Object-Level Planning and Abstraction

David Paulius
Department of Computer Science
Brown University, Providence, RI, United States
{dpaulius}@cs.brown.edu

Abstract: Task and motion planning (TAMP) aims to integrate higher-level symbolic task planning with lower-level motion planning. However, task-level representations must include detailed information expressing the robot's own constraints, mixing logical object-level requirements (e.g., a bottle must be open to be poured) with robot constraints (e.g., a robot's gripper must be empty before it can pick up an object). We propose an additional level of planning that will rest above the current TAMP pipeline known as *object-level planning* (OLP). OLP is concerned with objects, but not with the robot and its constraints. It allows the robot to generate plan sketches that describe how objects in the environment must be changed to realize a plan, leaving the details of how these are achieved to task-level planning. Object-level plans have the benefit of being interpretable and portable while supporting generalization across objects and the details of any specific task. Using *functional object-oriented networks* (FOONs), a representation we introduced in prior work, we will briefly outline how object-level plans can be used to model object-level manipulation planning, resulting in plan sketches that can be extracted from natural language, inform a task-level PDDL planner, and generalize across object types.

Keywords: Object-level planning, Task and motion planning, Abstraction

1 Introduction

The objective of *task and motion planning* (TAMP) is to interweave the processes of task planning (which pertains to finding a sequence of actions for a robot that realizes higher-level state transitions to advance to a goal state) and motion planning (which pertains to finding collision-free motions that realize the effects of its task plan) [1, 2, 3]. This has the advantage of reducing the complexity of mobile manipulation, as robots can take advantage of structure of its actions and world to achieve its intended goals. This complexity becomes most apparent in unstructured and dynamic environments, where motion planning alone is unlikely to succeed. For this reason, combining task-level knowledge with low-level spatial reasoning allows the robot to reason about its actions and objects in its world and identify key sub-goals required to achieve its target goal.

However, although TAMP simplifies motion planning and execution through task-level planning, the domain knowledge used in TAMP is specific to both to a single robot and its environment, as a task planning domain definition (conventionally using the *Planning Domain Definition Language* [4] (PDDL)) must be written to completely specify the robot's plan, including logical constraints related to the robot's own state, to provide a plan that can be realized by motion planning. As a result, permissible actions and features of the robot do not allow plans to be *portable*. Furthermore, domain specifications written in languages such as PDDL are not easily *interpretable*, as their domain and problem definitions are not naturally written or communicated in human language because they include too much detail. Moreover, objects described in PDDL are not truly object-centric. Actions are parameterized using arguments without semantic meaning. Similarly, task planning does not take full advantage of abstraction and generalization, as task plans are designed for specific settings and do not generalize to new situations or environments.

We therefore propose an additional level of planning and domain knowledge that rests above the current TAMP pipeline, which we call *object-level planning* (OLP). The objective of OLP is to generate plan sketches that are portable across robots and environments, at the level at which language is normally expressed, using domain knowledge for robotic manipulation and to bootstrap task and motion planning processes. An *object-level representation* is one which is *agnostic* to the robot or agent, meaning that constraints related to the robot do not appear in object-level plans, which describe only the objects relative to a plan – using semantically meaningful labels – and the changes they must undergo. In other words, an object-level plan should serve as a schema that describes the objectives of a given task in a way that is closer to human language. Recipes or other instructional materials present information in an object-level manner, where nouns refer to object types rather than instances and actions describe high-level skills or policies, whose implementations may vary across robots (and indeed across kitchens, homes or other human-centered environments).

2 Motivation for Object-Level Planning

We propose several key motivations for the use of OLP for robotic planning:

OLP promotes portability across robots and domains: Recipes in cookbooks or other forms of instructional manuals convey domain knowledge and plans in an abstract yet useful way, and through them, a robot is presented with an object-level representation of knowledge that is agnostic to the robot's environment or functionality. In other words, any robot can pick up an object-level plan, understand and reason with it, and carry out its steps as long as the robot is equipped to ground words and phrases to its surroundings. These plans are designed to work across environments, as they provide the bare minimum details needed to work in new settings or environments. Robots using object-level plans do not have to commit to a specific way of solving a problem, as task-level solutions can vary depending on the robots' capabilities, thus requiring task-level planning.

OLP supports plan generalization: Object-level plans present information with language using verbs and nouns to describe actions and objects respectively. As a result, adapting these plans is made possible through natural language processing techniques, which is markedly easier than modifying plans at a lower level of TAMP hierarchy. For instance, in previous work, I showed that semantic similarity can be used as a method to infer relationships or commonality between objects to extend domain knowledge to new object labels in a FOON knowledge graph [5, 6]. As a follow-up to this work, Sakib et al. [7] showed how semantic similarity can be used to generate object-level plans for novel recipes in the Recipe1M+ dataset, where, given a set of ingredients and a dish type, a reference task tree can be found that closely resembles the requested dish. This reference task tree can then be modified to match the ingredient set by adding or removing references to these ingredients. Such object-level plans can then be used to formulate task-level planning definitions.

OLP promotes interpretability and explainability: A significant limitation to task specifications written using representations like PDDL [4] is that they are not easily interpretable. PDDL notation conveys knowledge using logical predicates, which can be true or false depending on the current state of the world. This type of notation is simple yet effective, but it does not provide a truly object-centric description, as several predicates may be written to represent a single object in the world. When dealing with significantly complex and realistic worlds such as human-centered environments, there may be many object types and instances, which would make a domain description overly complicated and difficult to verify. Using an object-level representation would convey details about the task at a level that is closer to human language and understanding, through which a robot can communicate its intentions or behaviours to a human.

OLP can bootstrap planning and execution: One of the greatest strengths of object-level plans is their use for bootstrapping TAMP processes, which are conventionally performed by task planning, though a combination of PDDL and off-the-shelf planners (e.g., Fast-Downward [8]), and motion planning algorithms. Formulating these problems in PDDL can be an arduous task, especially when

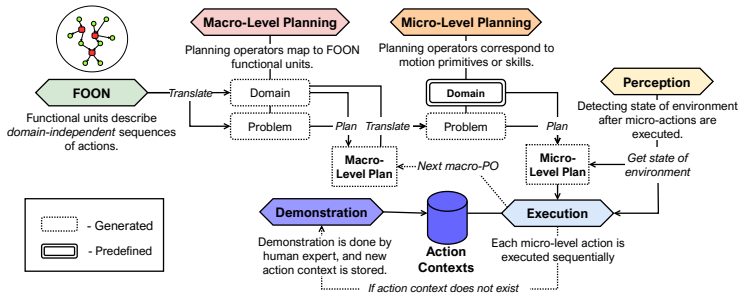

Figure 1: Overview of approach [9] for: 1) translating a FOON into a macro-level domain and problem representation in PDDL, 2) decomposing each macro-level action (i.e., planning operator) into a micro-level PDDL problem, 3) planning at the micro-level using pre-defined motor skills (written with object-centered predicates [10]), and 4) executing the acquired micro-level manipulation plan using action contexts (learned from demonstration) and their associated movement primitives [11].

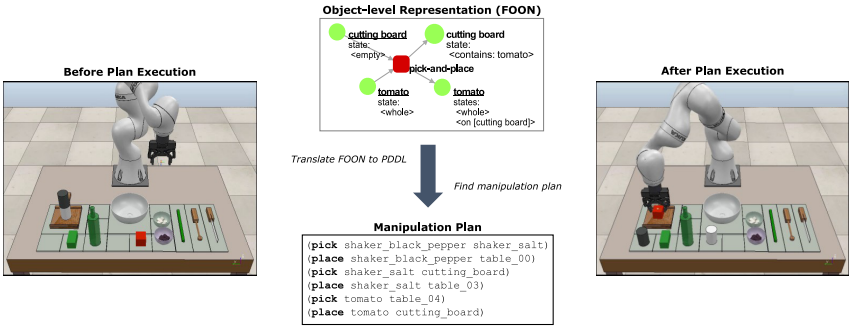

Figure 2: Illustration showing how a FOON functional unit can be used to define a PDDL problem instantiation for finding a manipulation plan executed in simulation.

dealing with domain knowledge of a large scale. Therefore, we can leverage object-level representations for seamless translation to PDDL domain and problem definitions, which can then be used for domain- and robot-specific manipulation or motion planning. In our recent work [9], we have created a pipeline around FOON for downstream PDDL translation, planning, and execution of manipulation plans. Although this approach is not fully TAMP (due to a lack of motion planning), it takes advantage of OLP to formulate task-level domain and problem definitions that can handle novel configurations of the world state by flexibly relying upon action contexts learned from demonstration and object-centered predicates [10]. Figure 1 gives an overview of how we decompose FOONs (either a universal FOON or a task tree derived from FOON using task tree retrieval [5]) into two planning levels: *macro-level planning* – which is concerned with higher level objectives closer to human understanding of tasks – and *micro-level planning* – which is concerned with task-specific actions and motion primitives that would reflect the intended effects of higher-level macro-actions. An example of a manipulation plan, given an object-level description (i.e., FOON functional unit) for picking-and-placing a tomato on a cutting board, is shown in Figure 2.

3 Conclusion

To conclude, in this paper, an additional layer of planning and representation on top of the conventional task and motion planning (TAMP) pipeline, known as object-level planning (OLP), promises benefits to OLP and object-level representations, some of which have been validated through previous work [5, 6] and ongoing investigations [9] with functional object-oriented networks (FOON). As future work, object-level representations will become formalized, where key properties of such representations will be identified and grounded more formally in a TAMP framework.

Acknowledgments

The author would like to thank George Konidaris for discussions that manifested the ideas presented in this work and for guidance in writing this paper. Additionally, the author would like to acknowledge the support of the Office of Naval Research (ONR) through grant number N00014-21-1-2584.

References

[1] L. P. Kaelbling and T. Lozano-Pérez. Hierarchical Planning in the Now. In *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.

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

[3] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez. Integrated Task and Motion Planning. *Annual Review of Control, Robotics, and Autonomous Systems*, 4:265–293, 2021.

[4] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL – The Planning Domain Definition Language. Technical report, CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, 1998.

[5] D. Paulius, Y. Huang, R. Milton, W. D. Buchanan, J. Sam, and Y. Sun. Functional Object-Oriented Network for Manipulation Learning. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2655–2662. IEEE, 2016.

[6] D. Paulius, A. B. Jelodar, and Y. Sun. Functional Object-Oriented Network: Construction and Expansion. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5935–5941. IEEE, 2018. doi:10.1109/ICRA.2018.8460200.

[7] M. S. Sakib, D. Paulius, and Y. Sun. Approximate Task Tree Retrieval in a Knowledge Network for Robotic Cooking. *IEEE Robotics and Automation Letters*, 7(4):11492–11499, 2022. doi:10.1109/LRA.2022.3191068.

[8] M. Helmert. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

[9] D. Paulius*, A. Agostini*, and D. Lee. Long-Horizon Manipulation Planning with Functional Object-Oriented Networks. *arXiv preprint arXiv:2207.05800*, 2022.

[10] A. Agostini, M. Saveriano, D. Lee, and J. Piater. Manipulation Planning using Object-centered Predicates and Hierarchical Decomposition of Contextual Actions. *IEEE Robotics and Automation Letters*, 5(4):5629–5636, 2020.

[11] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal. Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviors. *Neural Computation*, 25-2:328–373, 2013.