferred to hold between both machines.

This is valuable information about the underlying agricultural process, which was previously covert in the telemetry data of both machines, but due to SEMAP's spatio-semantic processing is now explicitly available within SEMAP's KB, where it can be used for further processing, such as rule-based reasoning and eventually for planning and controlling the agricultural machines.

5. Summary and Discussion

In this article we presented a semantic map representation framework called SEMAP that uses spatial database technology, to effectively ground qualitative spatial relations in order to make them available for knowledge-based reasoning. We extended PostGIS to support spatial queries in 3D and used its quantitative geometric analysis, to derived qualitative facts about the spatial relations of entities within an environment model. To bridge between geometric and semantic representations, we linked the entities from the geometric storage in the PostGIS database to concepts in an ontology modeled in OWL and implemented an data management and query interface that inserts these spatial predicates into a dedicated knowledge base, represented through Apache Jena, which allows for subsequent qualitative spatial reasoning on a symbolic level. To effectively realize the evaluation of geometric tests for complex geometries, we integrated suitable geometric abstractions into SEMAP's spatial model and added automatic optimizations to its querying strategies, such that time consuming tests are only executed when needed.

We presented the database schema to store static and articulated objects within the spatial database and the core ontology that is used to represent their semantic counterparts in the knowledge base. The separation between geometric core concepts and application domain in the ontology allows to use the proposed framework in different contexts. We demonstrated the basic functionalities of SEMAP in an office domain. These application examples showed, how the current implementation is able to utilize the spatial analysis capabilities in classic tasks of mobile robotics, like map building, scene classification, object retrieval and navigation. To demonstrate that the framework can be easily adapted to represent different semantic contexts, we switched to an agricultural domain model. In this application example, we used SEMAP to detect overloading positions in an harvesting process, based on recorded machine telemetry and thus provided valuable insight into a real-world application. For changing the application domain, we simply substituted the underlying domain ontology, while re-using SEMAP's core ontology and its PostGIS database for representing and querying geometric environment data.

The main drawback of the current implementation is that the linking between geometric models and qualitative knowledge has to be maintained via the query interface. Currently, we trigger all relevant updates manually to ensure that derived information from the database is inserted in the knowledge base. This is an issue concerning performance and data redundancy, and is also inconvenient during application development. To solve this problem, a formal query language that includes querying over qualitative spatial relations directly could be used and integrated into SEMAP's query system. With such a formalism, it should be possible to detect whether relations are already qualified or not to call the respective spatial operators only if needed. A candidate for such a formalism could be GeoSPARQL, which we indent to investigate in future work.

Another issue is the performance of the spatial database back end. Even though GIS technology provides spatial operators off-the-shelf, their 3D spatial representations and geometric processing lacks the efficiency required for real time processing. Although we tried to minimize the query times, some queries produce significant latency which may lead to data loss when the environment model is updated with high frequency, e.g., when telemetry information from actual machines is analyzed. A possible solution would to integreate a optimized spatial back tailored specifically for 3D data.

To improve qualitative spatial reasoning, it would be beneficial to integrate a dedicated qualitative spatial reasoning system, like SparQ [24] in addition to the geometric analysis based on PostGIS and CGAL. It will be necessary to evaluate which calculi are suitable and whether the current set of spatial operators supports the chosen calculi or not.

Currently, SEMAP can only handle a most likelihood model. It would be desirable to combine the strength of the current implementation with probabilistic methods to further enrich the stored and derived knowledge. Additionally, handling the histories of objects would be beneficial to track the positions of objects over time to support anchoring processes. These properties should be relatively easy to implement in terms of the used database, but making these information usable for knowledge based reasoning is an open issue and will definitely require to redefine the structure of our semantic mapping framework.

In spite of these conceptional and implementational issues, the general approach to integrate a spatial database into semantic maps was proven to be beneficial and the SEMAP framework provides a functional proof-of-concept. Having operators for quantitative spatial analysis readily available in the semantic map's representation helps solving the qualification problem of spatial relations and effectively supports further spatial reasoning in robotics. Placed in a processing chain where the data is pre-processed, e.g., using stream processing and probabilistic approaches, SEMAP in its current state can already solve a number of relevant problems in semantic mapping as the presented application examples clearly demonstrate.

References

[1] S. Coradeschi, A. Saffiotti, An introduction to the anchoring problem, Robotics and Autonomous Systems 43 (2) (2003) 85 – 96.

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

[3] C. Galindo, J.-A. Fernndez-Madrigal, J. Gonzlez, A. Saffiotti, Robot task planning using semantic maps, Robotics and Autonomous Systems 56 (11) (2008) 955 – 966, semantic Knowledge in Robotics.

[4] J. Wu, H. I. Christensen, J. M. Rehg, Visual place categorization: Problem, dataset, and algorithm, in: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009, pp. 4763–4770.

[5] A. Pronobis, B. Caputo, P. Jensfelt, H. Christensen, A realistic benchmark for visual indoor place recognition, Robotics and Autonomous Systems 58 (1) (2010) 81 – 96.

[6] A. Borkowski, B. Siemiatkowska, J. Szklarski, Towards semantic navigation in mobile robotics., Graph transformations and model-driven engineering 5765 (2010) 719–748.

[7] J. Crespo, R. Barber, O. M. Mozos, Relational model for robotic semantic navigation in indoor environments, Journal of Intelligent & Robotic Systems 86 (3) (2017) 617–639.

[8] E. Bastianelli, D. D. Bloisi, R. Capobianco, F. Cossu, G. Gemignani, L. Iocchi, D. Nardi, On-line semantic mapping, in: 16th International Conference on Advanced Robotics (ICAR), 2013, pp. 1–6.

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

[10] A. Pronobis, P. Jensfelt, Large-scale semantic mapping and reasoning with heterogeneous modalities, in: 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 3515–3522.

[11] D. Lang, D. Paulus, Semantic maps for robotics, in: Proc. of the Workshop Workshop on AI Robotics at ICRA, 2014.

[12] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. A. Fernandez-Madrigal, J. Gonzalez, Multi-hierarchical semantic maps for mobile robotics, in: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005, pp. 2278–2283.

[13] P. Buschka, A. Saffiotti, Some notes on the use of hybrid maps for mobile robots, in: Proc. of the 8th Int. Conf. on Intelligent Autonomous Systems, 2004, pp. 547–556.

[14] R. Capobianco, G. Gemignani, D. Bloisi, D. Nardi, L. Iocchi, Automatic extraction of structural representations of environments, in: Intelligent Autonomous Systems 13, Springer, 2014, pp. 721–733.

[15] A. Bouguerra, L. Karlsson, A. Saffiotti, Monitoring the execution of robot plans using semantic knowledge, Robotics and autonomous systems 56 (11) (2008) 942–954.

[16] F. Baader, The description logic handbook: Theory, implementation and applications, Cambridge university press, 2003.

[17] M. Tenorth, M. Beetz, Knowrob – a knowledge processing infrastructure for cognition-enabled robots, International Journal of Robotics Research 32 (5) (2013) 566 – 590.

[18] D. Pangercic, B. Pitzer, M. Tenorth, M. Beetz, Semantic object maps for robotic housework - representation, acquisition and use, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012, pp. 4644–4651.

[19] M. Waibel, M. Beetz, J. Civera, R. D Andrea, J. Elfring, D. Galvez-Lopez, K. Haussermann, R. Janssen, J. Montiel, A. Perzylo, et al., A world wide web for robots, IEEE Robotics & Automation Magazine 18 (2) (2011) 69–82.

[20] M. Beetz, F. Blint-Benczdi, N. Blodow, D. Nyga, T. Wiedemeyer, Z. C. Mrton, Robosherlock: Unstructured information processing for robot perception, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 1549–1556.

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

[22] J. F. Allen, An interval-based representation of temporal knowledge, in: Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 1, IJCAI'81, Morgan Kaufmann Publishers Inc., 1981, pp. 221–226.

[23] A. G. Cohn, B. Bennett, J. Gooday, N. M. Gotts, Qualitative spatial representation and reasoning with the region connection calculus, GeoInformatica 1 (3) (1997) 275–316.

[24] D. Wolter, J. Wallgrün, Qualitative spatial reasoning for applications: New challenges and the sparq toolbox, Qualitative Spatio-Temporal Representation and Reasoning: Trends and Future Directions (2012) 336–362.

[25] K. Sjöö, A. Pronobis, P. Jensfelt, Functional topological relations for qualitative spatial representation, in: 2011 15th International Conference on Advanced Robotics (ICAR), 2011, pp. 130–136.

[26] C. Landsiedel, V. Rieser, M. Walter, D. Wollherr, A review of spatial reasoning and interaction for real-world robotics, Advanced Robotics 31 (5) (2017) 222–242.

[27] A. Guttman, R-trees: A dynamic index structure for spatial searching, in: Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data, SIGMOD '84, ACM, 1984, pp. 47–57.

[28] M. J. Egenhofer, R. D. Franzosa, Point-set topological spatial relations, International Journal of Geographical Information System 5 (2).

[29] E. Clementini, P. D. Felice, P. v. Oosterom, A small set of formal topological relationships suitable for end-user interaction, in: Proceedings of the Third International Symposium on Advances in Spatial Databases, SSD '93, Springer, 1993, pp. 277–295.

[30] Simple Feature Access Specification web, `http://www.opengeospatial.org/standards/sfs` (2015).

[31] S. Zlatanova, A. A. Rahman, W. Shi, Topological models and frameworks for 3D spatial objects, Computers & Geosciences 30 (4) (2004) 419–428.

[32] A. Borrmann, E. Rank, Query support for bims using semantic and spatial conditions, Handbook of Research on Building Information Modeling and Construction Informatics: Concepts and Technologies: Concepts and Technologies (2009) 405.

[33] S. Daum, A. Borrmann, Processing of Topological BIM Queries using Boundary Representation Based Methods, Advanced Engineering Informatics.

[34] J. Renz, B. Nebel, I. Pratt-Hartmann, Qualitative Spatial Reasoning Using Constraint Calculi, Handbook of Spatial Logics.

[35] A. Borrmann, E. Rank, Specification and implementation of directional operators in a 3D spatial query language for building information models, Advanced Engineering Informatics 23.

[36] A. Abdul-Rahman, M. Pilouk, Spatial data modelling for 3D GIS, Springer Science & Business Media, 2007.

[37] M. Breunig, S. Zlatanova, 3D geo-database research: Retrospective and future directions, Computers and Geosciences 37 (7) (2011) 791–803.

[38] PostGIS Website, `http://postgis.com/` (2015).

[39] GEOS library, `http://trac.osgeo.org/geos/` (2015).

[40] SFCGAL Project Website, `http://www.sfcgal.org/` (2015).

[41] CGAL library, `http://cgal.com/documentation` (2015).

[42] A. Borrmann, E. Rank, Topological operators in a 3d spatial query language for building information models, in: In Proc. of the 12th Int. Conf. on Computing in Civil and Building Engineering (ICCCBE), 2008.

[43] M. Günther, T. Wiemann, S. Albrecht, J. Hertzberg, Model-based furniture recognition for building semantic object maps, Artificial Intelligence 247 (2017) 336 – 351.

[44] M. Tenorth, S. Profanter, F. Balint-Benczedi, M. Beetz, Decomposing CAD models of objects of daily use and reasoning about their functional parts, IEEE International Conference on Intelligent Robots and Systems.

[45] T. Wiemann, K. Lingemann, A. Nüchter, J. Hertzberg, A toolkit for automatic generation of polygonal maps – las vegas reconstruction, in: Proc. ROBOTIK 2012, VDE Verlag, 2012.

[46] P. K. Nathan Silberman, Derek Hoiem, R. Fergus, Indoor segmentation and support inference from rgbd images, in: ECCV, 2012.

[47] J. R. Ruiz-Sarmiento, C. Galindo, J. González-Jiménez, Robot@home, a robotic dataset for semantic mapping of home environments, International Journal of Robotics Research.

[48] H. Deeken, S. Puetz, T. Wiemann, K. Lingemann, J. Hertzberg, Integrating semantic information in navigational planning, in: ISR/Robotik 2014; 41st International Symposium on Robotics; Proceedings of, VDE, 2014, pp. 1–8.

[49] H. Deeken, T. Wiemann, J. Hertzberg, A spatio-semantic model for agricultural environments and machines, in: The 31st International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems (IEA-AIE 2018), Lecture Notes in Artificial Intelligence, 2018, submitted to.