Learning Hierarchical Abstractions for Efficient Taskable Robots – Dissertation Abstract

Naman Shah
Advisor: Prof. Siddharth Srivastava
School of Computing and Augmented Intelligence (SCAI),
Arizona State University,
Tempe, AZ, USA, 85281
shah.naman@asu.edu

Abstract

Although state-of-the-art hierarchical robot planning algorithms allow robots to efficiently compute long-horizon motion plans for achieving user desired tasks, these methods typically rely upon environment-dependent state and action abstractions that need to be hand-designed by experts. On the other hand, non-hierarchical robot planning approaches fail to compute solutions for complex tasks that require reasoning over a long horizon. My research addresses these problems by proposing an approach for learning abstractions and developing hierarchical planners that efficiently use learned abstractions to boost robot planning performance while providing strong guarantees of reliability.

1 Introduction

Recent years have seen a sharp increase in the usage of robots in various areas such as manufacturing, household chores, and delivery. Such robots interact with their environments by moving their links around. To efficiently interact with their environments, a robot needs to compute a trajectory or a motion plan that takes the robot from its current configuration to its desired configuration. Generally, robots use sampling-based motion planners such as RRT (LaValle 1998) and PRM (Kavraki et al. 1996) to compute these motion plans that excel at computing short-horizon motion plans between pairs of configurations.

Complex tasks such as arranging a room or delivering items to different locations require robots to deal with changing configuration spaces and complex motion plans, requiring robots to reason over a long horizon. Typically, sampling-based motion planners fail to perform well in reasoning over a long horizon to compute such complex motion plans due to the infinite branching factor of the configuration space. This prevents robots from autonomously solving such complex tasks.

On the other hand, humans excel at such tasks which require extended reasoning owing to their capacity of abstracting information about the task. E.g, consider a human that has to arrange a dining table. To accomplish this task, humans would not think which joints they would move and by how much. However, they would think in high-level abstract actions such as "pick up the plate from the counter", "place the plate on the table", "place glass on the table", etc. Similarly, a person that has to reach the kitchen from a room (Fig. 1) in a house would "plan" using high-level actions such as "exit the door", "pass through the corridor", and "enter the kitchen". Such abstract actions allow humans to reason over a long horizon without considering the intricacies of the domain.

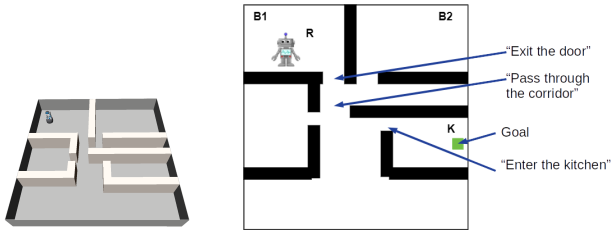
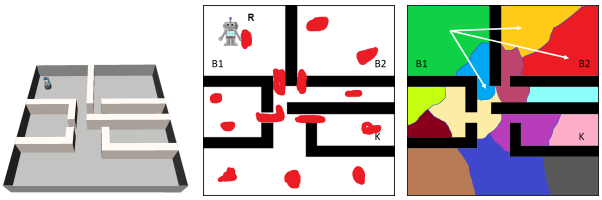


Figure 1: (a) A 3D model of a real life house hold environment. (b) A 2D map of the 3D environment. R shows the current position of the robot. Green rectangle shows the target position of the robot. The arrows show the hypothetical high-level actions that the robot must take to reach its goal from the current position that my work aims to learn automatically.

Hierarchical planning systems such as combined task and motion planning frameworks (Srivastava et al. 2014; Dantam, Kingston, and Chaudhuriand L. Kavraki 2018; Garrett, Lozano-Pérez, and Kaelbling 2020; Shah et al. 2020) use such high-level actions to guide the low-level (concrete) motion planning. These systems use hand-coded abstractions to generate high-level task specifications for these robot planning problems and use them to perform hierarchical planning. A domain expert is required to write these hand-coded abstractions, which limits the scope of the domains where these approaches can be applied.

Through my work, I aim to answer the following two crucial research questions: 1) Can we automatically learn hierarchical state and action abstractions for new environments and 2) can we efficiently use these abstractions to perform hierarchical robot planning? As part of my thesis, I aim to develop a set of approaches that answer these questions by automatically identifying hierarchical state and action abstractions for new environments and using them to perform hierarchical planning with various robots.



(a) (b) (c)

Figure 2: (a) An illustrative environment for a motion planning problem. The robot (R) is tasked to reach the kitchen (K). Red blobs in (b) show a set of candidate critical regions in the environment. Lastly, (c) shows an example of state abstraction. Each colored cell represents an abstract state. White arrows show a few abstract actions that take the robot from one abstract state to another abstract state.

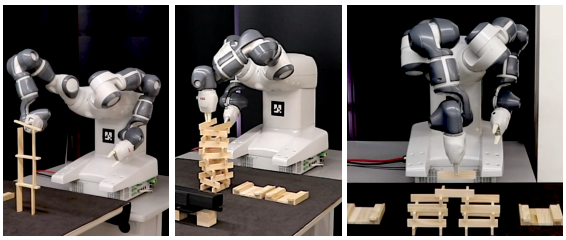


Figure 3: ABB YuMi builds Keva structures using a STAMP policy: 3π (left), twisted 12-level tower (center), and 3-towers (right).

Now in the rest of the paper, I present the approach that I would propose as part of my thesis, show preliminary results using the approach, and discuss a few related approaches.

2 Proposed Approach

To answer the research questions outlined in Sec. 1, I propose to learn hierarchical state and action abstractions by identifying regions in the environment that are critical for solving the given class of motion planning problems. E.g., consider a household environment (Fig. 2(a)).Here, the robot is currently in room B1. If the robot is tasked to bring a bottle of water from the kitchen (K) to the room (B1), then every motion plan accomplishing this task must pass through the doors and the passage. All these motion plans must take the robot to a configuration from which the robot is able to grasp the bottle and also to a configuration where the robot is holding the bottle in its gripper. Fig. 2(b) shows a few candidate critical regions for the given environment. This implies that these regions are loosely similar *landmarks* in the symbolic planning literature. But, contrary to landmarks, these regions are not a necessary condition to reach the goal. Molina, Kumar, and Srivastava (2020) define such regions as *critical regions* as proposes a method for identifying critical regions in an environment using a DNN.

I propose to learn hierarchical state and action abstractions using such automatically identified critical regions. Precisely, abstract states can be identified by constructing regions around these critical regions. Similarly, abstract actions can be automatically identified as transitions between these abstract states (similar to Fig. 2(c)). Once the abstract states and actions are identified, we can use them with a high-level planner to compute a high-level plan that can be refined into a motion plan using hierarchical planning.

One major technical challenge in this approach would be that the heuristic used to perform high-level planning would not be very informative. It would also fail to identify actions that the low-level planner would fail to refine due to obstacles in the environment. To overcome this issue, I propose to use a multi-source algorithm for high-level planning. Now, typically multi-source approaches can not be used with robot planning problems as we do not have any information about what the intermediate states would be. But, as we are identifying high-level abstract states automatically using critical regions, we can use these abstract states as candidate intermediate states for multi-source search.

On the other hand, a probabilistically-complete interleaved approach that tries to search for a high-level plan which has low-level refinements can also be used to perform hierarchical planning with such imprecise and lossy abstractions. This interleaved search approach would search for motion planning refinements for high-level actions while continually updating the high-level abstractions to compute accurate high-level solutions.

Now, I discuss some of the approaches that we have developed using these methods and their preliminary results.

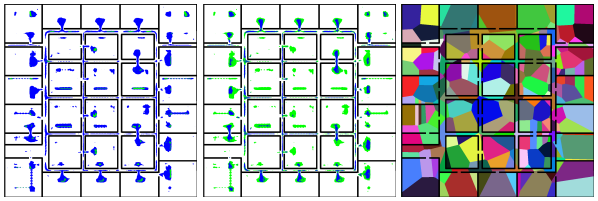
3 Preliminary Results

We developed two algorithms that use the methods discussed in the previous approach for solving robot planning problems. The first method (Shah et al. 2020) uses interleaved search for performing combined task and motion planning in stochastic environments (Sec. 3.1) and the second approach -- ***H**ierarchical **A**bstraction-guided **R**obot **P**lanning (HARP)* (Shah and Srivastava 2022) -- performs hierarchical robot planning using automatically identified abstract states and actions and a multi-source planning algorithm (Sec. 3.2).

3.1 Stochastic Task and Motion Planning

Our work (Shah et al. 2020) presents an anytime *probabilistically complete* framework using the approach outlined in the previous section. It uses entity abstraction and stochastic shortest path (SSP) problems to formulate a *STAMP* problem for robots with stochastic actions. It performs an interleaved search to compute a high-level policy that also has valid low-level motion planning refinements. We use concretization functions called *generators* to refine each abstract action in the high-level solution. Our algorithm also continually updates the abstraction to generate more accurate high-level solutions if any action in the current high-level policy fails to admit a low-level refinement.

We evaluate our work in multiple settings where combined task and motion planning is necessary to compute feasible solutions. Refining each possible outcome in the solution tree can take a substantial amount of time. Here, we reduce the problem of selecting scenarios for refinement to a knapsack problem and use a greedy approach to prioritize



(a) (b) (c)

Figure 4: Critical regions and generated abstraction for a 4-DOF hinged robot. (a) and (b) shows the CRs predicted by the model. Blue regions in (a) show that model predicted the robot's base link to be horizontal while green regions show that the model predicted the robot's base link to be vertical. Blue regions in (b) show that the network predicted the hinge to be closer to $\mathbf{180}^\circ$ and green regions show that the network predicted it to be closer to $\mathbf{90}^\circ$ or $\mathbf{270}^\circ$. (c) shows 2D projection of the state abstraction generated by our approach though our approach does not need to explicitly generate these abstractions.

more likely outcomes for refinement. Our empirical evaluation shows that doing so allows the robot to start executing an action much earlier. Fig. 3 shows one of the test domains where the YuMi robot uses the task and motion policies computed using our framework to construct geometric structures using Keva planks.

3.2 Learning and Using Abstractions for Robot Planning

In this work (Shah and Srivastava 2022), we automatically learn hierarchical state and action abstractions and use them with a novel probabilistically-complete hierarchical planner to solve motion planning problems. These state and action abstractions are generated using automatically identified critical regions as described in Sec. 2. We use a custom robot-specific deep neural network to learn to identify critical regions in unseen environments for each degree of freedom of the robot and location of the robot's end-effector (base link for navigational problem) in the workspace. Our approach automatically generates this custom architecture using robot geometry and the number of degrees of freedom and using UNet (Ronneberger, Fischer, and Brox 2015) as the base architecture.

Once a set of critical regions is identified for a given configuration space and a robot, our approach generates abstract states by growing Voronoi cells around these critical regions. We call this structure a *region-based Voronoi diagram (RBVD)*. Each cell in an RBVD is an abstract state and transitions between these cells are abstract actions. We show that the abstractions generated using this approach are sound and fulfill the downward refinement property for holonomic robots.

We extend Beam search (Lowerre 1976) to develop a novel multi-source bidirectional high-level search algorithm to compute a set of high-level plans using the generated abstract states and actions. Our approach computes a set of high-level plans using this multi-source bidirectional Beam search and refines them simultaneously using a multi-source multi-directional Learn and Link Planner (LLP) (Molina, Kumar, and Srivastava 2020) while continually updating the heuristic for high-level search. We also prove that the overall planning algorithm is probabilistically complete.

We extensively evaluate our approach in a total of twenty different settings with four different robots that included holonomic as well as non-holonomic robots. Fig. 4 shows critical regions predicted by our learned model for a hinged robot with 4 degrees of freedom and abstract states generated by our approach. We empirically evaluate our approach against sampling-based motion planning algorithms such as RRT (LaValle 1998), PRM (Kavraki et al. 1996), and BiRRT (Kuffner and LaValle 2000) and learning-based motion planner (Molina, Kumar, and Srivastava 2020). Our exhaustive empirical evaluation shows that our approach that combines learning with hierarchical planning not only significantly outperforms state-of-the-art sampling-based motion planners but also outperforms learning-based LLP which does generate hierarchical abstractions and does not perform hierarchical planning.

This work learns state and action abstractions for motion planning problems and robots with deterministic actions. In the future, we plan to extend this work for robots with stochastic action dynamics and learn high-level states and actions for combined task and motion planning problems. Lastly, I discuss a few approaches related to my existing work.

4 Related Work

Combined Task and Motion Planning Hierarchical planning with abstractions has been used in multiple ways with automated planning to improve the efficiency of planners in order to compute plans that achieve complex goals. Some of the earliest planning systems such as ABSTRIPS (Sacerdoti 1974) and ALPINE (Knoblock 1990) used abstractions with symbolic planning problems defined in STRIPS representations to perform hierarchical planning. Approaches such as FF (Hoffmann 2001), HSP2.0 (Bonet and Geffner 2001), and GraphPlan (Blum and Furst 1997) uses abstractions to solve a relaxed problem in order to automatically synthesize heuristics for planning.

A significant number of approaches have been developed to solve task and motion planning (TAMP) problems in recent years. Major works on TAMP can be categorized into three categories: *1)* approaches that use symbolic planners to guide motion planning (Cambon, Alami, and Gravot 2009), *2)* approaches that extend high-level representations to simultaneously search high-level plans along with continuous parameters (Garrett, Lozano-Pérez, and Kaelbling 2020), and *3)* approaches that use interleaved search for valid high-level plans with low-level refinements for its actions (Srivastava et al. 2014; Dantam, Kingston, and Chaudhuriand L. Kavraki 2018). These approaches focus on solving TAMP problems in deterministic environments where robots are expected to carry out their tasks accurately, while our approach provides a method to consider stochastic actions while computing task and motion policies to allow the robot to efficiently perform its tasks in the real world.

Learning for Motion Planning Multiple approaches use statistical learning to guide motion planning. Ichter, Harrison, and Pavone (2018) and Kumar et al. (2019) use *CVAE* to learn sampling distributions for motion planning. Molina, Kumar, and Srivastava (2020) use an image-based approach to learn sampling distributions for path-planning problems. These approaches use learning to bias the sampling distribution for sampling-based motion planning, while our approach focuses on learning state and action abstractions and using them efficiently to perform hierarchical planning.

Learning Abstractions While a lot of approaches (Blum and Furst 1997; Bonet and Geffner 2001; Knoblock 1990; Cambon, Alami, and Gravot 2009; Garrett, Lozano-Pérez, and Kaelbling 2020; Shah et al. 2020) have tried using abstractions to perform automated planning, not a lot of them have tried learning it. Konidaris, Kaelbling, and Lozano-Perez (2018) propose an approach that learns high-level action descriptions of low-level behaviors by computing sets of regions reachable by those behaviors. Contrary to their approach that requires a detailed description of reachability for low-level composite actions which are extremely difficult to obtain for complex AI agents, my thesis aims to develop an approach that solely requires low-level observations.

Chitnis et al. (2020) use CNNs to learn context-specific regions of the state space. Silver et al. (2020) uses GNNs to predict importance for each object in the environment for classical planning problems. These approaches use abstraction to reduce the planning space, but they still need handcrafted action descriptions in order to perform planning. Silver et al. (2021) propose an approach that uses low-level transitions along with high-level vocabulary to learn the symbolic representation of these actions for combined task and motion planning. Their approach requires an abstraction that is sufficient to represent low-level actions that the proposed work aims to learn.

References

Blum, A. L.; and Furst, M. L. 1997. Fast planning through planning graph analysis. *Artificial intelligence*, 90(1-2): 281–300.

Bonet, B.; and Geffner, H. 2001. Heuristic search planner 2.0. *AI Magazine*, 22(3): 77–77.

Cambon, S.; Alami, R.; and Gravot, F. 2009. A hybrid approach to intricate motion, manipulation and task planning. *IJRR*, 28: 104–126.

Chitnis, R.; Silver, T.; Kim, B.; Kaelbling, L. P.; and Lozano-Perez, T. 2020. CAMPs: Learning Context-Specific Abstractions for Efficient Planning in Factored MDPs. *arXiv preprint arXiv:2007.13202*.

Dantam, N.; Kingston, Z.; and Chaudhuriand L. Kavraki, S. 2018. An incremental constraint-based framework for task and motion planning. *IJRR*, 37(10): 1134–1151.

Garrett, C.; Lozano-Pérez, T.; and Kaelbling, L. 2020. PDDLStream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning. In *Proc. ICAPS*.

Hoffmann, J. 2001. FF: The fast-forward planning system. *AI magazine*, 22(3): 57–57.

Ichter, B.; Harrison, J.; and Pavone, M. 2018. Learning sampling distributions for robot motion planning. In *Proc. ICRA*.

Kavraki, L. E.; Svestka, P.; Latombe, J.-C.; and Overmars, M. H. 1996. Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces. *IEEE transactions on Robotics and Automation*, 12(4): 566–580.

Knoblock, C. A. 1990. Learning Abstraction Hierarchies for Problem Solving. In *AAAI*, 923–928.

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 *Proc. ICRA, 2000.*

Kumar, R.; Mandalika, A.; Choudhury, S.; and Srinivasa, S. 2019. LEGO: Leveraging Experience in Roadmap Generation for Sampling-Based Planning. In *Proc. IROS*.

LaValle, S. M. 1998. Rapidly-Exploring Random Trees: A New Tool for Path Planning.

Lowerre, B. T. 1976. *The Harpy Speech Recognition System.* Carnegie Mellon University.

Molina, D.; Kumar, K.; and Srivastava, S. 2020. Identifying Critical Regions for Motion Planning using Auto-Generated Saliency Labels with Convolutional Neural Networks. In *Proc. ICRA*.

Ronneberger, O.; Fischer, P.; and Brox, T. 2015. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Proc. MICCAI, 2015*.

Sacerdoti, E. D. 1974. Planning in a hierarchy of abstraction spaces. *Artificial intelligence*, 5(2): 115–135.

Shah, N.; Kala Vasudevan, D.; Kumar, K.; Kamojjhala, P.; and Srivastava, S. 2020. Anytime Integrated Task and Motion Policies for Stochastic Environments. In *Proc. ICRA*.

Shah, N.; and Srivastava, S. 2022. Using Deep Learning to Bootstrap Abstractions for Hierarchical Robot Planning. *arXiv preprint arXiv:2202.00907*.

Silver, T.; Chitnis, R.; Curtis, A.; Tenenbaum, J. B.; Lozano-Pérez, T.; and Kaelbling, L. P. 2020. Planning with Learned Object Importance in Large Problem Instances using Graph Neural Networks. *CoRR*, abs/2009.05613.

Silver, T.; Chitnis, R.; Tenenbaum, J.; Kaelbling, L. P.; and Lozano-Pérez, T. 2021. Learning Symbolic Operators for Task and Motion Planning. *arXiv preprint arXiv:2103.00589*.

Srivastava, S.; Fang, E.; Riano, L.; Chitnis, R.; Russell, S.; and Abbeel, P. 2014. A Modular Approach to Task and Motion Planning with an Extensible Planner-Independent Interface Layer. In *Proc. ICRA*.