Task and Motion Planning Methods: Applications and Limitations

Kai Zhang[1,2] [a], Eric Lucet[1] [b], Julien Alexandre Dit Sandretto[2] [c],
Selma Kchir[1] [d] and David Filliat[2,3] [e]

[1] *Université Paris-Saclay, CEA, List, F-91120, Palaiseau, France*

[2] *U2IS, ENSTA Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France*

[3] *FLOWERS, INRIA, ENSTA Paris, Institut Polytechnique de Paris, 91120 Palaiseau, France*

Keywords: Task and Motion Planning, Simulation Environment, Learning Methods.

Abstract: Robots are required to perform more and more complicated tasks, which raises the requirement of more intelligent planning algorithms. As a domain having been explored for decades, task and motion planning (TAMP) methods have achieved significant results, but several challenges remain to be solved. This paper summarizes the development of TAMP from solving objectives, simulation environments, methods and remaining limitations. In particular, it compares different simulation environments and methods used in different tasks aiming to provide a practical guide and overview for the beginners.

1 INTRODUCTION

With the development of manufacturing and software technology, robots are playing a more and more important role in our society. For example, we can see them in the factories to assist or replace people in the dangerous and tedious work. To enhance our life experience, they come to our life as autonomous cars, housekeepers, etc. However, their competence on this kind of tasks is not always convincing since they are not as intelligent as expected. The essential reason is that the human environment is unstructured and dynamic, which is more complicated than the structured factory environment. As a consequence, the tasks of the robots are more difficult in human environment, like household affairs. When a robot performs a task, firstly it needs to find feasible plans to accomplish the goal, then during executing the plan, it has to consider the surrounding complex and changing environment before stepping ahead. This process can be summarized as two steps, task planning and motion planning.

Task planning aims to compute solvable plans to complete a long-horizon task. It usually decomposes a long-horizon task into some short-horizon and elementary subtasks. For example, when a robot is ordered to fetch some object in a room with door closed, after decomposition, it can complete the task by solving several simple tasks, including opening the door, searching the object and returning. Hence, the challenge in this step is to decompose the complicate task into several simple subtasks.

Motion planning focuses on converting a subgoal into a sequence of parameters so that the software of the robot can control the hardware parts to reach the subgoal. For example, the "open door" task is transformed into some parameters to control the joints of robot's arm so that the end-effector could touch and push the door. Due to the constraints of the environment, it is often challenging to generate applicable control parameters that will avoid colliding with other objects.

Although it seems that task planning and motion planning share some similar designs, they are operated in different spaces. Task planning is usually considered as planning in a discrete space while the motion planning is taken in continuous space. Great progress has been made to integrate the discrete and continuous planning methods to solve TAMP problems. Recently, an overview paper (Garrett et al., 2021) focused on the integration of TAMP, summarizes different kinds of methods to solve multi-model motion planning and TAMP. It provides general concepts but the scope focuses on the operator-based methods operating in fully-observable environments,

[a] https://orcid.org/0000-0003-1129-9944
[b] https://orcid.org/0000-0002-9702-3473
[c] https://orcid.org/0000-0002-6185-2480
[d] https://orcid.org/0000-0003-3047-6846
[e] https://orcid.org/0000-0002-5739-1618

Zhang, K., Lucet, E., Sandretto, J., Kchir, S. and Filliat, D.
Task and Motion Planning Methods: Applications and Limitations.
DOI: 10.5220/0011314000003271
In *Proceedings of the 19th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2022)*, pages 476-483
ISBN: 978-989-758-585-2; ISSN: 2184-2809

which is far from the real applications. Besides, it demonstrates the solution of TAMP problems in a theoretical way, which is not user-friendly to beginners who want to get into this field by practicing. Therefore, this paper intends to provide a practical and broader overview for readers to easily start applying the TAMP methods to solve different tasks.

The organization of this paper is as follows: after the introduction, the background knowledge on TAMP is introduced in section 2. Section 3 describes the popular tasks solved by TAMP methods and some available simulation environments. Besides, the recent TAMP methods are compared and some limitations are pointed out. Finally, section 4 concludes this paper and proposes some potential research directions in this field.

2 BACKGROUND

Let's first present task and motion planing separately. Task planning usually works in a higher-level discrete state space, giving a global plan, while motion planning aims to follow such guidance in a lower-level action space.

2.1 Task Planning

Given an initial state and a global task, task planning aims to generate a sequence of intermediate elementary tasks or abstract actions to guide the agent to accomplish the original complicate task.

Depending on the types of tasks, the predefined actions of a robot could be discrete actions or continuous actions. Discrete actions contain a finite set of options that the agent can choose to apply, such as move left or right. Continuous actions are configured with a value, such as the rotation of a robot base, where there is an infinite choice of actions. For example, 60 degrees clockwise rotation is different from 60.1 degrees clockwise rotation. This second situation is more complex to deal with as the search space is much larger.

Lots of efforts have been made by robotics researchers and several planning methods have been proposed, such as hierarchical methods (Kaelbling and Lozano-Pérez, 2013), heuristic searching methods, operator planning methods, etc. A more detailed overview and discussion can be found in the introduction book (LaValle, 2006). Due to the simplicity and efficiency, they are widely used in the decision making games like chess, Tower of Hanoi, etc.

Instead of the handcrafted methods, reinforcement learning (RL) methods learn policies that map the observations to subgoals by maximizing a numerical reward signal. By default, these methods learn the solution to a single task, hence they are not solving the full planning problem

2.2 Motion Planning

Motion planning can be considered to bridge the low-level control parameters and the high-level tasks. Given a feasible task, the motion planning algorithm would generate a series of concrete parameters to achieve the task. For example, in the navigation task, given a goal position, motion planning algorithm will generate a trajectory so that the robot could follow the trajectory to reach the goal without collision.

Several algorithms have been proposed for motion planning, such as the shortest path searching methods in navigation task, or inverse-kinematic methods in manipulation task. A more detailed introduction can be found in Ghallab's planning book (Ghallab et al., 2016). Besides, learning methods, especially the RL methods have drawn lots of attention for intelligent motion planning. Some examples can be found in (Sun et al., 2021).

Apart from the previous passive motion planning algorithms, which focus on satisfying the predefined collision constraints, an active motion planner could consider the context of local environment before making a plan. For instance, a context-aware costmap is generated by integrating several semantic layers in (Lu et al., 2014), each of which describes one type of obstacle or constraint, including mobile and static obstacles, or dangerous regions. Planning on the context-aware costmap could produce a practical and intelligent trajectory. Moreover, in (Patel et al., 2021), an active obstacle avoidance method is introduced, where the robot intends to avoid humans from its back region.

3 TASK AND MOTION PLANNING

TAMP is the integration of task planning and motion planning. In other words, it links the planning in discrete space and continuous space. In this section, we highlight the common TAMP problems and methods categorized by whether they use deep learning techniques. In contrast to (Garrett et al., 2021), which focuses on symbolic operator based planning methods, we extend it to a broader view, including end-to-end learning methods for TAMP. Besides, we present a comparison of related global tasks and experimental environments.

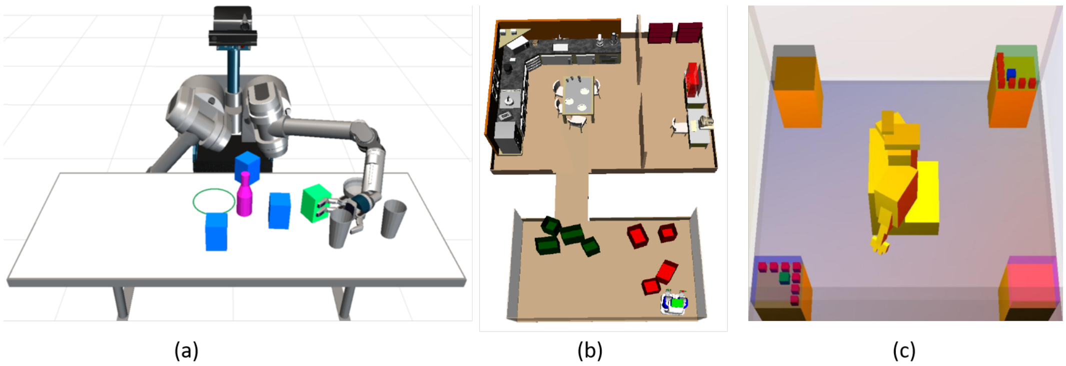

Figure 1: Demonstration of tasks. (a) Rearrangement task. The robot needs to push the green box from its start pose to the goal region indicated by the green circle (King et al., 2016). (b) Navigation among movable obstacles. The robot needs to remove the green obstacles before moving the red boxes to the kitchen region(Kim et al., 2019). (c) Pick-Place-Move task. The robot needs to pick the blue cube and place it in the box containing green cube (Garrett et al., 2015).

3.1 Objectives

There are various global tasks for TAMP in human environment but most of them could be regarded as the combination of basic tasks. We believe that if the TAMP methods could deal well with the basic tasks, they could be generalized to solve more complex global tasks. Three of the fundamental tasks are described as follows:

- Rearrangement (Re). As shown in Figure 1(a), the robot needs to manipulate several objects so that it could reach a target object without collision. The rearrangement task for multiple robots requires the collaboration among robots, which usually happens when a robot's arm cannot reach some regions of the environment due to its physical limitation (Driess et al., 2020).
- Navigation among movable obstacles (NAMO). Different from pure navigation task, NAMO requires the robot to interact with environment during navigation to reach the goal position. The interaction aims at clearing the obstacles actively so that a blocked trajectory becomes feasible. An example can be found in Figure 1(b), where the robot should remove the obstacles before entering the kitchen.
- Pick-Place-Move (PPM) task. As shown in Figure 1(c), the primitive operations of the robot are to pick up an object, move and place it in a box. Furthermore, the PPM task can serve for the Assembly and/or Disassembly task, in which the order of manipulated object should be considered.

3.2 Methods

After introducing the global tasks, we describe the TAMP methods corresponding to these objectives into three categories, namely classical methods, learning methods and hybrid methods that combine the previous two categories.

3.2.1 Classical Methods

The classical methods mainly include two types of methods, sampling-based and optimization-based methods. Given a long-horizon task with description of initial and final state, sampling methods could sample several useful intermediate states from the continuous infinite state space. Afterward, the searching methods are used to find a sequence of feasible transition operators between the intermediate states. The frequently adopted searching-based sampling methods include heuristic search, forward search, or backward search. An overview on searching methods can be found in (Ghallab et al., 2016). With a sequence of operators, the classical motion planning methods, including RRT Connect(Kuffner and LaValle, 2000) for the robot base and inverse kinematics for the robot arm(Garrett et al., 2020b), are applied to transit robot's state to the target state.

However, sampling methods are usually not complete over all problem instances. First, they cannot generally identify and terminate infeasible instances. Second, sampling process can only be applied to the explored space, which means they cannot find solutions to instances that require identifying values from unknown space (Garrett et al., 2018). For example, in a partial observation case, the robot can only find a path to a waypoint within the range of observation. Third, when the task description is not lucid (for example, a pouring task where the goal is to pour as much milk as possible into the cup), the sampling methods tend to fail.

Accordingly, optimization based methods are proposed to compensate the sampling methods. The objective is primarily given in terms of a cost function along the temporal axis. An optimization strategy is applied to minimize the cost with respect to constraints and finally output the feasible solutions. The optimization method is ideal to solve the problems with continuous solutions since time axis is directly integrated in the objective function. Toussaint(Toussaint, 2015) uses this approach in a manipulation problem where a robot picks and places cylinders and plates on a table to assemble the highest possible stable tower. The action sequences are generated by a simple symbolic planning approach but the best final and intermediate positions of all the objects are found through optimization.

A comprehensive review on sampling methods and optimization methods to solve TAMP problems can be found in (Garrett et al., 2021).

Besides, there are also some TAMP methods with hand-crafted strategy. For example, (Meng et al., 2018) presents an active path cleaning algorithm for NAMO task. The proposed system integrates obstacle classification, collision detection, local environment reconstruction and interaction with obstacles. To solve the situation where the obstacle is unknown, an affordance-based method (Wang et al., 2020) is developed to help robot decide if the obstacle is movable by interacting with it.

3.2.2 Learning based Methods

In learning based methods, the robot acquires the skills from experiences. The most common framework is RL which learns a policy that maps a state of the environment to an action by reward and penalty (Driess et al., 2020). A TAMP problem usually contains a long-horizon task, which can be converted to sparse reward when the task is completed. However, exploring the environment through taking random actions requires a prohibitive number of samples until a solution can be found (Li et al., 2020). Therefore, hierarchical RL (HRL) has been proposed to solve the sparse reward problem by generating subtasks to guide the robot to accomplish final task (Barto and Mahadevan, 2003).

An intuitive idea of HRL is to design and train two networks, one is dedicated to high level task generation while the other one is for primitive motion control, as described in (Kulkarni et al., 2016), a top-level module learns a policy over subgoals and a bottom-level module learns actions to accomplish the objective of each subgoal. Considering the task dependence and generalization problem of previous methods, a task-independent method (Nachum et al., 2018) is designed by reformulating the task description. Instead of using the observation from the robot, they use the observation from the environment, like position and distance, to reduce the dependence on task.

Training the high-level policy and low-level action separately misses the ability of joint optimization. Hence, in (Levy et al., 2018), they describe a joint training strategy to learn the policy in three levels for a navigation task. The highest level takes in the current state and a task to generate subtasks, while the middle level decomposes a subtask to a visible goal. The lowest level generates action parameters to reach the goal. However, in a NAMO task, given a final position, the high-level subgoal creation network should not only generate subgoals for robot base but also the interaction position for arms. Accordingly, a HRL method is proposed (Li et al., 2020) to generate heterogeneous subgoals so that the robot could interact with the obstacles during navigation. To find an appropriate action to interact with various obstacles, a Neural Interaction Engine (Zeng et al., 2021) that predicts the action effect, is integrated to a policy generation network.

Although the learning methods have achieved satisfying results in simulation environment, the transfer from simulation to real applications is difficult because the trained models cannot be used directly in the real scenarios and under most circumstances, they should be retrained in the application environment. For example, in the solution proposed in (Li et al., 2020), the trained model maps the sensor data to actions. However, due to the large difference between environments, the change of sensor data could lead to strange actions. Moreover, the training data in real environment is expensive, hence we can see few real applications relying on pure learning methods.

3.2.3 Hybrid Methods

Although both the classical methods and learning based methods can solve several TAMP tasks, they suffer some limitations. For example, the operators used in sampling methods are usually designed manually, which is time-consuming and tends to be very task-specific. The learning methods avoid the manual work but they offer less freedom to add extra constraints, like no collision tolerance. Besides, the transferability of learning methods from simulation to real environment is proved difficult since the expensive cost of constructing the training dataset and the inaccurate representation of environment, which might be caused by sensor noise, illumination, occlusion, etc.

Therefore, some researchers adopt hybrid strategies, such as learning symbolic operators from dataset (Silver et al., 2021; Pasula et al., 2007; Konidaris et al., 2018), learning to guide the operator search(Kim et al., 2019; Kim and Shimanuki, 2020)

Table 1: List of TAMP methods on three tasks.

	Classical methods	Learning based methods	Hybrid methods
Re	(Toussaint, 2015) (Garrett et al., 2020b)	(Driess et al., 2020)	(Chitnis et al., 2016) (Wang et al., 2021)
NAMO	(Meng et al., 2018) (Wang et al., 2020)	(Li et al., 2020) (Zeng et al., 2021)	(Kim and Shimanuki, 2020) (Xia et al., 2021)
PPM	(Kaelbling and Lozano-Pérez, 2013) (Garrett et al., 2015)		(Kim et al., 2019) (Konidaris et al., 2018) (Garrett et al., 2018)

or learning to generate feasible subgoals(Xia et al., 2021).

Learning symbolic operators from a dataset provides the primitive skills for the task planning. With the operators, a conventional tool such as PDDL(Ghallab et al., 1998) or its extension (Garrett et al., 2020a) is applied to search the feasible plans. Then, the motion planning algorithm could directly converts the primitive operators to executable control parameters. A supervised learning strategy is introduced in (Pasula et al., 2007) to learn the symbolic operators from a training dataset. Each training example contains the current state, an action and the state after applying the action. An action model is trained by maximizing the likelihood of the action effects, subject to a penalty on complexity. To reduce the requirement for an expensive training dataset, a learning-from-experience method (Konidaris et al., 2018) applies actions to the agent and obtains the states through the experience. Then, it converts the continuous states into a decision tree, and finally into symbolic operators.

Given a large problem that contains lots of actions and states, the classical searching methods are less efficient since the search space is too large. Instead of traversing the whole space to find a solution, reinforcement learning methods provide an efficient way to learn the searching strategy from experience(Chitnis et al., 2016). In (Kim et al., 2019), a graph is taken as the searching space due to its extensibility. The nodes are abstract actions while the edges are the priority of transition. A Q-value function is learned from a training dataset to calculate the priority of actions, which provides guidance for efficient searching. Apart from the guidance of discrete searching, in continuous action space, they apply a generative model to generate multiple feasible candidates to avoid being blocked by an infeasible solution (Kim et al., 2021). Similarly, a model is applied to a dataset to learn the probability of success(Wang et al., 2021). Then, in the same domain but a new scenario, given the action, the model predicts a success rate. By picking the actions with a higher success rate, the searching space is significantly reduced.

In addition to the operator based methods, a few methods are proposed to directly generate the subtasks based on RL methods. With a feasible subtask, classical motion planning methods are used to control the robots. In a NAMO task, a Soft Actor Critic (Haarnoja et al., 2018) algorithm is applied to generate the subgoals for the arm and the base of a robot (Xia et al., 2021) through the observation of environment. Subsequently, RRT connect (Kuffner and LaValle, 2000) and inverse kinematics methods are employed to reach the subgoals.

In summary, the hybrid methods usually apply learning to task planning, or a part of the task planning process, then classical motion control algorithms are adopted to generate control parameters. This strategy benefits from better transferability to real application than pure learning algorithms and provides more efficient strategies than classical methods.

Table 1 provides an overview of the application of the presented methods on the basic tasks proposed in section 3.1.

3.3 Environments and Tools

When developing robotic algorithms, the validation of interaction results is an essential step. Testing the effects of interactions in a real environment is a straightforward approach, but it can be time-consuming, expensive, unstable and potentially unsafe. Therefore, several interactive simulation environments are recently proposed to advance the robotic research and facilitate the experiments.

In this subsection, we compare several interactive simulation environments that are designed for navigation and manipulation tasks, which include iGibson2(Li et al., 2021), AI2THOR(Kolve et al., 2017), TDW(Gan et al., 2021), Sapien(Xiang et al., 2020), Habitat2(Szot et al., 2021) and VirtualHome(Puig et al., 2018). Different from the low-level view that focuses on which type of rendering they use, we pay attention to their usability for TAMP tasks and transferability to real environment.The comparison results can be found in Table 2.

Table 2: Comparison among different interactive simulation environments.

		iGibson2	AI2THOR	TDW	Sapien	Habitat2	VirtualHome
Provided environment		15 homes (108 rooms)	120 rooms	-	-	-	build from 8 rooms
Interactive objects		1217	609	200	2346	-	308
ROS support		✓	×	×	✓	✓	×
Uncertainty support		✓	×	×	×	✓	×
Supported tasks	Re	+	+++++	++++	++++	++++	++++
	NAMO	+++++	++	+++	+++	++	++++
	PPM	+++++	+++++	++++	++++	+++++	+++++
Speed		GPU ++	++	++	+++	++++	++
Sensors		RGBD, Lidar	RGBD	RGBD	RGBD	RGBD	RGBD

3.4 Challenges

Although TAMP methods have been explored for decades, they are still not robust and face limitations in practical applications. In this section, we present several potential directions of improvement.

3.4.1 Observation Uncertainty

Observation uncertainty is usually caused by the sensor noise, which is unavoidable in real applications. There are mainly two kinds of solutions, (a) modeling the noise and reducing it through multiple observation; (b) using learning methods to directly map the noisy data with actions.

An operator-based TAMP method is presented in (Kaelbling and Lozano-Pérez, 2013) to solve the observation uncertainty in PPM task. The uncertainty appears in the localization of the robot and the target object. They ask the robot to observe the object multiple times and use Gaussian model to approximate the noise of localization. In (Driess et al., 2020), the raw sensor data is directly inputted to a neural network aiming to map raw observation data to action sequence through reward optimization. The approach is simple since it doesn't require a complex modeling process but requires a large amount of training scenes, 30000 in their experiment.

In summary, although the previous methods complete the task with observation uncertainty, their experiment environment is quite simple, giving the robot a large free space of manipulation. Therefore, it raises the questions of their practicability in a constrained and complex environment to finish household works, and their efficiency to find a feasible solution.

3.4.2 Action Uncertainty

With the same precondition and symbolic operator, an action may produce different effects. For example, *pick* action may indicate the grasp of the object from its top or from its side. This ambiguity may lead to failure when trying to place the object steadily. In a PPM task described in (Silver et al., 2021), the robot needs to pick an object and place it in a shelf, which demands the robot to choose appropriate action of picking since the space is narrow under the ceiling of the shelf. They collect a dataset, from which they obtain several kinds of pick operators, like picking from side and picking from top. The solution is found through backtracking because the robot could infer suitable pick action from the goal state. However, backtracking requires the full observation of environment, which usually cannot be satisfied in real applications.

3.4.3 Situational Mapping

A real task tends to be more complex and the robot needs to deduce the solution by considering the semantic information of the environment. For example, imagine a blocks building task, with different kinds of blocks and the objective to assemble a car model. Without considering the type or shape information of each block and the car, it is impossible to complete the assembly task.

Situational analysis and mapping could benefit various domains, including safe navigation, action verification, understanding of ambiguous task, etc. For example, to achieve safe navigation, situational mapping allows robot to build respective danger zone according to the characteristics of obstacles. The danger zone is relative small for the static obstacles, like walls, desks, while it is large for the mobile obstacles, like humans, vehicles. Specifically, the shape of danger zone is related to the moving direction and velocity of mobile obstacles. In (Samsani and Muhammad, 2021), the real-time behavior of humans are analyzed to generate the danger zone to guarantee the safe navigation in crowed scenes.

4 CONCLUSION

This paper reviews the recent development of TAMP, including the popular tasks, practical simulation environments, methods and existing challenges. Three fundamental tasks, including rearrangement, navigation among movable obstacles and Pick-Place-Move task, are described. Besides, some popular simulation environments are listed and compared to facilitate the choice of experiment environment. What's more, some TAMP methods are classified by whether they use deep learning methods and their tasks, which helps readers to start from a baseline according to the problems encountered and their background knowledge. Finally, we describe the existing problems aiming at indicating the possible exploration direction. In summary, algorithms that are robust to perception and action uncertainty and are able to exploit the environment semantics, should be explored.

ACKNOWLEDGEMENTS

This work was carried out in the scope of OTPaaS project. This project has received funding from the French government as part of the "Cloud Acceleration Strategy" call for manifestation of interest.

REFERENCES

Barto, A. G. and Mahadevan, S. (2003). Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1):41–77.

Chitnis, R., Hadfield-Menell, D., Gupta, A., Srivastava, S., Groshev, E., Lin, C., and Abbeel, P. (2016). Guided search for task and motion plans using learned heuristics. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 447–454. IEEE.

Driess, D., Ha, J.-S., and Toussaint, M. (2020). Deep visual reasoning: Learning to predict action sequences for task and motion planning from an initial scene image. In *Robotics: Science and Systems 2020 (RSS 2020)*. RSS Foundation.

Gan, C., Schwartz, J., Alter, S., Mrowca, D., Schrimpf, M., Traer, J., De Freitas, J., Kubilius, J., Bhandwaldar, A., Haber, N., et al. (2021). Threedworld: A platform for interactive multi-modal physical simulation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Garrett, C. R., Chitnis, R., Holladay, R., Kim, B., Silver, T., Kaelbling, L. P., and Lozano-Pérez, T. (2021). Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4:265–293.

Garrett, C. R., Lozano-Pérez, T., and Kaelbling, L. P. (2015). Ffrob: An efficient heuristic for task and motion planning. In *Algorithmic Foundations of Robotics XI*, pages 179–195. Springer.

Garrett, C. R., Lozano-Pérez, T., and Kaelbling, L. P. (2018). Sampling-based methods for factored task and motion planning. *The International Journal of Robotics Research*, 37(13-14):1796–1825.

Garrett, C. R., Lozano-Pérez, T., and Kaelbling, L. P. (2020a). Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 30, pages 440–448.

Garrett, C. R., Paxton, C., Lozano-Pérez, T., Kaelbling, L. P., and Fox, D. (2020b). Online replanning in belief space for partially observable task and motion problems. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5678–5684. IEEE.

Ghallab, M., Howe, A., Knoblock, C., Mcdermott, D., Ram, A., Veloso, M., Weld, D., and Wilkins, D. (1998). PDDL—The Planning Domain Definition Language.

Ghallab, M., Nau, D., and Traverso, P. (2016). *Automated planning and acting*. Cambridge University Press.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. (2018). Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*.

Kaelbling, L. P. and Lozano-Pérez, T. (2013). Integrated task and motion planning in belief space. *The International Journal of Robotics Research*, 32(9-10):1194–1227.

Kim, B. and Shimanuki, L. (2020). Learning value functions with relational state representations for guiding task-and-motion planning. In *Conference on Robot Learning*, pages 955–968. PMLR.

Kim, B., Shimanuki, L., Kaelbling, L. P., and Lozano-Pérez, T. (2021). Representation, learning, and planning algorithms for geometric task and motion planning. *The International Journal of Robotics Research*, page 02783649211038280.

Kim, B., Wang, Z., Kaelbling, L. P., and Lozano-Pérez, T. (2019). Learning to guide task and motion planning using score-space representation. *The International Journal of Robotics Research*, 38(7):793–812.

King, J. E., Cognetti, M., and Srinivasa, S. S. (2016). Rearrangement planning using object-centric and robot-centric action spaces. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3940–3947. IEEE.

Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., Gordon, D., Zhu, Y., Gupta, A., and Farhadi, A. (2017). Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*.

Konidaris, G., Kaelbling, L. P., and Lozano-Perez, T. (2018). From skills to symbols: Learning symbolic representations for abstract high-level planning. *Journal of Artificial Intelligence Research*, 61:215–289.

Kuffner, J. J. and LaValle, S. M. (2000). Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 2, pages 995–1001. IEEE.

Kulkarni, T. D., Narasimhan, K., Saeedi, A., and Tenenbaum, J. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *Advances in neural information processing systems*, 29.

LaValle, S. M. (2006). *Planning algorithms*. Cambridge university press.

Levy, A., Konidaris, G., Platt, R., and Saenko, K. (2018). Learning multi-level hierarchies with hindsight. In *International Conference on Learning Representations*.

Li, C., Xia, F., Martín-Martín, R., Lingelbach, M., Srivastava, S., Shen, B., Vainio, K. E., Gokmen, C., Dharan, G., Jain, T., et al. (2021). igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. In *5th Annual Conference on Robot Learning*.

Li, C., Xia, F., Martin-Martin, R., and Savarese, S. (2020). Hrl4in: Hierarchical reinforcement learning for interactive navigation with mobile manipulators. In *Conference on Robot Learning*, pages 603–616. PMLR.

Lu, D. V., Hershberger, D., and Smart, W. D. (2014). Layered costmaps for context-sensitive navigation. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 709–715. IEEE.

Meng, Z., Sun, H., Teo, K. B., and Ang, M. H. (2018). Active path clearing navigation through environment reconfiguration in presence of movable obstacles. In *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 156–163. IEEE.

Nachum, O., Gu, S. S., Lee, H., and Levine, S. (2018). Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31.

Pasula, H. M., Zettlemoyer, L. S., and Kaelbling, L. P. (2007). Learning symbolic models of stochastic domains. *Journal of Artificial Intelligence Research*, 29:309–352.

Patel, U., Kumar, N. K. S., Sathyamoorthy, A. J., and Manocha, D. (2021). Dwa-rl: Dynamically feasible deep reinforcement learning policy for robot navigation among mobile obstacles. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6057–6063. IEEE.

Puig, X., Ra, K., Boben, M., Li, J., Wang, T., Fidler, S., and Torralba, A. (2018). Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502.

Samsani, S. S. and Muhammad, M. S. (2021). Socially compliant robot navigation in crowded environment by human behavior resemblance using deep reinforcement learning. *IEEE Robotics and Automation Letters*, 6(3):5223–5230.

Silver, T., Chitnis, R., Tenenbaum, J., Kaelbling, L. P., and Lozano-Pérez, T. (2021). Learning symbolic operators for task and motion planning. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3182–3189. IEEE.

Sun, H., Zhang, W., Runxiang, Y., and Zhang, Y. (2021). Motion planning for mobile robots–focusing on deep reinforcement learning: A systematic review. *IEEE Access*.

Szot, A., Clegg, A., Undersander, E., Wijmans, E., Zhao, Y., Turner, J., Maestre, N., Mukadam, M., Chaplot, D., Maksymets, O., Gokaslan, A., Vondrus, V., Dharur, S., Meier, F., Galuba, W., Chang, A., Kira, Z., Koltun, V., Malik, J., Savva, M., and Batra, D. (2021). Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Toussaint, M. (2015). Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

Wang, M., Luo, R., Önol, A. Ö., and Padir, T. (2020). Affordance-based mobile robot navigation among movable obstacles. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2734–2740. IEEE.

Wang, Z., Garrett, C. R., Kaelbling, L. P., and Lozano-Pérez, T. (2021). Learning compositional models of robot skills for task and motion planning. *The International Journal of Robotics Research*, 40(6-7):866–894.

Xia, F., Li, C., Martín-Martín, R., Litany, O., Toshev, A., and Savarese, S. (2021). Relmogen: Integrating motion generation in reinforcement learning for mobile manipulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4583–4590. IEEE.

Xiang, F., Qin, Y., Mo, K., Xia, Y., Zhu, H., Liu, F., Liu, M., Jiang, H., Yuan, Y., Wang, H., et al. (2020). Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107.

Zeng, K.-H., Weihs, L., Farhadi, A., and Mottaghi, R. (2021). Pushing it out of the way: Interactive visual navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9868–9877.