reMap: Spatially-grounded and Queryable Semantics for Interactive Robots

Lorenzo Ferrini[1,2], Jozsef Palmieri[3]
Alessandro Marino[3], Dongheui Lee[2], and Séverin Lemaignan[1]

[1] PAL Robotics, Barcelona,
firstname.lastname@pal-robotics.com
[2] Institute of Computer Technology, TUW, Wien,
dongheui.lee@tuwien.ac.at
[3] Università degli studi di Cassino e del Lazio Meridionale, Cassino,
firstname.lastname@unicas.it

Abstract. The semantic information available to a robot enhances its understanding of the world and allows it to adapt its behaviour accordingly. This information can be spatially-grounded, meaning it is associated with specific areas of the environment. For a robot to use this information effectively during its tasks, it is crucial to provide an efficient system for storing and retrieving spatially-grounded semantics. In this paper, we present reMap, a novel framework for the efficient representation, storage, and retrieval of spatially-grounded semantics. In reMap, we formally introduce Representation Maps (RMs), three-dimensional functions that each represent a different type of semantic information in space. These structures can be combined through operators to extract additional spatially grounded semantic information. reMap includes a SPARQL-based language that serves as a programmatic interface for retrieving spatially-grounded semantics stored in RMs. We provide an open-source ROS-based implementation of reMap, enabling efficient three-dimensional information storage and processing using dense voxel maps based on the high-performance OpenVDB format. Finally, we describe the execution of the framework over real-world data recorded in a semantically rich real-world environment.

Keywords: Semantic mapping, Interactive robots, Human-Robot Interaction

1 Introduction

Semantic information is crucial in allowing robots to understand the world around them [10]. Robots can use semantic information for a vast range of tasks, i.e, for all those where understanding the *meaning* behind the components involved is the key for a successful execution [4, 15].

In social robotics, semantic information allows robots to understand social signals and adapt their behaviour accordingly. In fact, advancements in machine

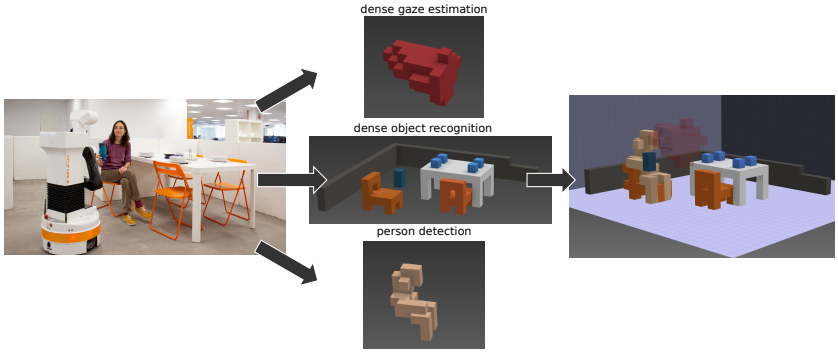

Fig. 1: The reMap spatio-semantic framework combines spatially-grounded semantic sources into a dense voxel map. In this example, three sources are combined: the human field-of-view map, the object map, and the personal space map. The resulting map can be queried using a SPARQL-based language.

learning have enabled robots to real-time estimate aspects such as human emotions [26], group dynamics [13], and engagement [25]. The semantics extracted can be used to manage the various aspects in the human-robot interaction spectrum, from maintaining appropriate personal space [9] to conversation management [12].

A part of these semantic data is linked to specific locations in space, creating a direct connection between the information and a volume. However, the spatial anchoring of these data is not necessarily fixed; while some elements may have stable characteristics in the environment, others undergo frequent changes in position, shape, and value. These dynamic attributes present challenges for representation due to the intricate and evolving nature of the data, the need for frequent updates requiring significant computational resources, and its inherently unstructured nature that makes it difficult to create concise symbolic representations. Tackling these challenges requires a specialised framework that provides efficient and standardised data structures designed to encode spatially anchored semantic information, along with APIs optimised for rapid access and modification of values. Despite the enhanced semantic capabilities of robots enabled by advances in areas such as Deep Learning, current frameworks lack comprehensive support for efficient representation of spatially-grounded semantic data.

Moreover, the semantic information available to interactive robots is rarely standalone. It often requires processing and integration, and the robot reasoning over it to derive further semantic insights. Therefore, a complete framework for spatially grounded semantic information should also include structures that facilitate semantic reasoning on the data.

This paper seeks to tackle these aspects by introducing ***reMap***, a comprehensive open-source framework for spatially anchored semantic information in robotics, incorporating structures that support semantic reasoning and flexible representation of dynamic semantic data.

Our contributions include:

1. A framework based on 3D voxel maps (Fig. 1), to represent and efficiently retrieve semantic information from unstructured – yet spatially-grounded – data for use in robotic tasks;
2. The definition of operators for combining the stored maps and extracting new spatially-grounded semantic information from them.
3. a proof-of-concept SPARQL-like query language for semantic information retrieval from maps;
4. an open-source ROS-based implementation of the presented framework[4].

2 Related Work

Anchoring semantic information in the spatial environment is central to many robotics tasks, and remains an on-going challenge [10, 15]. Various strategies have been explored, often tailored to specific application domains. For instance, in navigation, numerous solutions have been proposed to effectively correlate semantic concepts with physical space [4]. In [8], authors define a multi-hierarchical structure to link semantic concepts with navigation maps, introducing a querying language for information retrieval pertinent to navigation tasks. The authors in [3] focus on the real-time grounding of symbolic information onto 2D maps for life-long information acquisition and representation. In [23], the authors pursue an ontology-based approach [27], focusing on the representation and acquisition of comprehensive semantic maps, with the goal of facilitating prolog-based querying systems for information retrieval in navigation tasks. In [6], the authors propose a framework for managing semantic maps, employing a client-server architecture equipped with custom plugins for updating grounded semantic information. While these projects have laid the foundations for the spatial grounding of semantic data, they do not directly address neither dense, highly dynamic spatial data, often found in human-robot interaction situations, nor continuous data fields (akin to *mightability maps* [22]).

Representation of spatial semantic data has also been a research subject in the Human-Robot Interaction (HRI) domain. In [17], authors define a perspective-taking approach aiming at solving ambiguities in grounding speech interactions between humans and robots. Here, they use an ontology-based [16] approach to associate objects in space with semantic concepts. The semantics in this case might refer to the object class or to the spatial relationships with the other objects in the scene. In [11], the authors implement a system aiming at the grounding of target objects in the scene based on human-robot speech interaction. They combine 2D and 3D features to speech detection to extract actions to perform with the robot targeting specific objects in space. A similar goal motivates the work in [24], where the authors propose INGRESS, a deep learning-based approach for unconstrained matching of spatially-grounded expressions and objects in human-robot verbal interaction.

[4] `https://github.com/RepresentationMaps`

	Vimantic [6]	*SOM+ [23]*	*SPARK [?]*	*reMap*
KB Integration	pull	pull	push	SPARQL API
Human Modelling	No	No	Yes	Plugin
3D Dense Fields	No	No	No	Yes
Architecture	Modular	Monolithic	Modular	Modular
Spatial Reasoning	No	Yes	Yes	Yes
Spatial API	No	No	No	Yes
Data Structure	Point cloud	Point cloud	CAD meshes	OpenVDB

Table 1: Comparison of various frameworks for spatially grounded semantics representation, highlighting their integration with knowledge bases (KB), support for human and 3D field modelling, architectural design, spatial reasoning capabilities, availability of an API to retrieve spatially localised volumes, and underlying data structures.

While these approaches prove to be effective for object-based contexts, they also leave some open challenges. One challenge is the need to scale these methods to accommodate newly-defined, custom semantic features, a capability not currently supported. Additionally, there is the need to structure these approaches for free-form representation of spatially-grounded semantic information, as they are presently tied to RGBD data or pre-acquired object meshes. Recently, there have been advances in the field of natural language-based spatial reasoning through large language models (LLMs) [2, 14]. In [5], authors present a novel methodology for zero-shot object-based navigation. Here, they combine LLMs and vision language models (VLMs) to generate sequential navigation decisions, based both on an explicit user request and the current state of the environment. The VLM is applied for grounding the semantic concepts expressed in the request into the detected objects in space. While LLMs and VLMs present promising results in reasoning and show zero-shot abilities in grounding semantics in space, they also suffer some limitations. In fact, they are not able to spatially represent unstructured, field-like semantic information with exact location in space. Moreover, they might suffer the definition of application-specific semantics that do not represent any type of traditional common knowledge.

With reMap, we aim at addressing the gaps in representation capabilities from previous works highlighted in this section. A comparison between the features presented in the implementation of our framework and those from previous works are presented in Table 1. In Section 6, we briefly discuss as future work the possible combination of this work with LLMs to directly query reMap using natural language.

3 Semantic Information in Space

Representing spatially grounded information in space means associating semantic labels to regions of space in the robot's vicinity. This representation can be

associated both to discrete phenomenons (e.g., associating the object class id to the areas occupied by the object itself) or to continuous ones (e.g., expressing the field of view of as a function of the distance from its gaze central axis). In this section, we formally introduce our framework for 3D-grounded semantic data representation and processing.

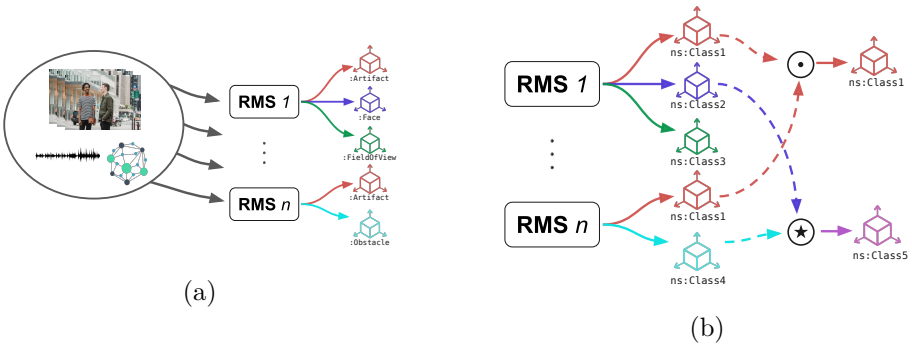

Fig. 2: RMSs build RMs from the percepts available to the robot. Each RMS might output multiple RMs, each representing different spatial information from the scene, and with an explicit data class attached (a). Operators (like $\odot$ and $\circledast$, representing generic operators) can combine several maps to output new ones, of the same type (e.g. in (b), $\odot$) or not ($\circledast$).

3.1 Conceptual Description of the Framework

3.1.1 Data representation and processing In the proposed model, we introduce the concepts of ***Representation Maps*** (RMs) and ***Representation Maps Sources*** (RMSs). The former are spatially-grounded representations of semantic information, that is, they map 3-dimensional world coordinates to values associated to specific semantics (e.g., the object class associated with the detected objects). They can be represented as functions m. Each m belongs to a specific functional $M = \{\mathbf{R}^3 \Rightarrow \mathbf{R}\}$, associated with the semantics of the information represented in m. To each map m is also associated additional explicit information regarding the semantics of values in m (see Section 3.1.2).

RMSs are themselves functions; their role is to transform information available to the robot at a given time into RMs. RMSs belong to the functional $P = \{\mathcal{P}(S) \rightarrow M^n\}$, where S is the set of information available to a robot at a given time and $\mathcal{P}(S)$ is its powerset (Fig. 2a); n depends on each RMS and represents the number of RMs each RMS can output. The value of n can change over time.

3.1.2 Semantics Each map represents data with specific semantics. Critically, each map is semantically homogeneous: its data represents a single type

of information. We rely on the Resource Description Format (RDF) terminology and formalism to formally assign the semantics of the data contained in each map. We do so by attaching an RDF *class* to the map (for instance, relying on the OpenRobot Ontology [16] semantics, we use the class `oro:Artifact` for maps representing objects in space). This semantic information is used by map operators (see below) to reason and combine maps in a semantic-aware manner (Fig. 2b).

3.1.3 Combining RMs with operators RMs can be combined by operators. An operator is a function that combines two or more RMs into a new RM. The input RMs might or might not be associated to the same class. The output RM might belong to the same class as one of the input RMs or to a new one. Given k RMs $m_1, \ldots, m_k$, an operator on these RMs is defined as $f_{1,\ldots,k} := m_1, \ldots, m_k \rightarrow m_{res}$.

An example of operator over same-class RMs is the one combining all the object-detection RMs into a single one. For instance, different RMSs might generate maps containing information of the detected objects in the scene, where each RMSs runs a differently specialised object detection algorithm (YOLO [28], RGBD-based table detecion, etc.). For a global view on the objects detected in the scene and for further semantic processing, we are interested in the combination of these different RMs. Therefore, we define the object-detection RMs combing operator. Given j object-detection RMs m_{o_k}, $k = 1, .., j$, with associated class `oro:Artifact`, this operator is defined as $f_o(m_{o_1}, m_{o_2}, ..., m_{o_j}) = m_{o_1} \cup m_{o_2} \cup ... \cup m_{o_j}$. The $\cup$ operation is the union between the different functions.

Once obtained the full object-detection RM, we might be interested in understanding which of these objects are in the FoV of an agent. This information can be obtained by defining the operator $f_{of}\,(m_o, m_f)$; this takes as input an RM with class `Artifact` (m_o) and one with class `FieldOfView` (m_f) and outputs a new RM containing only those voxels of m_o that happen to be in the represented field of view in m_f (that is, m_f acting as a boolean mask).

4 Operators as semantic queries

Practically, operators can be seen as transformations over the available RMs. These transformations can be described as SQL-like queries over the available semantic data. To this end, we have developed a simple query language whose grammar is based on the RDF graph query language SPARQL[5]. The language is based on (semantic) pattern matching, where the user defines a set of constraints that the RMs must satisfy. The query is then executed over the available RMs, and a new RM, simultaneously satisfying all the constraints, is returned. As such, the query is the declarative specification of an operator.

For example, the following query retrieves the voxels containing a *mug* in the field of view of a human `human1` – it is a possible implementation of an *objectInFieldOfView* operator:

[5] `https://www.w3.org/TR/sparql11-overview/`

```
PREFIX : http://kb.openrobots.org/
SELECT ?voxels
WHERE
?voxels :inFieldOfView human1 .
?voxels :containsObject 'mug'
STORE
:containsObject
```

The `STORE` clause is an extension indicating the datatype of the resulting RM. In this case, the query returns a RM where each voxel contains the class label of the objects in the selected voxels.

In practice, to actually implement a prototype of this query language, each RM m is attached to exactly one RDF property p. We denote $D(p)$ the *domain* of the property, and $R(p)$ its range. For instance, the property `oro:containsObject` (which associates an object to its COCO-80 [19] class) has `oro:Artifact` as domain. In Section 5.5 we present a first implementation of the query execution system.

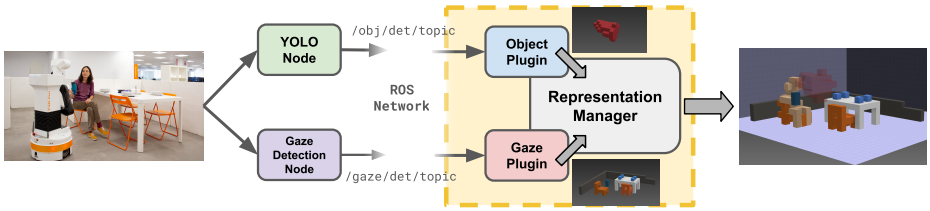

Fig. 3: The reMap framework (boarded in yellow) interacting with non-reMap ROS components. The plugins generate RMs with information provided by the non-reMap nodes. These RMs are then handled and exposed to queries by `representation_manager`.

5 Implementation

Given the interactive robot-oriented nature of the proposed model, a software implementation is required to prove its capacity to model semantic information in the scenario around a robot. Considering the limited computational resources available to robots and that these usually have to sustain several other computation-heavy processes (e.g., navigation stack), it is required for the implementation to be lightweight. Ideally, the model should reach real-time performance, so that the robot's response based on the the values in RMs is not negatively impacted.

We opted for a ROS-based implementation of reMap. At this point, ROS is a highly mature framework for robot programming, coming with all the tools

required to achieve the aforementioned desired performances. The architecture built to implement the model is plugin-based. Plugins were implemented through ROS `pluginlib` C++ library; we developed two different open source ROS packages:

- `rep_map`: a package implementing the API and the basic tools required to manage the output of the executing RMSs, that is, the active representation plugins. This includes:
  - the interfaces for the definition of inter-maps operators;
  - a query API, presented in Section 4;
  - `representation_manager`, a ROS node for handling the multiple maps generated by the different RMS and the input queries.
- `rep_plugins`: a package providing the basic interfaces and tools to implement RMSs as plugins. It comes with a group of ready-made plugins.

In Fig. 3, we graphically highlight the reMap ROS components (plugins, `representation_manager`) within a generic ROS architecture.

5.1 rep_map

This package provides the software required to manage the plugins activation (that is, the software implementing a specific RMS), as well as their de-activation at runtime. In this way, the robot can dynamically adapt to the semantic context and always run only the required representation processes.

The package provides the required interfaces to define same-domain and inter-domain operators. Some operators are already defined, for instance the operator to combine object-detection RMs into a comprehensive one. A ROS node (`representation_manager`) is part of the package, instantiating the required ROS structures for communication between the plugins and the other ROS nodes part of the robot infrastructures. The package also incorporates utilities for RMs visualisation and debugging.

5.2 rep_plugins

The `rep_plugins` package provides the required structures to develop the semantic plugins (that is, the RMS software implementation) and the API to interact with them. It requires every plugin to be implemented as a class inheriting from `semantic_plugin_base`. This class already provides the required function to set values for specific areas and shapes of the space in RMs. Each plugin can access the information available to the robot through the ROS communication interface. The package comes with a limited number of plugins already implemented.

5.2.1 Person detection plugin This plugin outputs a single RM where the contained voxels represent the space portions occupied by people. It relies on YOLOv8 [28] image segmentation results, mapping each person's mask to space using an RGB-alinged depth image of the environment. The plugin does not directly perform image segmentation; this task is performed by an independent node, publishing the results as a custom message.

5.2.2 Gaze detection and FoV plugin This plugin handles the semantic information associated with the gaze direction of agents. In our implementation, detected n agents, it outputs n RMs. Each one of them is a boolean one, representing the field of view of each detected person. Each map might also be associated with a specific person, according to the face recognition pipeline.

This plugin does infer the agents gaze direction: it retrieves information about the agents gaze through the ROS4HRI [20] tools and utilities. In ROS4HRI, every gaze is associated with a `gaze_<face_id>` TF frame. The z axis of this frame points to the agent's estimated gaze direction. Accordingly, the plugin represents the region of space currently looked at by the agent as a cone.

We set the aperture of the field of each gaze cone to $\theta = 0.5rad$. The equation for each gaze cone in the output RMs is:

$$m(x) = \begin{cases} true & \text{for } x \text{ inside the cone} \\ false & \text{elsewhere} \end{cases} \tag{1}$$

5.2.3 Object detection plugin This plugin handles the semantic information related to the objects detected in the scene. Every object might be associated with various types of semantic information. In this case, we focused on the object class, expressed as an integer value.

As for person detection, this plugin does not directly perform object detection, but subscribes to a topic where information on the detected objects in the scene are published in the form of segmentation masks. Then, as in person detection, the plugin uses the RGB-alinged depth image to generate an RM where voxels are associated to pixels from the original image; here, each activated voxel contain an integer value representing the object-class of the associated object.

5.3 RMs implementation

The framework described in 3 requires, for effective implementation, a lightweight and efficient structure to represent 3D information. We opted to represent each RM as an independent VDB. VDBs are dynamic structures that use a sparse hierarchical data structure to efficiently store volumetric data based on voxels. This structure allows for memory-efficient representation of 3D grids, where only voxels with nonempty values are stored, optimising both storage and computational performance. In this case, we used the OpenVDB [21] library, which is the *de facto* standard for python and C++ implementation of VDBs. Each RM is

represented as a `Grid<T>` object, where `T` specifies the type of the value associated with the grid voxels. Therefore, faces RMs are `Grid<float>`, object RMs are `Grid<int>` and FOV maps are `Grid<bool>`.

In OpenVDB, each `Grid<T>` object contains a `Transform` object. This is continuously updated with the transform between the original data frame and a reference frame, which is shared among the whole architecture. This way, the corresponding points in the index space of the various `Grid` objects can map to the same real-world coordinates.

The OpenVDB API provides the required functions and interfaces for RMs representation and processing, including for the implementation of the operators described in Sec.3.1.3.

5.4 Demonstration on real-world data

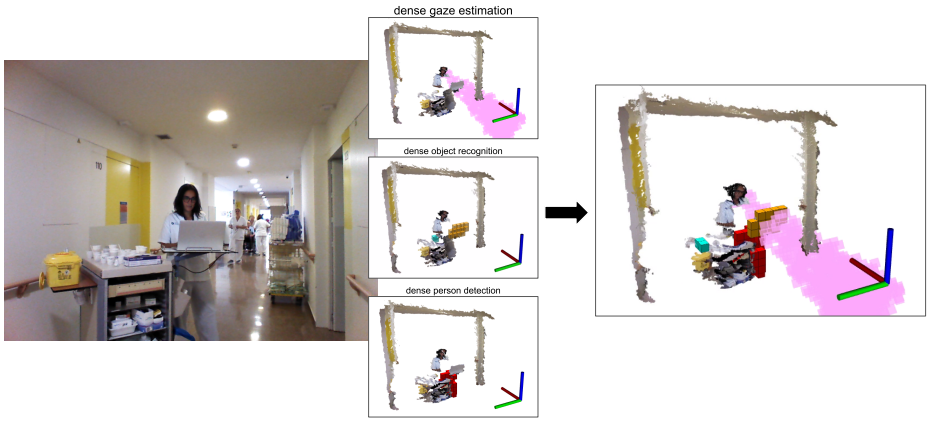

Fig. 4: RMs generation from real-world data acquired in an hospital. For reference, we overlay the perceived point cloud over the voxel maps.

To validate the implementation of `rep_map` and `rep_plugins`, we tested the framework to generate RMs on a set of pre-recorded bag files through the previously described plugins (Fig. 4).

The bag files were recorded during a 2-month deployment in a hospital in the metropolitan area of Barcelona and depict a corridor in one of the rehabilitation wards. This environment is representative of complex human environments where we expect social robots to have impact. During the recording session, performed using a PAL Robotics TIAGo robot, both nurses and patients were present in the area. For privacy reasons, we will not disclose the bag files recorded to test the framework.

Each plugin generates its RMs at the same pace as their input stream: for instance, when running YOLOv8 largest pre-trained model (`yolov8x`) on the

available hardware, the YOLO dedicated node outputs results at $5Hz$; the face detection pipeline, directly running on the robot, outputs results at $20Hz$; for this reason, reMap allows for asynchronous generation and combination (through operators) of different RMs. In this demonstration, reMap shows real-time capabilities in handling the RMs generated by the different plugins.

5.5 Proof-of-concept implementation of the query language

```
PREFIX : http://kb.openrobots.org/
SELECT ?voxels
WHERE
?voxels :containsObject *
STORE
:containsObject
```

```
PREFIX : http://kb.openrobots.org/
SELECT ?voxels
WHERE
?voxels :inFieldOfView human1 .
?voxels :containsObject *
STORE
:containsObject
```

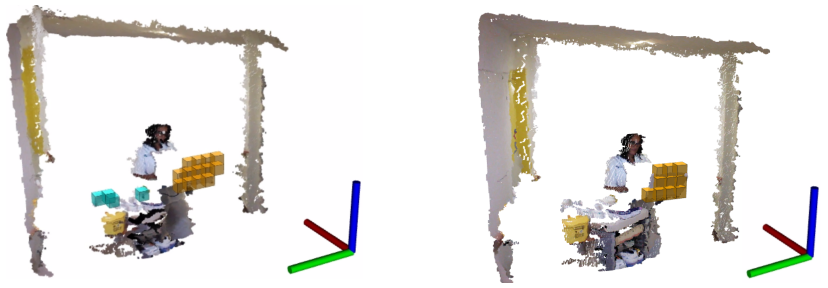

Fig. 5: Two examples of reMap query execution. On the left, a query extracting all the objects detected in the scene. On the right, combining object information and the detected field of view of the person in the scene, we extract information on the object looked-at by the nurse.

We have implemented a first version of the query language described in Section 4. This includes a simple algorithm for query parsing, RMs selection, operator definition and execution.

The query parsing disassembles the queries and selects the target domain (expressed by the `STORE` extension) and the RMs involved in the operator executions. This is possible by matching the properties from the query with those associated to the RMs, as specified in Section 3.1.2. The API introduced in the previous subsections enables the actual processing of the query as operator. At runtime, the system exposes a ROS service for queries. In case of successful execution, a new RM containing the query result is created. In case of non-matching attributes in the query, the system aborts the query execution and communicates it to the requester as a result of the service execution. Two examples of query-based data extraction are reported in Fig. 5. The reported queries have been executed over the real-world data collected for validation purposes.

6 Discussion

6.1 Query language

Although our query language demonstrates effectiveness in a range of scenarios, it remains at an early stage of development. Future iterations should aim to incorporate features akin to those found in SPARQL (including, for instance, the implementation of the `FILTER` clause), enabling users to formulate more intricate queries with ease. We also plan to formally examine the mapping of SPARQL semantics to spatial data maps.

6.2 Future work

6.2.1 Spatial queries from natural language While query languages provide an effective programmatic interface to retrieve (spatially-grounded) semantic information, they do not automatically fit HRI scenarios with humans communicating with natural language. Therefore, what is required for the deployment of the querying system in HRI scenarios is the automatic translation of natural language utterances into queries based on the aforementioned language. In this sense, LLMs have proved to be a good fit for the adaptation and translation of natural language to query languages [7, 29, 30]. In this context, we plan to adapt our framework to the extraction of spatially-grounded semantic information from natural language using GPT-4 [1].

6.2.2 Time modelling While reMap can ground semantics in 3-dimensional space, it does not address temporal aspects. This is a feature available in previous world-modelling works in HRI [18]. We plan to extend reMap to include a time-consistent modelling of space. This will enable the system to extend its modelling capabilities beyond the immediately observable scenario.

7 Conclusion

In this work we introduced *reMap*, a novel framework for the representation, processing and querying of spatially-grounded semantic information in robotics. Firstly, we defined the theoretical concepts behind the framework, introducing the concepts of *Representation Maps*, *Representation Map Sources* and *Maps Operators*. We also explained how it is possible to define a SPARQL-based querying system to extract spatial information from *Representation Maps*.

We then proceeded illustrating a ROS-based implementation of the conceptual framework introduced in the previous chapters, with the description of `rep_map` and `rep_plugin`. The described reMap implementation is publicly available as an open-source package; the link is provided in previous sections.

Finally, we tested the information representation capabilities of the implemented solution over real-world data, reporting the processing performance and the visual representation of queries performed over the generated maps.

Acknowledgments

This work has been funded by the H2020 PERSEO project (no. 955778).

References

1. Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F.L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al.: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
2. Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Fu, C., Gopalakrishnan, K., Hausman, K., et al.: Do as i can, not as i say: Grounding language in robotic affordances. arXiv preprint arXiv:2204.01691 (2022)
3. Bastianelli, E., Bloisi, D.D., Capobianco, R., Cossu, F., Gemignani, G., Iocchi, L., Nardi, D.: On-line semantic mapping. In: 2013 16th International Conference on Advanced Robotics (ICAR). pp. 1–6. IEEE (2013)
4. Crespo, J., Castillo, J.C., Mozos, O.M., Barber, R.: Semantic information for robot navigation: A survey. Applied Sciences **10**(2), 497 (2020)
5. Dorbala, V.S., Mullen Jr, J.F., Manocha, D.: Can an embodied agent find your cat-shaped mug? llm-based zero-shot object navigation. IEEE Robotics and Automation Letters (2023)
6. Fernández-Chaves, D., Ruiz-Sarmiento, J.R., Petkov, N., Gonzalez-Jimenez, J.: Vimantic, a distributed robotic architecture for semantic mapping in indoor environments. Knowledge-Based Systems **232**, 107440 (2021)
7. Futia, G., Vetro, A., Melandri, A., De Martin, J.C.: Training neural language models with sparql queries for semi-automatic semantic mapping. Procedia Computer Science **137**, 187–198 (2018)
8. Galindo, C., Saffiotti, A., Coradeschi, S., Buschka, P., Fernandez-Madrigal, J.A., González, J.: Multi-hierarchical semantic maps for mobile robotics. In: 2005 IEEE/RSJ international conference on intelligent robots and systems. pp. 2278–2283. IEEE (2005)
9. Gao, Y., Huang, C.M.: Evaluation of socially-aware robot navigation. Frontiers in Robotics and AI **8**, 721317 (2022)
10. Garg, S., Sünderhauf, N., Dayoub, F., Morrison, D., Cosgun, A., Carneiro, G., Wu, Q., Chin, T.J., Reid, I., Gould, S., et al.: Semantics for robotic mapping, perception and interaction: A survey. Foundations and Trends® in Robotics **8**(1–2), 1–224 (2020)
11. Guadarrama, S., Riano, L., Golland, D., Go, D., Jia, Y., Klein, D., Abbeel, P., Darrell, T., et al.: Grounding spatial relations for human-robot interaction. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 1640–1647. IEEE (2013)
12. Hanschmann, L., Gnewuch, U., Maedche, A.: Saleshat: A llm-based social robot for human-like sales conversations. In: International Workshop on Chatbot Research and Design. pp. 61–76. Springer (2023)
13. Hedayati, H., Muehlbradt, A., Szafir, D.J., Andrist, S.: Reform: Recognizing f-formations for social robots. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 11181–11188. IEEE (2020)
14. Huang, W., Abbeel, P., Pathak, D., Mordatch, I.: Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In: International Conference on Machine Learning. pp. 9118–9147. PMLR (2022)

15. Kleeberger, K., Bormann, R., Kraus, W., Huber, M.F.: A survey on learning-based robotic grasping. Current Robotics Reports **1**, 239–249 (2020)
16. Lemaignan, S., Ros, R., Mösenlechner, L., Alami, R., Beetz, M.: Oro, a knowledge management platform for cognitive architectures in robotics. In: 2010 IEEE/RSJ International conference on intelligent robots and systems. pp. 3548–3553. IEEE (2010). https://doi.org/10.1109/IROS.2010.5649547
17. Lemaignan, S., Ros, R., Sisbot, E.A., Alami, R., Beetz, M.: Grounding the interaction: Anchoring situated discourse in everyday human-robot interaction. International Journal of Social Robotics **4**, 181–199 (2012)
18. Lemaignan, S., Sallami, Y., Wallbridge, C., Clodic, A., Belpaeme, T., Alami, R.: Underworlds: Cascading situation assessment for robots. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 7750–7757. IEEE (2018)
19. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13. pp. 740–755. Springer (2014)
20. Mohamed, Y., Lemaignan, S.: Ros for human-robot interaction. In: 2021 IEEE/RSJ international conference on intelligent robots and systems (IROS). pp. 3020–3027. IEEE (2021)
21. Museth, K., Lait, J., Johanson, J., Budsberg, J., Henderson, R., Alden, M., Cucka, P., Hill, D., Pearce, A.: Openvdb: an open-source data structure and toolkit for high-resolution volumes. In: Acm siggraph 2013 courses, pp. 1–1 (2013)
22. Pandey, A.K., Alami, R.: Mightability maps: A perceptual level decisional framework for co-operative and competitive human-robot interaction. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 5842–5848. IEEE (2010)
23. Pangercic, D., Pitzer, B., Tenorth, M., Beetz, M.: Semantic object maps for robotic housework-representation, acquisition and use. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4644–4651. IEEE (2012)
24. Shridhar, M., Mittal, D., Hsu, D.: Ingress: Interactive visual grounding of referring expressions. The International Journal of Robotics Research **39**(2-3), 217–232 (2020)
25. Sorrentino, A., Fiorini, L., Cavallo, F.: From the definition to the automatic assessment of engagement in human–robot interaction: A systematic review. International Journal of Social Robotics pp. 1–23 (2024)
26. Spezialetti, M., Placidi, G., Rossi, S.: Emotion recognition for human-robot interaction: Recent advances and future perspectives. Frontiers in Robotics and AI **7**, 532279 (2020)
27. Tenorth, M., Beetz, M.: Knowrobknowledge processing for autonomous personal robots. In: 2009 IEEE/RSJ international conference on intelligent robots and systems. pp. 4261–4266. IEEE (2009)
28. Terven, J., Cordova-Esparza, D.: A comprehensive review of yolo: From yolov1 to yolov8 and beyond. arXiv preprint arXiv:2304.00501 (2023)
29. Trummer, I.: Codexdb: Synthesizing code for query processing from natural language instructions using gpt-3 codex. Proceedings of the VLDB Endowment **15**(11), 2921–2928 (2022)
30. Trummer, I.: Demonstrating gpt-db: Generating query-specific and customizable code for sql processing with gpt-4. Proceedings of the VLDB Endowment **16**(12), 4098–4101 (2023)