Robots, Skills, and Symbols

[Extended Abstract]

George Konidaris
MIT CSAIL
32 Vassar Street
Cambridge MA 02139
gdk@csail.mit.edu

ABSTRACT

This extended abstract summarizes recent work on skill acquisition, which shows that autonomous robot skill acquisition is feasible, and that a robot can thereby improve its own problem-solving capabilities; and on the symbolic representation of plans composed of sequences of skills. It establishes a formal link between skills and symbols, and is aimed at allowing the bottom-up (or *skill-first*) development of robot control hierarchies.

Categories and Subject Descriptors

I.2.9 [**Computing Methodologies**]: Artificial Intelligence Robotics

General Terms

Robotics, Artificial Intelligence, Reinforcement Learning, Planning, Representation, Hierarchy, Skills

1. INTRODUCTION

Intelligent behavior in robots requires symbolic reasoning. For example, a human asking a robot to make a cup of tea might expect the robot to remember that they take their tea with milk and sugar. An intelligent robot would know to look for these items in the kitchen. A particularly intelligent robot, upon discovering an absence of sugar in the house, might reason that it should go and get some at the local convenience store, and that to do so requires money, which it should pick up on its way out of the house.

The core challenge of designing intelligent robots is that this kind of high-level symbolic reasoning is necessary for intelligent behavior, but action and perception must necessarily ultimately take place at a much lower level. We might describe these two levels as *swampy* and *crispy*, and they are depicted in Figure 1.

At the swampy level, we are concerned with performing robust and reliable control, state estimation and perception in a high-dimensional continuous space, in the presence of

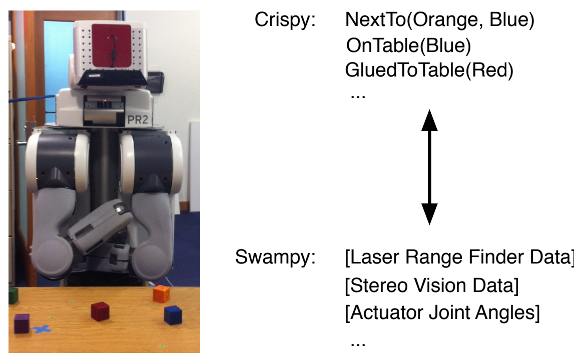


Figure 1: A robot considers a blocks-world problem. At the swampy (low) level it must process high-dimensional visual and laser data, and actuate its joints such that it can pick up and move blocks without colliding with itself or the table. At the crispy (high) level, it can make abstract plans about the placement of blocks on top of each other, without considering the details of movement or perception.

noise and partial observability. Methods that deal with low-level detail and pervasive uncertainty at this level are critical for robust execution in unstructured environments. At the crispy level, we would prefer to deal with very abstract (preferably discrete and near-deterministic) symbolic models of the world. These models allow us to perform long-horizon planning (such as deciding to go to the convenience store for sugar) without having to consider the details of how such plans are ultimately executed (e.g., building an explicit plan listing the low-level motor commands necessary to do so).

A critical question is therefore: what goes between these two levels? How do we design and implement whatever symbolic representations are used in the crispy level, so that they are properly grounded in the swampy level?

The majority of robot systems that have implemented hierarchical control with both levels have followed the example of the Shakey project [12], probably the first robot to comprehensively combine symbolic reasoning and low-level action, in using a strategy that we might term *symbol-first*. Here, the robot's designers create a crispy symbolic description of the world, and then construct the swampy motor and perceptual programs required to execute the resulting plans and ground the resulting symbols, respectively.

One difficulty with a symbol-first approach is that it begs

MLIS'13, August 04 2013, Beijing, China
Copyright 2013 ACM 978-1-4503-2019-1/13/08 ...$15.00.

the question of where the symbols come from. If we were to learn (rather than design) such a hierarchy, it is hard to see how a robot could invent ungrounded symbols and operators, and then acquire the perceptual and motor programs to ground them through experience. Another difficulty is that we have no guidance as to how to design the symbolic description of the world such that it is suitable for our robot system—in many cases, this leads to the robot designers iterating over various symbolic descriptions until they reach one that is satisfactory.

The fundamental cause of these difficulties is that the swampy level is "real" (in that it is grounded in interactions between the robot and its environment) and the crispy level is "imaginary"—in that it is not a property of the world, but instead is something that only exists to help the robot plan; such plans only ground out when executed at the swampy level. Consequently, I propose an alternative method for creating a robot control hierarchy that starts with the swampy level and first creates discrete operators (or *skills*) through interaction with the environment, and then the necessary symbolic representations for reasoning about plans composed of those skills. We can consider such an approach *skill-first*.

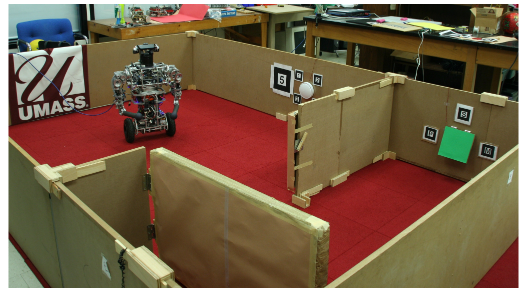
2. SKILL ACQUISITION

Symbolic representations used for planning largely follow the STRIPS model [3], consisting of a set of operators describing actions that the agent may take, and a set of predicate symbols used to create descriptions of the preconditions for executing each action and modeling its effects once executed.

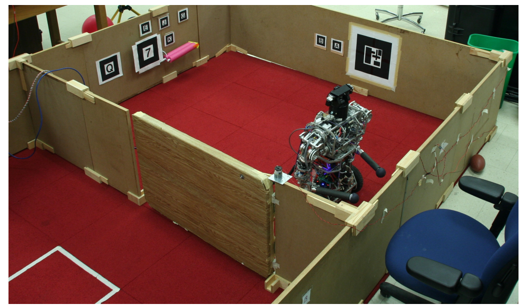
In the skill-first approach to hierarchical robot control, the first step in creating a symbolic abstraction is creating discrete motor control skills that can be retained, refined, and then deployed by the robot when convenient. The necessary technical framework for approaching this problem is provided by recent work on hierarchical reinforcement learning [1], where an agent solving a problem posed as a Markov decision process (or MDP) can isolate subproblems to create, learn and use appropriate macro-actions. These macro-actions can be useful both in helping to solve the original problem, and as a means of transfer for solving subsequent problems more efficiently [5, 10].

My thesis work [4] focused on scaling up so-called *skill discovery* methods—methods for discovering new macro-actions from experience—to high-dimensional, continuous domains. It introduced *skill chaining* [7]. a method for discovering new skills from interaction with an environment, whereby the skills are constructed in a tree, such that executing a sequence of them leads the agent from any state in the problem to a solution state. A complimentary method, *abstraction selection* [6], selects a skill-specific abstraction for each newly created skill, from a library of available abstractions. When combined into a fast skill discovery algorithm called *CST* [9, 11], these two methods enable autonomous robot skill acquisition [10].

Figure 2 shows the uBot-5 humanoid robot [2] in a training and test task. The robot used reinforcement learning to solve the training task, and then extracted useful skills from the resulting solution. Since these skills were constructed using their own abstractions, they were themselves abstract, and could be applied in tasks other than the task in which they were learned. When the robot was given access to these skills while learning to solve a test task, they reduced the time taken to learn to solve it by nearly half [10].



(a)



(b)

Figure 2: The uBot-5 learned to solve a training task (a), and in the process acquired skills which meant it could solve a second test task (b) more quickly than using only its pre-existing controllers.

This work shows that it is indeed feasible for a robot to discover modular components of behavior through interaction with an environment—without any human intervention, and without the use of symbolic descriptions of that environment—and thereby improve its own ability to solve problems.

3. SYMBOL ACQUISITION

Given a set of (acquired or designed) skills, our recent work [8] considers the following question: what symbols are required to be able to reason about plans composed of sequences of those skills?

First, we must operationalize the notion of a symbol. An abstract symbol (like `LightsOn`) refers to a set of real-world states (those in which the light is on); in our terminology, a crispy symbol refers to a set of swampy states. Listing those states gives us an *extensional* representation of the symbol; unfortunately, this is impractical because in almost all cases our symbols will refer to an infinite set of swampy states. Instead, we wish to use a compact representation of those states (i.e., an *intensional* representation of the set). One way to do so is using a classifier: a function that returns true for the states in the set, and false for the states not in the set. If that classifier supports certain operations, then we may form a concrete boolean algebra for performing symbolic operations over classifiers (i.e., symbols).

Our question then becomes: which sets of states must we represent in order to determine whether or not any plan (composed of sequences of skills) is feasible? It turns out that we must be able to represent the set of states in which each skill can be executed (its *precondition* set), and be able to compute each skill's *image*: given a set of states the robot could execute a skill from, we must compute the set of states executing that skill could leave it in. This operation is incomputable in general, but is easy to compute given some common types of skills (e.g., subgoal skills, and abstract subgoal skills). These conditions provably provide the necessary and sufficient symbolic representations for planning using a robot's skills [8].

This result shows two things. First, the appropriate symbolic representation for a robot depends directly on both the environment and the robot's skills. There can be no "correct" symbolic description of the environment that does not take the robot's skills into account; it is easy to produce an example where the robot and environment stay fixed but the robot's skills change, and the appropriate symbolic representation also changes. Second, the symbolic description the robot uses to plan in an environment does not require creative designer effort—it is in fact entirely specified by the combination of skills and environment. Moreover, it has a concrete and precise definition which (once we pick a classifier class) can serve as the target function for a suitable machine learning technique [8]. This allows an agent to learn a symbolic representation of the world that is suitable for planning using its skills.

4. SUMMARY AND CONCLUSIONS

The work cited here has demonstrated, first, that autonomous skill acquisition is feasible on robots; and, second, that symbolic representation can be driven by the skill-acquisition process. A robot can therefore acquire new procedural knowledge (in the form of skills) through direct interaction with its environment, and then derive the necessary abstract symbolic representations to plan using those skills. This formal relationship allows us to design robots (or have them develop autonomously) from the bottom-up—skill-first—by constructing controllers grounded in interaction with the real world, and then synthesizing the symbolic representations suitable for planning using them.

5. ACKNOWLEDGMENTS

I would like to thank my coauthors on the various papers cited here—Scott Kuindersma, Andy Barto, Rod Grupen, Leslie Kaelbling, and Tomas Lozano-Perez—for their insight and their patience.

6. REFERENCES

[1] A. Barto and S. Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13:41–77, 2003.

[2] P. Deegan, B. Thibodeau, and R. Grupen. Designing a self-stabilizing robot for dynamic mobile manipulation. In *Proceedings of the Robotics: Science and Systems Workshop on Manipulation for Human Environments*, August 2006.

[3] R. Fikes and N. Nilsson. STRIPS: a new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[4] G. Konidaris. *Autonomous Robot Skill Acquisition*. PhD thesis, University of Massachusetts Amherst, May 2011.

[5] G. Konidaris and A. Barto. Building portable options: Skill transfer in reinforcement learning. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence*, 2007.

[6] G. Konidaris and A. Barto. Efficient skill learning using abstraction selection. In *Proceedings of the Twenty First International Joint Conference on Artificial Intelligence*, July 2009.

[7] G. Konidaris and A. Barto. Skill discovery in continuous reinforcement learning domains using skill chaining. In *Advances in Neural Information Processing Systems 22*, pages 1015–1023, 2009.

[8] G. Konidaris, L. Kaelbling, and T. Lozano-Perez. Symbol acquisition for task-level planning. In *The AAAI 2013 Workshop on Learning Rich Representations from Low-Level Sensors*, July 2013.

[9] G. Konidaris, S. Kuindersma, A. Barto, and R. Grupen. Constructing skill trees for reinforcement learning agents from demonstration trajectories. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1162–1170, 2010.

[10] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto. Autonomous skill acquisition on a mobile manipulator. In *Proceedings of the Twenty-Fifth Conference on Artificial Intelligence*, pages 1468–1473, 2011.

[11] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto. Robot learning from demonstration by constructing skill trees. *International Journal of Robotics Research*, 31(3):360–375, March 2012.

[12] N. Nilsson. Shakey the robot. Technical report, SRI International, Apr. 1984.