Workflow for completing natural-language request with metric-semantic representation of environment

Nguyen Van Hung[1*], Truong Xuan Tung[2], Le Viet Hong[1], Le Khanh Thanh[1]

[1]Control, Automation in Production and Improvement of Technology Institute (CAPITI), Academy of Military Science and Technology, 89 Ly Nam De, Hoan Kiem, Hanoi, Vietnam;
[2]Le Quy Don Technical University, 236 Hoang Quoc Viet, Bac Tu Liem, Hanoi, Vietnam.
[*]Corresponding author: nvhung.v2k@gmail.com
Received 28 Nov. 2024; Revised 17 Jan. 2025; Accepted 04 Apr. 2025; Published 15 Apr. 2025.
DOI: https://doi.org/10.54939/1859-1043.j.mst.102.2025.12-22

ABSTRACT

In mobile robotics and autonomous systems, a natural-language request can be completed by converting it into high-level and low-level tasks. To accomplish such a request, both these types of tasks must be implemented, along with an efficient method to bridge them. However, this problem is still open. This work presents a two-phase workflow (figure 1), including Comprehension and Implementation, based on a metric-semantic map to address this problem. In the Comprehension phase, also known as automated planning, the natural language request is converted into actionable plans using semantic information from the map. These plans are then passed to the Implementation phase, where tasks like navigation or manipulation are executed utilizing geometric information from the map. Moreover, we also conduct an experiment to illustrate how a natural-language request is implemented on a specific metric-semantic presentation of the environment, namely a 3D Scene Graph, with the following complete sequence: from creating the 3D Scene graph until getting the feasible output path. In addition, this work highlights limitations that need to be addressed in the future to enhance the proposed workflow.

Keywords: Natural-language request; Path planning; Task planning; Metric-semantic map; 3D scene graph.

1. INTRODUCTION

The next generation of robots and autonomous systems must be able to understand and serve human (or human-like) instructions (in other words, it is an efficient interaction between human and robot), which is known as *natural-language request* (e.g., “Pick the computer mouse on the ground and place it on the computer desk in bedroom”). So, to support the completion of this request, the environmental representation needs to include not only geometric information but also semantic information (reconsidering the example above, the robot needs to understand what is computer mouse, bedroom,...). Moreover, it also needs to understand the relationship between objects in the environment (e.g., the bedroom contains the computer desk,...).

The traditional geometric map representation (e.g., 2D occupancy grids or 3D voxel-based maps) is each cell as either an “obstacle” or “free-space” (e.g., SLAM [1]). This representation is critical to low-level planning such as safe navigation and to manipulating objects. But for *natural-language requests*, obviously, it could not afford. Because there is not semantic information (a.k.a symbolic knowledge) which is crucial for this. To overcome this problem, the *metric-semantic* map is proposed which integrates the semantic information into tranditional geometric map.

At this point, a key question arises: which semantic information should be included? Typically, the answer depends on the specific objective. Our objective is the metric-semantic map which supports to complete the natural-language request.

The choice of symbols clearly depends on the task the robot has to execute. A mobile robot may implement a path planning query by just using the “free-space” symbol and the corresponding grounding, while a more complex domestic robot may instead need additional symbols (e.g., “cup

of coffee", "table" and "kitchen") and their groundings to execute instructions such as "bring me the cup of coffee on the table in the kitchen".

A potential way to choose the symbols to include in the map therefore consists of inspecting the set of instructions the robot has to execute, and then extracting the list of symbols (e.g., objects and relations of interest) these instructions involve. For instance, in this paper, we mostly focus on solving the natural-language request. Therefore, the symbols we include in our maps include free-space, objects, rooms, and buildings. After establishing the list of relevant symbols, the goal is for the robot to build a compact sub-symbolic representation (i.e., a geometric map), and then integrate these symbols into the sub-symbolic representation (figure 2).

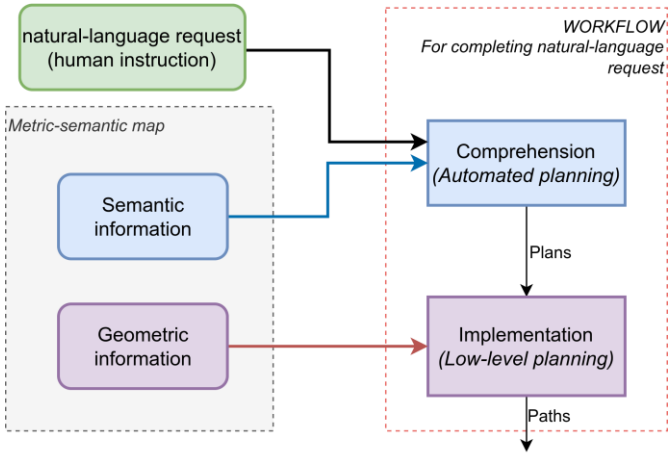


Figure 1. *Workflow and relevant information of metric-semantic map to serving natural-language request.*

The Comprehension and Implementation in figure 1 is detailed in figure 3 and figure 4 respectively.

Given a formal metric-semantic representation of the world, to complete the natural-language request, the autonomous robot needs to do the workflow (depicted in figure 1) of two steps: *Comprehension* and *Implementation*. *Comprehension* (depicted in figure 3 in detail), also known as automated planning, is a field in artificial intelligence (AI) that focuses on generating a sequence of actions or decisions to achieve a specific goal or solve a problem. In other words, this is the process of translating the natural-language request into actionable plans based on the combination of techniques of Natural language processing (NLP) [2, 3], the planning problem [4] is defined by planning domain description language (PDDL) [5], AI planner [6] and metric-semantic map.

Implementation (illustrated in Figure 4 in detail) is doing low-level planning such as motion planning (for mobile robots) and manipulation planning (for manipulators). This is required for computing the actual movements that the robot should carry out. This process receives the plans from *Comprehension* step and then does low-level planning based on the geometric information from the representation of environment.

The contributions of this work are as follows:

1. Proposing the workflow for completing natural-language requests using the information which is contained in metric-semantic representation of environment.
2. Conducting the experiment to demonstrate the practicality and completeness of the proposal.

2. PRELIMINARIES AND RELATED WORK

Natural language processing (NLP) [2, 3], based on transformer-based (neural) architecture, is

a branch of artificial intelligence (AI) that enables computers to comprehend, generate, and manipulate human language.

A planning problem [4] is composed of a planning domain $\mathcal{D}$ and an instance p, defined by PDDL language.

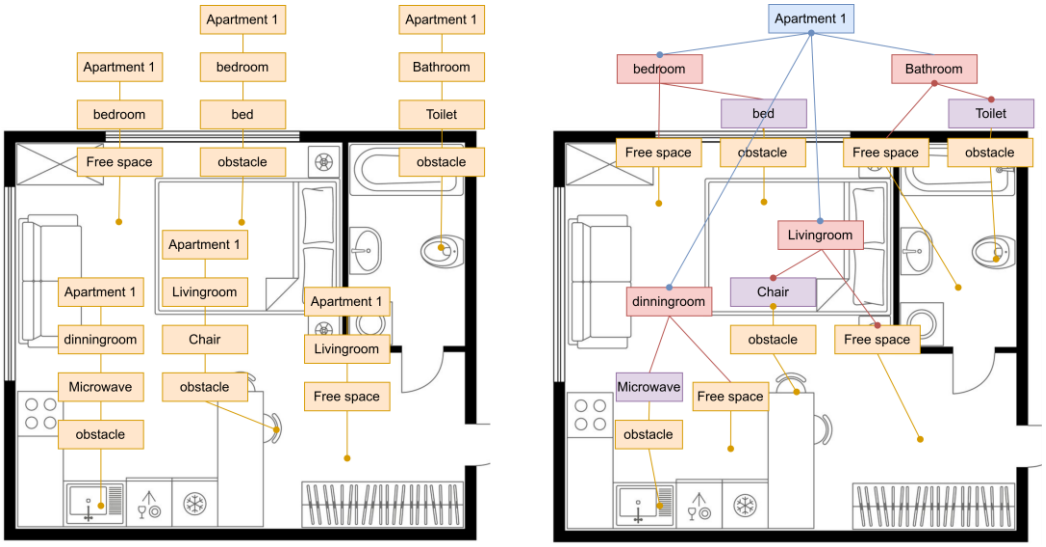


Figure 2. *Flatted and Heirarchical metric-semantic map.*

The planning domain $\mathcal{D}$ is made up of several action models. An action model is defined by a tuple of $\mathcal{A} = \big(a, pre(a), eff(a)\big)$, where a is an action name, pre(a) is a set of preconditions requiring to be satisfied when executing a, eff(a) is the result of a. In PDDL, a planning environment is described in terms of objects in the world, predicates that describe relations between these objects, and actions a that modify the world by manipulating these relations.

AI planner [6, 7] is a system or algorithm designed to generate sequences of actions, or plans, to achieve specific goals based on a given set of inputs, constraints, and objectives. The output plan consists of a series of time steps, each of which can have one or more instantiated actions with concurrency semantics. A planner devises plans by searching in the space of states, where a state is a configuration of physical objects or partial plans.

A semantic map is a representation of an environment enriched with meaningful annotations or labels that describe the nature and function of its regions, objects, or elements. In the literature on metric-semantic map, they are categorized into two main types of structure: flat and hierarchy. A flat metric-semantic map [8, 9], is that the semantic information added to the geometric object and do not separate or sort them into the layers (as depicted in left pane of figure 2). It is called *Flat* because each cell is assigned a list of symbols of interest.

The second type should be organized as a hierarchy. Davision et al. [10] argued that such representations must be hierarchical in nature, since a suitable hierarchical organization reduces storage during long-term operation and leads to provably efficient inference. The specific line of the hierarchical metric-semantic map is called as “3D Scene Graph” (shortly 3DSG, depicted in right pane of figure 2). It is a layered undirected graph where nodes represent spatial concepts (e.g., objects, rooms, agents) and edges represent spatio-temporal relations. The graph is layered, in that nodes are grouped into layers that correspond to different levels of abstraction of the scene [8, 11–13].

Probably Armeni et al. [11] suggested firstly the 4-layers structure of 3DSG including metric-semantic mesh (i.e., low-level geometry), Rooms, Object and Camera. But its drawback is only representing the static 3D space as well as only use offline. Rosinol et al. [12, 13] suggested the spacial perception which creates the 5-layers 3DSG structure including (i) Metric-semantic mesh, (ii) Objects and agents, (iii) Places and structures, (iv) Rooms and (v) Buildings from visual-odometric data. It represents for not only static space but also dynamic agent (in objects and agents layer). So, it is suitable to the dynamic environment.

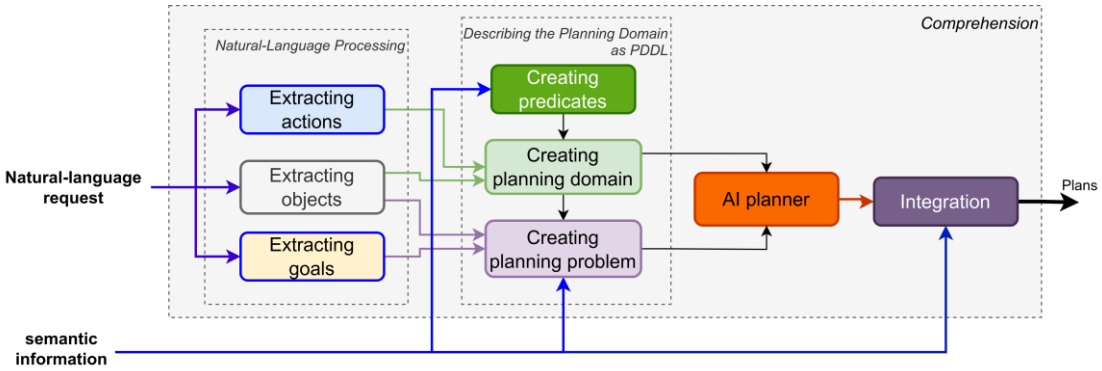


Figure 3. *The diagram of comprehension steps.*

Hughes et al. [14] suggested the system called *Hydra* which as well creates the same 5-layers 3DSG structure as Rosinol's work. Especially, Hydra is highly parallel and it's most of the parts can run on CPU. So, it highly fits with the systems which have the high mobility and the restricted resources.

Motion planning [15] is the process of determining a feasible path or trajectory $\tau(t)$ for a robot to move from a starting configuration $q_s = \tau(0)$ to a goal configuration $q_g = \tau(T)$ while avoiding obstacles $\tau(t) \in C_{free}$ and satisfying constraints. Manipulation planning involves planning for a robot to interact with objects in its environment, often requiring both motion planning and reasoning about the object's properties and constraints. This includes tasks like grasping, pushing, or placing objects. Its output is robot configurations $\{q_1, q_2, \dots, q_k\} \in C$ (C is robot's configuration space) and object states $\{o_1, o_2, \dots, o_k\} \in O$ (O is object's state space).

3. COMPLETING NATURAL-LANGUAGE REQUEST WITH METRIC-SEMANTIC MAP

3.1. Comprehension

This procedure is illustrated in figure 3 and implemented in figure 4 in detail.

Algorithm 1	Doing the comprehension
Input:	$\mathcal{M}$ (Metric-semantic map), $\mathcal{R}$ (natural-language request)
Output:	$\boldsymbol{\pi}$ (plans)

1 $(\mathcal{A}, \mathcal{O}, \mathcal{G}, s_0) \leftarrow$ processNaturalLanguage($\mathcal{R}$)
2 $\mathcal{D} \leftarrow$ createPlanningdomain($\mathcal{A}, \mathcal{O}, \mathcal{M}$)
3 $\mathcal{P} \leftarrow$ createPlanningproblem($\mathcal{G}, \mathcal{O}, s_0, \mathcal{M}$)
4 $\mathcal{P}_\ell \leftarrow$ AIPlanning($\mathcal{D}, \mathcal{P}$)
5 $\boldsymbol{\pi} \leftarrow$ integrate($\mathcal{P}_\ell, \mathcal{M}$)

Because of the request $\mathcal{R}$ is natural-language, the first and foremost task to be carried out is to

process the natural language. A general way is to extract objects $\mathcal{O}$, initial conditions s_0 and goals $\mathcal{G}$ by parsing sentences for annotations (e.g., OpenNLP[1] and Stanford CoreNLP [16]), and then learn causal relationships between them and formalize the relations by actions $\mathcal{A}$ (line 1 of Algorithm 1).

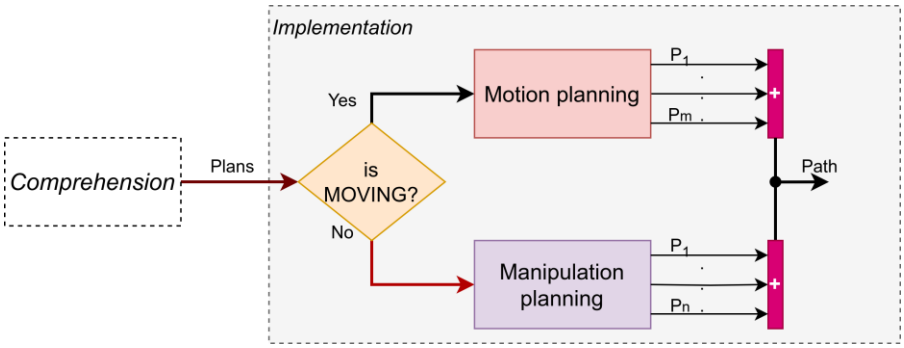


Figure 4. *The diagram of implementation steps.*

Next step, the planning domain $\mathcal{D}$ and instance of problem p are created and described as PDDL language [5] from the result of step 1 and semantic information $\mathcal{M}$ (line 2 and 3 of Algorithm 1 respectively).

Classic AI planners like FF [17], Downward [7], etc are used as task planner with inputs of domain and problem (line 4 of Algorithm 1). The output from the AI planner is then integrated with the necessary information derived from the semantic data provided by the semantic map $\mathcal{M}$ (line 5 of Algorithm 1). The final result is plans $\boldsymbol{\pi}$ which is ready for implementation.

For example, the robot receives the request "***Bring the book to and place it on the desk***". Then it will do comprehension as depicted in detail in figure 3 as following: The actions are *Pick-up* (grasping the book), *move* (approaching the desk) and *Place* (place the book on the desk, the objects are *book* and *desk*, the goal is *book on the desk* are inferred.

Algorithm 2	Doing the implementation
Input:	$\boldsymbol{\pi}$ (plans), $\mathcal{M}$ (Metric-semantic map)
Output:	$\boldsymbol{\mathcal{P}}$ (resulting path)
1 **for**	plan $\mathcal{T} \in \boldsymbol{\pi}$ **do**
2 **if**	$\mathcal{T}$ is moving **then**
3	$\mathcal{P}_{mo} \leftarrow$ doMotionPlanning$(\mathcal{T}, \mathcal{M})$
4 **else**	
5	$\mathcal{P}_{ma} \leftarrow$ doManipulatePlanning$(\mathcal{T}, \mathcal{M})$
6 $\boldsymbol{\mathcal{P}} \leftarrow \mathcal{P}_{mo} \oplus \mathcal{P}_{ma}$	

The planning domain includes some definition of actions, such as *Pick-up an object*, *Move an object* and *Place an object on a location*, and some predicates like *object is at a location*, *Location is clear*, *Agent's hand is empty*, etc. The planning problem consists of INIT is *book at table*, *desk is clear* and *agent's hand is empty* and GOAL is *book must be on the desk* and *agent's hand is empty*, etc. The output of semantic plans coming from AI planners is *pick-up book from table*, *move agent from living room to bedroom* and *place book on desk*. They are arranged in the order of execution.

[1] https://opennlp.apache.org/

The information of objects needed for plans are extracted and integrated into plans to output the final plans as following: *pick-up bbox(0.2, 0.15, 0.03) from (1.0, 1.0, 1.0)*, *move agent from (0.0, 0.0, 0.0) to (5.0, 5.0, 0.0)* and *place bbox(0.2, 0.15, 0.03) on (7.0, 6.0, 1.0)*. And then they will be passed to implementation to carry out.

3.2. Implementation

Based on the plans **π** from comprehension, the robot do motion and manipulation planning step by step in sequense. This progress is detailed in figure 4 and implemented in algorithm 2.

With every plan $\mathcal{T}$ in sequence of plans $\boldsymbol{\pi}$ is checked either motion or manipulation (line 2 of Algorithm 2). After that, the planning algorithm [15] such as Graph-Search-based, heuristics-based, sampling-based, optimization-based algorithms, is chosen according to the data and representation of map and plan $\mathcal{T}$.

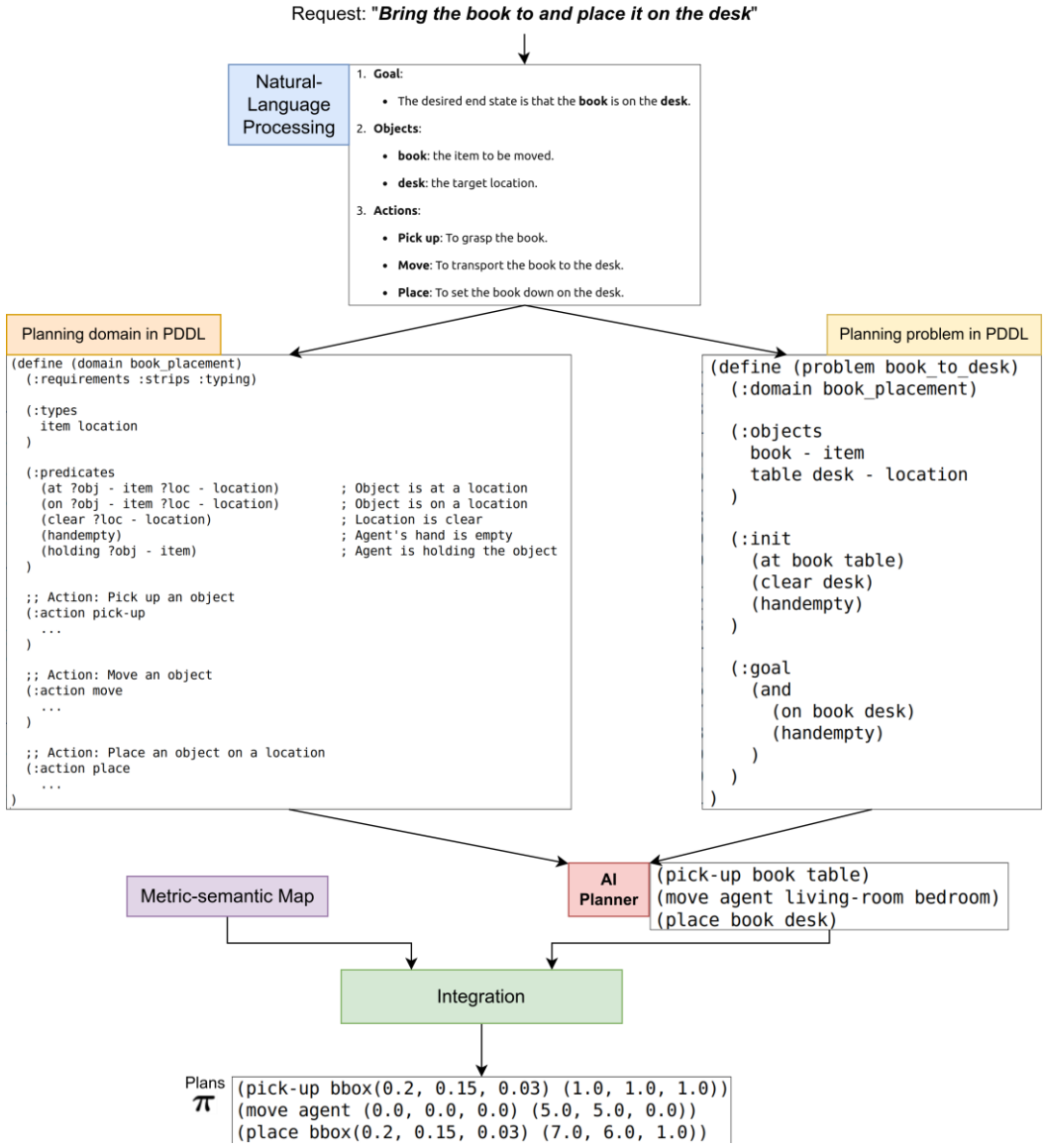


Figure 5. *The detailed comprehension about specific example "Bring the book to and place it on the desk".*

Every step will output the resulted path $\mathcal{P}_{mo}$ or $\mathcal{P}_{ma}$ respectively. Finally, all of these resulted paths are concentrated together into completed path $\boldsymbol{\mathcal{P}}$ (line 6 of Algorithm 2). A specific example is shown in figure 6, the final path includes three bolder and thicker segments: The first one is the path for pick the book up, the second is the path for approaching the desk, the last one is the path for put off the book on the desk.

4. EXPERIMENTS

This section shows the steps practically how to finish a natural-language request from creating metric-semantic map until the resulting path. As mentioned in section 2, the metric-semantic map has two types of structure: flatted and hierarchical. Without loss of generality, we are illustrating the workflow for completing natural-language request here. This workflow is independent of the map's structure, so the experiment only is demonstrated on a hierarchical metric-semantic map. It is similar for the remaining types of map structures. A typical type of natural hierarchical metric-semantic map is the 3D Scene Graph [12, 13]. This section conducts experiments on it. Additionally, the places layer of it enables motion planning directly without requiring any additional processing.

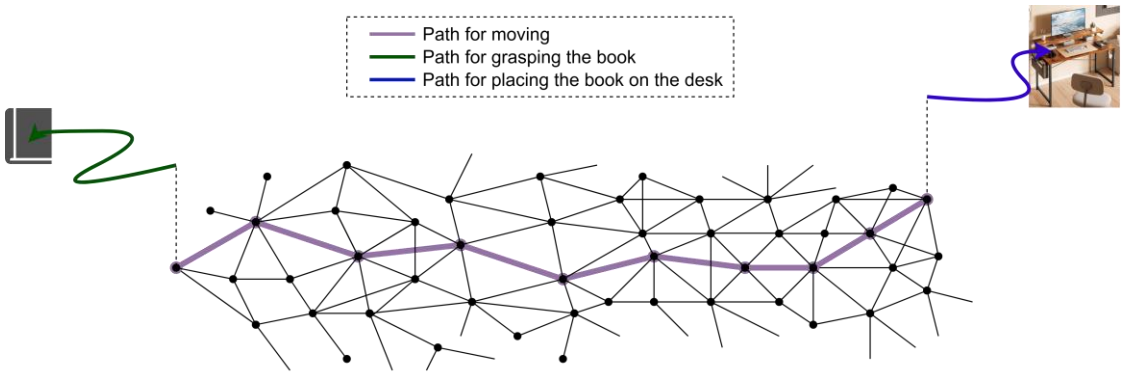


Figure 6. *The resulted path encompassing three segments.*

In the experiments, we have created the our own PARSER based on CoreNLP [16] to do NLP (line 1 of Algorithm 1) and the planning domain and the problem instance will be handcrafted (line 2-3 of Algorithm 1). Finally, Downward [7] is used as AI planner (line 4 of Algorithm 1).

The hardware is used to be: a ThinkPad P1 Gen 3 with an Intel Core i7-10875H @ 2.30 GHz with 16 cores and one Nvidia TU117GLM [Quadro T1000 Mobile].

4.1. Creating the metric-semantic map (3D Scene Graph)

Datasets: We created 3D Scene Graph based on a standard dataset, namely uHumans2 (uH2)[2]. This dataset [13] are generated using a photorealistic Unity-based simulator provided by MIT Lincoln Laboratory. It includes three scenes: a small apartment, an office, and a subway station. This section uses the scene of office for illustration.

Hydra [14] (Depicted in figure 7) is used to create the 3DSG. Each block in the figure denotes an algorithmic module.

Hydra starts with fast early perception processes (i.e., low-level perception tasks) such as 2D semantic segmentation, stereo-depth reconstruction and visual-inertial odometry. The result of early perception processes are passed to mid-level perception processes. These include algorithms that incrementally construct the agent layer, the mesh and places layers, and the object layer. Mid-

[2] https://web.mit.edu/sparklab/datasets/uHumans2/

level perception also includes the scene graph frontend, which is a module that collects the result of the other modules into an "unoptimized" scene graph.

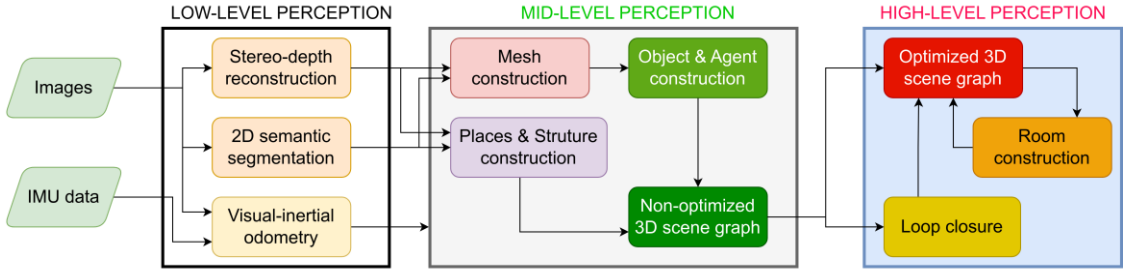


Figure 7. *Hydra's functional blocks [14].*

In figure 7, a tool is for creating 3D scene graphs online with minimal GPU requirements.

Finally, the high-level perception processes perform loop closure detection, execute scene graph backend which outputs optimized 3D scene graph, and finally extracting rooms.

Hydra runs in real-time on a multi-core CPU. The only module that relies on GPU computing is the 2D semantic segmentation, which uses a standard off-the-shelf deep network. The fact that most modules run on CPU has the advantage of (i) leaving the GPU to learning-oriented components, and (ii) being compatible with the power limitations imposed by current mobile robots. The result from Hydra is a globally consistent, persistent 3D scene graph in figure 8. The 3DSG data is saved as JSON file for using in latter step of processing.

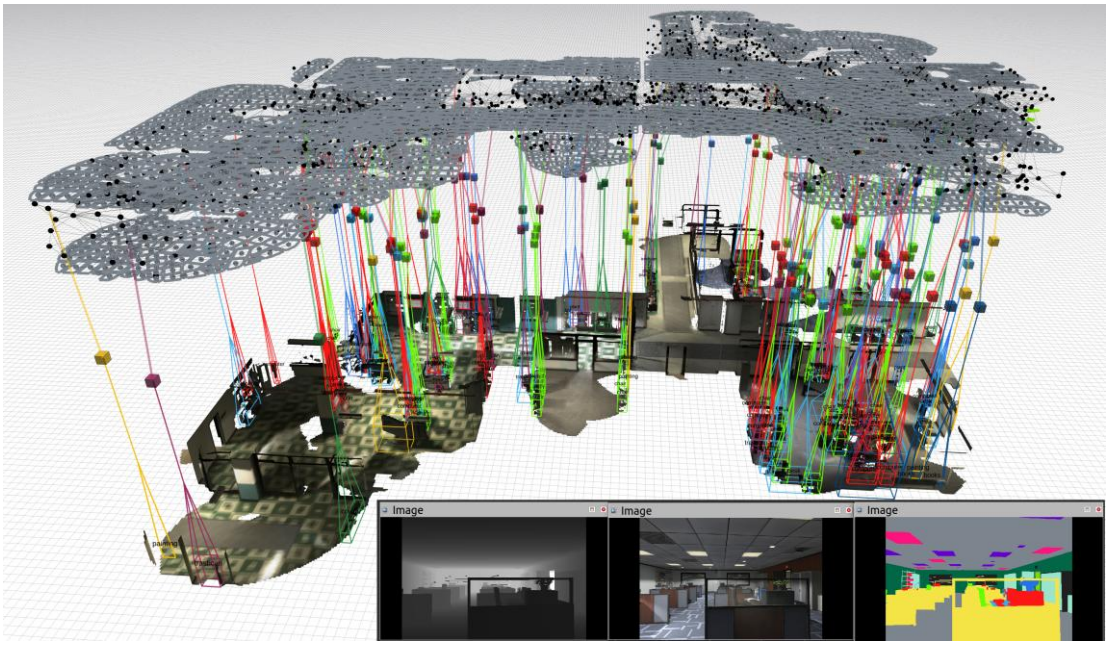


Figure 8. *3D scene graph created by Hydra in the uH2 Office dataset.*

4.2. Serving the natural-language request

We present the steps to completing any natural-language request. For example, it is "Pick-up the Book at table in Room1 and then bring it to desk in Room9". First and foremost, after we had the DSG from sub-section 4.1 which was saved as JSON file. We must get it by loading this JSON file (line 1 of Algorithm 3).

Algorithm 3	Completing the specific request

Input: $\mathcal{F}_{3dsg}$ (Saved file contains 3D scene graph), $\mathcal{R}$ (natural-language request)

Output: $\boldsymbol{\mathcal{P}}$ (resulting path)

1 $\mathcal{M} \leftarrow$ readFromFile($\mathcal{F}_{3dsg}$)
2 $(\mathcal{A}, \mathcal{O}, \mathcal{G}, s_0) \leftarrow$ myOwnParsing($\mathcal{R}$)
3 $\mathcal{D} \leftarrow$ create Planning domain from $(\mathcal{A}, \mathcal{O}, \mathcal{M})$
4 $\mathcal{P} \leftarrow$ create Planning problem from $(\mathcal{G}, \mathcal{O}, s_0, \mathcal{M})$
5 $\pi \leftarrow$ planningByDownward($\mathcal{D}, \mathcal{P}$)
6 **for** plan $\mathcal{T} \in \boldsymbol{\pi}$ **do**
7 **if** $\mathcal{T}$ is moving **then**
8 $\mathcal{T}' \leftarrow$ integrate($\mathcal{T}, \mathcal{M}$)
9 $\boldsymbol{\mathcal{P}} \leftarrow$ doA*($\mathcal{T}', getPlacesLayer(\mathcal{M})$)
10 visualizePath($\boldsymbol{\mathcal{P}}$)

With our own coreNLP-based parser, we have been extracted: The actions are *Pick-up* (grasping the book), *move* (approaching the desk) and *Place* (place the book on the desk), the objects are *book* and *desk*, the goal is *book on the desk*, relationships are *Book at table*, *table in Room1* and *desk in Room9* (line 2 of Algorithm 3).

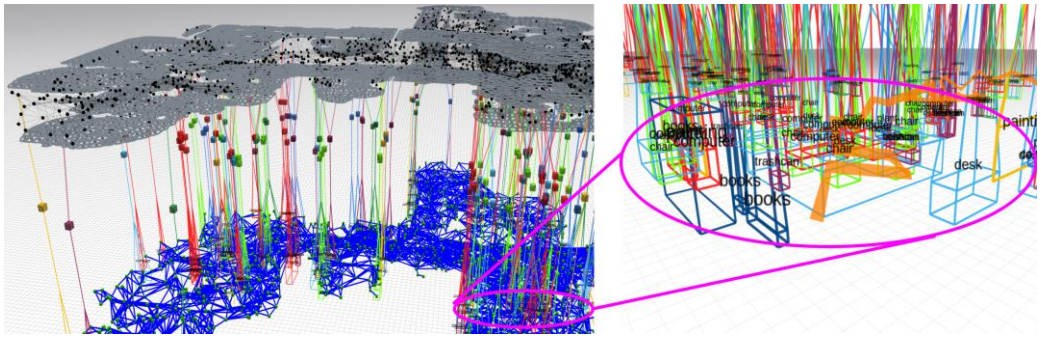


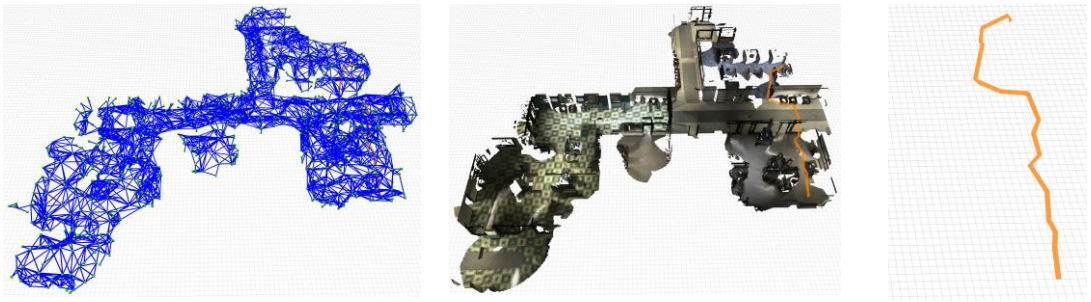
Figure 9. *Extracted 3DSG's place layer and attribute of name.*

In figure 9, right pane is the place layer (Black is origin, Blue is extracted one); Left pane is a part of name of all objects.

From these actions, we formalize them into the planning domain (line 3 of Algorithm 3) as *Pick-up an object*, *move an object* and *Place an object on a location*. From relationships, we create some predicates like *object is at a location*, *object is in room*, *location is clear*, *robot's hand is empty*. The other predicates come from the map like *Room is connected to the corridor*, *RoomX and RoomY is at same floor*, etc.

The planning problem (line 4 of Algorithm 3) consists of INIT is *book at table*, *table in Room1*, *desk in Room9*, *desk is clear* and *robot's hand is empty* and GOAL is *book must be on the desk* and *agent's hand is empty*.

The output of semantic sequential plans coming from AI planners is *(pick-up book table)*, *(move agent room1 room9)* and *(place book desk)*. Based on the semantic information of objects, the data is necessary for plans are extracted and integrated into to create the final plans as following: *(pick-up bbox(0.2, 0.15, 0.03) (-12.81, 17.63, 1.97))*, *(move robot (-11.412, 17.4467, 2.35333) (23.2246, 43.8797, 2.55507))* and *(place bbox(0.2, 0.15, 0.03) (18.90, 42.85, 1.42))*.



a) Extracted places and Path (orange)

b) Path vs Mesh

c) Only Path

Figure 10. *(a) The resulting path (orange) coming from A* on extracted places layer; (b) The resulting path along with mesh to enable more intuitively; (c) The resulting path alone for clarity.*

Without loss of generality, because of this experiment is for demonstrating the workflow, so we do not need to implement all these low-level plans. For simplicity and to leverage the places layer of the 3DSG, we choose to perform only motion planning. So, we ignored to do manipulation planning. Based on the semantic symbols derived from their names (e.g., book, desk, Room1, Room9) as specified in the request, we explored the 3DSG to extract their attributes, such as location, bounding box, and others. These attributes were then integrated to formulate the motion task (line 8 of Algorithm 3). Next, we do A* search [18] on places player (line 9 of Algorithm 3) to be extracted from 3DSG. The extraced places layer and some semantic information of 3DSG are depicted in figure 9. Finally, the resulting path is visualized (line 10 of Algorithm 3) in the figure 10.

5. CONCLUSIONS

This work introduced a workflow for completing natural-language request based on the metric-semantic representation of the environment. This capability is vital for the next generation of robotics, enabling them to reason and interact with humans in a more intelligent manner. Through experiments in section 4 on a particular type of metric-semantic map, the 3D scene graph, it is shown that the proposed workflow is practical and completed.

However, there are still some issues that need to be addressed to improve the quality and efficiency of this workflow: (i) Some steps are not yet fully automated, such as creating the planning domain and the instance of the planning problem; (ii) The environment description using predicates is not yet detailed or fully automated; (iii) The combination between comprehension and implementation remains basic. Specifically, comprehension generates a sequence of plan, and implementation executes it without considering whether the plan is feasible, thus lacking the feedback for comprehension to perform re-planning if necessary.

REFERENCES

[1]. Cadena C., Carlone L., Carrillo H., et al. "*Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age*," IEEE Transactions on Robotics, **32**(**6**), 1309–1332, (2016).

[2]. Khurana D., Koli A., Khatter K., et al. "*Natural language processing: state of the art, current trends and challenges*," Multimedia Tools and Applications, **82**(**3**), 3713–3744, (2022).

[3]. Torfi A., Shirvani R.A., Keneshloo Y., et al. "*Natural Language Processing Advancements By Deep Learning: A Survey*", (2021). https://arxiv.org/abs/2003.01200

[4]. Jin K. and Zhuo H.H. "*Integrating AI Planning with Natural Language Processing: A Combination of Explicit and Tacit Knowledge*", (2023). https://arxiv.org/abs/2202.07138

[5]. Aeronautiques C., Howe A., Knoblock C., et al. "*Pddl\textbar the planning domain definition language,*" Technical Report, Tech Rep, (1998).
[6]. Ghallab M., Nau D., and Traverso P., "*Automated planning and acting,*" Cambridge University Press, (2016).
[7]. Helmert M. "*The Fast Downward Planning System,*" Journal of Artificial Intelligence Research, **26**, 191–246, (2006).
[8]. Rosinol A., Abate M., Chang Y., et al. "*Kimera: an Open-Source Library for Real-Time Metric-Semantic Localization and Mapping*", (2020).
[9]. Grinvald M., Furrer F., Novkovic T., et al. "*Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery,*" IEEE Robotics and Automation Letters, **4**(**3**), 3037–3044, (2019).
[10]. Davison A.J. "*FutureMapping: The Computational Structure of Spatial AI Systems*", (2018).
[11]. Armeni I., He Z.-Y., Gwak J., et al. "*3D Scene Graph: A Structure for Unified Semantics, 3D Space, and Camera*", (2019). https://github.com/StanfordVL/3DSceneGraph
[12]. Rosinol A., Gupta A., Abate M., et al. "*3D Dynamic Scene Graphs: Actionable Spatial Perception with Places, Objects, and Humans*", (2020). https://arxiv.org/abs/2002.06289
[13]. Rosinol A., Violette A., Abate M., et al. "*Kimera: from SLAM to Spatial Perception with 3D Dynamic Scene Graphs*", (2021). https://arxiv.org/abs/2101.06894
[14]. Hughes N., Chang Y., and Carlone L. "*Hydra: A Real-time Spatial Perception System for 3D Scene Graph Construction and Optimization,*" Robotics: Science and Systems (RSS), (2022).
[15]. Steven M. LaValle, "*Planning algorithms,*" Cambridge University Press, (2006).
[16]. Manning C., Surdeanu M., Bauer J., et al. "*The Stanford CoreNLP Natural Language Processing Toolkit,*" Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, Baltimore, Maryland, Association for Computational Linguistics, 55–60, 55–60, (2014).
[17]. Hoffmann J. "*FF The Fast-Forward Planning System,*" AI Mag, **22**(**3**), 57–62, (2001).
[18]. Hart P.E., Nilsson N.J., and Raphael B. "*A Formal Basis for the Heuristic Determination of Minimum Cost Paths,*" IEEE Transactions on Systems Science and Cybernetics, **4**(**2**), 100–107, (1968).

TÓM TẮT

Quy trình hoàn thành yêu cầu ngôn ngữ tự nhiên dựa trên metric-semantic map

Trong lĩnh vực robotic và các hệ thống tự hành, một yêu cầu ngôn ngữ tự nhiên có thể được hoàn thành bằng cách chuyển đổi nó thành các nhiệm vụ bậc cao và bậc thấp. Vậy để hoàn thành yêu cầu này, cả hai loại nhiệm vụ này đều phải được thực hiện, và làm sao để kết nối chúng hiệu quả. Tuy nhiên, vấn đề này vẫn còn đang được mở. Công trình này giới thiệu một quy trình hai giai đoạn (hình 1) bao gồm Comprehension và Implementation, dựa trên metric-semantic map để giải quyết vấn đề này. Trong giai đoạn Comprehension, còn gọi là lập kế hoạch tự động, yêu cầu ngôn ngữ tự nhiên được chuyển đổi thành một chuỗi kế hoạch bằng cách sử dụng thông tin ngữ nghĩa (semantic) từ bản đồ. Các kế hoạch này sau đó được chuyển sang giai đoạn Implementation, nơi các nhiệm vụ như điều hướng (navigation) hoặc thao tác (manipulation) được thực hiện bằng cách tận dụng thông tin hình học (metric) từ bản đồ. Hơn nữa, chúng tôi cũng đã tiến hành thử nghiệm để minh họa cách một yêu cầu ngôn ngữ tự nhiên được thực hiện trên một metric-semantic map cụ thể, đó là 3D Scene Graph, với trình tự hoàn chỉnh sau: từ việc tạo 3D Scene Graph đến khi nhận được kết quả là đường đi khả thi. Ngoài ra, công trình này cũng chỉ ra các hạn chế cần được khắc phục trong tương lai để cải thiện quy trình đã đề xuất.

Từ khoá: Yêu cầu bằng ngôn ngữ tự nhiên; Lập kế hoạch đường đi; Lập kế hoạch nhiệm vụ; Bản đồ hình học-ngữ nghĩa; Đồ thị cảnh 3D.