Smart And Autonomous Systems For Repair And Improvisation
Christopher G. Atkeson and Katerina Fragkiadaki, CMU
Proposal to the NSF Smart and Autonomous Systems Program 7/31/18

Summary

This integrative project addresses a longer-term vision and intellectual challenge of developing a library approach to robot behavior generation and learning. The researchers will build a large-scale long-term multi-domain intelligent physical system that can repair and improvise desired processes and devices. The system will learn from instruction in the form of educational kits and books, and from a curriculum of training tasks, watching humans do tasks, coaching, practice, less directed play, and reflection. Evaluation of the proposed work will focus on how well robots can perform suggested activities from the instructional material, repair broken processes and devices, and create processes and devices that achieve new task specifications. The following hypotheses will be tested: 1) appropriate libraries can be built in practice and grown over time, 2) relevant experience can be accessed from a large multi-domain library and combined to exhibit rich behavior comparable to humans, and 3) such a library can support life-long learning for thousands of tasks in multiple domains, rather than a one-shot demo of one task in one domain. Can they overcome the curse of knowledge for AI systems: The more the system knows, the slower and stupider it gets?

Keywords: library-based; hybrid system; hybrid learning; learning from demonstration; case-based learning; memory-based learning

Intellectual Merit: The potential for the proposed work to advance knowledge and understanding is based on the issues they will explore and the ideas they will develop including: a) learning from instruction in the form of educational kits and books, b) learning from narrated and annotated demonstrations, c) enabling a single robot to truly work in multiple domains, d) using a task-level approach to perception, modeling, planning, and learning, e) representing and learning about alternative strategies and policies, f) learning from play, g) space and object aware feature representations, h) using Big Simulation to reflect and learn to learn, and i) combining many robot control and learning approaches in a single integrated system. The integration plan is based on their successful development of their control system for the DARPA Robotics Challenge. The evaluation plan describes methods to assess both formatively and summatively the integrated system. The team is well qualified based on the PI’s track record and preliminary results. The project has adequate resources in the form of several robots and hands to carry out the proposed activities, as demonstrated in the preliminary research.

Broader Impacts: They expect to make programming robots cheaper, and make robots more useful, particularly for domestic and care robots supporting everyday life activities; repair, construction, and decommissioning robots; and explorer and worker robots in the oceans and space. In addition to curriculum development inspired by the proposed research, they have several outreach efforts. A major outreach initiative led by Atkeson is the creation of a physical and virtual Robot Museum. The impact of Atkeson’s work will be increased by a new Disney TV show based on the characters from the Disney movie Big Hero 6, including the inflatable medical robot Baymax inspired by Atkeson’s work on inflatable robots. They will publicize the proposed work in synergy with this TV show by emphasizing they are “Building Baymax’s Brain”. Fragkiadaki organizes the CMU chapter of the AI4ALL national program.
Figure 1: Some domains we will use to inspire and evaluate our work, taken from educational kits and books involving physical processes and devices a robot can work with, learn and reason about, and repair, including mechanical, electrical, thermal, chemical, and combustion processes.

**Smart And Autonomous Systems For Repair And Improvisation**

We will develop a large scale long term intelligent physical system that can repair and improvise desired processes and devices. The system will learn from instruction and a curriculum of training tasks, and from watching humans do tasks, coaching, practice, less directed play, and reflection. Recently the built-in driver seat in a motorboat I was using broke. I improvised a new seat by putting in an appropriately sized box cooler. Improvisation is typically required when unanticipated problems occur, other agents fail or just don’t do what you want, and when resources, time, or tools are limited. Classic examples of improvisation with limited resources are saving the Apollo 13 astronauts, the survival of the stranded astronaut described in the book and movie “The Martian”, and the inventiveness shown in the two versions of the TV show *MacGyver*. Robots need to improvise to solve problems to keep working or exploring, just like humans do.

We will use a variety of educational kits and books about processes and devices in various domains to develop and evaluate our ideas and algorithms on how to learn knowledge about repairing and improvising, including mechanical Lego-based Rube Goldberg machines with rigid, elastic, and plastically deformable components as well as kits to explore cooking, combustion, chemistry, and electricity (Figure 1). We will extend these domains with the robot’s ability to design and 3D-print new components and tools. Figure 2 shows some of the other domains we will work with, including elastic and plastically deformable objects, liquids, and granular materials (for example from food preparation). For all of these domains there is a rich set of instructional material and videos, as well as recipes and suggested experiments and other activities that can be used to inspire exploration, play, and evaluation (Figure 3). Evaluation of the proposed work will focus on how well robots can perform suggested activities from the instructional material, repair broken processes and devices, and create processes and devices that achieve new task specifications.

One practical goal is to reduce the cost of robot programming. Programming a robot to do a desired task is difficult and expensive (typically requiring one graduate-student-year for state of the art dynamic tasks). Adding the necessary error handling is many times more expensive, as it is very difficult to anticipate all the things that will go wrong. We learned from our participation in the DARPA Robotics Challenge (DRC) that designing robust behaviors for robots is very difficult, even for professionals. Small changes in the task caused robots to fail, and even to fall or crash [16]. We propose methods that will help automate robot programming and error prevention and handling. We want to enable robot workers and explorers to make simple plans and solve minor problems autonomously, and be able to attain a safe state and ask for help when major errors
Figure 2: Our Baxter robot pouring, our PR2 robot pouring, cutting a tomato, trying to peel a banana using teleoperation (1 is a human-like strategy of ripping the skin at the stem, and 2 and 3 explore alternative strategies more appropriate for the robot), cutting a peeled banana (easy), and cutting an apple (hard).

or problems occur. Endowing robots with the ability to solve simple problems, learn, and ask for help when needed is more cost-effective than trying to make robot programs free of bugs and conceptual errors.

Intellectual Merit

This integrative project addresses a longer-term vision and intellectual challenge: developing a large-scale long-term multi-domain library approach to robot behavior generation and learning. Our main ideas include: We have focused the proposed research on educational kits and books about physical processes (Figures 1 and 3). This allows us to explore a rich set of tasks, multiple skills, and learning across multiple domains, while providing a reasonable way to evaluate our work. We will go beyond rigid objects to handle deformable and workable objects, liquids and granular material, and electrical, thermal, chemical, and combustion processes. We will go beyond just using robot hands and limbs as effectors to learn to use tools. We will develop approaches to learn to use tools better, learn to make better tools, and learn to invent new tools. We will develop a task-level approach to perception, modeling, planning, and learning: We have found that modeling complex physics in the context of both a task and a particular robot strategy is much more effective than trying to learn general models. Specializing perception, planning, and learning algorithms to the current task and skill makes perception, planning, and learning much easier as well. We will explore ways to represent alternative strategies and policies. Smart systems need to be able to reason about the second and third best way to do a task, as well as just the best way. Often on the actual robot, the 2nd best strategy turns out to be better than the best strategy from simulation. One key idea is using narrated and annotated demonstrations to simplify learning symbolic task theories and strategies from demonstration. Language used in our human-robot interaction is deliberately simplified. Another key idea is to learn from “play”, defining play as intelligent exploration of variations of a physical process or device. We are exploring a space and object aware feature representation in perception in addition to traditional vector representations. This gives us more flexibility in using deep learning. We are combining many robot learning approaches, such as learning from instruction, demonstration, practice, coaching, play, and reflection, as well as combining model-free and model-based reinforcement learning. A major focus is combining symbolic theories and strategies and numeric models and policies. We will explore combining knowledge-based planners (such as geometric motion and grasp planners) and learning systems (such as deep neural networks). We will explore combining task-level, mid-level, and action primitive-level models, policies, and learning. We will explore combining memory-based and parametric learning approaches. We will make extensive use of Big Simulation to “learn to learn”, in which robots reflect on their simulated behavior and learning and tune algorithm parameters to be more effective. We have a high likelihood of success in building systems that plan probabilistically to handle predicted uncertainty, errors, hardware failure, and unexpected dynamically changing situations. For example, we have extensive experience in error and failure detection in
Figure 3: **Top Row:** Projects described in detail in the Lego: Chain Reactions kit [85]. Note that accompanying video is provided with information on assembly and the working devices. **Middle Row:** Details of the first project. Diagrams and text are presented. The device is decomposed into named parts, which are re-used in other projects. Assembly instructions are provided. There are theories presented as to what causes what and why the device works. **Bottom Row:** Theories continued. Diagrams and vague instructions are provided for additional projects.

complex humanoid robots. Our Atlas robot in the DRC was the only robot that tried all tasks, did not fall down, and did not require physical rescue by humans (Figure 13). This high level of performance was due to rapid detection of robot errors, failures, and other unexpected situations, and the autonomous ability to get the robot into a safe state, and call for remote human assistance (Figure 14). We were also among the most consistent teams, performing essentially the same on both test days.

Although it looks like we are proposing too much, many parts of this work have been done already for the DRC, and the main effort is integrating the new parts (See Integration Plan). We are restricting the language we will consider. We are restricting our research to single agent tasks. Although hybrid approaches have promised much and delivered little in the past, we are betting it is worth another try. Overall, we are proposing to integrate our work of the last few decades into a non-trivial system. Integration has not and will not be attained simply by a collection of smaller projects provided with similar resources. The DRC is the first time we attempted major integration. This proposal builds on that work. Such integration seemingly does not happen without specific funding for it. We just keep building toy systems.

This integrative project integrates seven desired components of Intelligent Physical Systems. **Cognizant:** The proposed system is self aware because it remembers training data, explicitly represents uncertainty, and uses simulation and representation of alternative outcomes to predict potential failures and pre-plan reflexes and re-planning. Learning systems that remember data (memory/case-based) can tell when parametric learners are not doing well. **Taskable:** The project will coordinate learning of a simplified language with other aspects of learning from instruction (educational kits and books) and learning from narrated and annotated demonstrations and coaching. **Adaptive:** The system will use many forms of learning. **Ethical:** The proposed system will be able to obey symbolic and numeric societal and legal rules in the form of constraints, and also trade off ethical considerations with other factors using optimization to make decisions. **Transparent:** The proposed
system will be able to explain decisions by referring to instances and statistical analysis of symbolic and numeric training data and prior experience. The system will also be able to provide predictions as to what is going to happen, and justify those predictions with reference to training data and prior experience. **Robust:** The proposed system will explicitly represent and reason about uncertainty, and be able to choose robust task strategies from a set of possible strategies. **Intelligent:** The proposed system will exhibit high-level cognition by perceiving, communicating, and acting at multiple levels of abstraction in complex physical environments in a knowledge-rich manner, including a variety of representation and reasoning mechanisms, such as symbolic, numeric, probabilistic, semantic, and meta-reasoning.

We will be guided by related work on behavior libraries: In computer animation, the use and adaptation of databases of human poses and motion are common [75, 102]. In artificial intelligence, the idea of storing, re-using, and adapting plans has a long history [84, 117]. A number of efforts have been made to use collections of stored trajectories to represent policies, including [69, 51, 52, 17, 108]. Library-based approaches in robot planning and control include [10, 13, 47, 46, 96, 114, 112, 113, 79, 85, 87, 121, 98, 25]

**Intellectual Framework: What Knowledge Is Useful?**

In this section we will use a case study (opening a jar whose lid is stuck (Figure 4)) to explore what types of knowledge are possibly involved in understanding and performing everyday tasks. The lid might be screw-top, or press fit (held by friction or interlocking shapes and deformation is required for removal or replacement). The jar might be brittle glass, deformable plastic, or even metal. There is a known rich library of strategies and theories about how to open a jar [111, 88, 34, 38, 27, 28, 123, 127, 128, 32, 126, 39].

A first step in defining a task is creating or selecting a goal condition or goal set, or an optimization criterion. We will maintain a library of possible goals and optimization criteria to simplify task definition for users and help make the system taskable (Figure 15). Symbolic and numeric goals can be explicitly programmed, taught, or coached by a human supervisor. Goals will also be learned using inverse reinforcement learning. This explicit representation of goals supports generalizing knowledge between tasks with similar goals as well as reflection to improve performance using mental practice. We also expect the goal library to simplify human specification of tasks by modifying existing goals. We will explore how well ethical constraints and optimization criteria can be included in the goal library to shape and limit behavior.

One may want to use symbolic theories of why the lid is stuck to guide reasoning about possible strategies to open the jar. For example, the robot might not be able to unscrew the lid because it is too weak. A grasp might slip because there is not enough friction. The lid might be stuck because it is too small and thus too tight a fit. The lid may be glued by material in the jar. The lid might be held down by a vacuum in the jar so
The friction between the lid and the jar is large. Although some of these theories may be wrong, they do focus knowledge-based search for relevant actions. Theories can be either deterministic or probabilistic. Numeric versions of theories are numeric models. Forward dynamic models encode the effects of actions, and can be used to simulate proposed actions (mental practice). Qualitative theories explain what inputs affect which outputs. A form of local numeric model that is quite general and easy to learn is a derivative or Jacobian of a transformation, numerically relating changes of inputs to changes in outputs.

We expect a robot to have access to a library of symbolic strategies (task topologies). We define strategies as qualitative descriptions of how to do a task. They typically describe relevant objects such as tools, contact sequences, whether contacts slip, and qualitative information about forces that are applied. Strategies can be either deterministic or probabilistic. Strategies can be motivated by and strongly linked to theories or models, or they can be theory/model-free in the sense that they are only known to be potentially useful in certain situations and can be executed. Strategies can be learned in a model-free way by learning which to select, or how to vary them if they have parameters.

For opening the jar, there are many possible strategies. For example, using a tool can provide more leverage, which makes the robot effectively stronger, and the tool can increase the frictional torque available (Figure 4). Heating the lid can make a metal lid relatively larger than a jar made of different material and soften glue-like material, and heating the jar can reduce any vacuum. Putting new material between the robot hand and the lid can increase friction. Prying the lid away from the jar a little bit or tapping the lid can break glue, and also reduce a vacuum. Making a hole in the lid can reduce a vacuum. Slapping the bottom of the jar makes use of a water hammer effect to unseal the vacuum. Smashing or cutting open the jar will provide access to the contents. Asking some other robot or human that is a more capable agent to do the task is a generally useful strategy. Giving up, or getting the desired contents of the jar from some other source is also reasonable. In task-level learning, these strategies can be used in model-based learning where a task-level model of the effect of any inputs and parameters on the outcome of the entire behavior is learned and is used to alter commands to attain a goal or optimize a criterion. Model-free reinforcement learning can also be used to optimize a criterion. We note that what we emphasize in robotics courses such as geometric motion planning and hand grasp planning is only a small part of generating or refining these strategies.

Each of these strategies can be implemented in many ways using more specific strategies. For example, in addition to a wide variety of pliers, vises, and clamps one can find in a hardware store, there are specialized tools to grip jar lids and jar bodies as seen in Figure 4. A heat source such as a hair dryer can be used to heat the lid, as will putting the jar close to a heat source such as a fire or stove burner. One can pour hot water on the lid, or put the jar upside down in a dish of hot water, to focus heat on the lid. Using a lit match to heat the lid is probably an ineffective strategy, but a lighter might work. A blow torch is overkill. A dish towel, rubber bands, a rubber pad, Saran wrap, and mouse pads are materials that are commonly used to increase friction. Thus, strategies form a strategy graph with alternatives and a hierarchy of more specific information (Figure 5). Closely related to the idea of strategies are symbolic affordances associated with objects. Affordances indicate possible strategies or uses for an object or part of an object. Strategies may also have information about expected perceptual results.
For example, the *slapping* strategy should result in a “pop” sound if the seal holding the vacuum is successfully broken.

Ultimately, numeric control laws or **numeric policies** are used to instantiate and execute a symbolic strategy graph. It is often useful to use a **policy graph** that matches the symbolic strategy graph. A policy graph can be used in simulation or in reality to generate an instance of a **behavior**, which records histories of actions, measurements (sensations), internal variables, and which path through the policy graph and possibly a corresponding strategy was taken.

Each policy consists of a combination of other previously learned policies and primitives that the robot can execute directly. This is a similar intellectual stance as found in options, macros, chunks, schemas, basic behaviors, subroutines, operators, etc [9, 18, 101, 109]. Policies form a hierarchy. We expect primitives to be manually defined, and might include holding a position or force, following a desired trajectory of joint values, effector positions and orientations, effector forces, joint or effector impedances, or a combination of these values, and executing a policy such as move until contact.

What we learn from this case study is that there are many kinds and levels of knowledge that could be involved in understanding and performing everyday tasks, including theories, models, strategies, and policies. We expect this knowledge base to form a heterogeneous graph with relations such as IS-A, IS-A-STRATEGY-FOR, IS-COMPOSED-OF, IMPLEMENTS, IS-IMPLEMENTED-BY, and MODELS. Levels range from complete tasks (task-level) to mid-level to primitives. Some of these components and tasks can be accurately simulated, and some can only be simulated approximately.

Some of the questions we would like to answer include: What role can knowledge in the form of theories and models play in repair and improvisation? Conversely, how little can an agent know in terms of theories and models of how processes work and what causes what and still successfully refine policies selected from a rich library? How can relevant multi-domain theories, models, strategies, and policies be used to perform a particular task? We will build on work from symbolic case-based reasoning [99] as well as our work on numeric memory-based reasoning [12]. How can a set of relevant multi-domain theories, models, strategies, and policies be used to choose a particular strategy for a given task? We will build on work from case-based reasoning as well as our work on memory-based reasoning here as well. How can symbolic theories and strategies, including tool creation and use, be acquired (either by human programming, demonstration, or language, or analysis of experience by the agent)? How can we handle learning multiple tasks using multiple strategies and policies in multiple domains? In the past we (and much of the robot learning field) have focused on learning one task at a time, and typically what are possible relevant features and actions have been limited to a small set so that current feature and action selection algorithms can work. We want to explore much richer situations. How can we use simulation and experience to learn to learn (become better at future learning tasks)? How can instances of behavior be segmented to achieve more effective strategy and policy re-use? How can data from segmented behavior be applied to refine the appropriate stored knowledge and not interfere with learning from other experiences?

**We will focus our proposed research on learning and using symbolic theories and strategies.** We will initially explore the use of manually created knowledge-bases of theories and strategies. In later work we will focus on learning additional theories and strategies from instruction, currently available human demonstrations (such as Youtube videos), and specially created narrated and annotated “teaching” demonstrations. This work necessarily requires us to address how to compose theories and strategies as well as relate different levels of theories and strategies in a hierarchy. We will also focus on connecting symbolic theories and strategies to simplified language, so that human users can easily provide and refine theories and strategies using spoken and written language. A major challenge is pulling this all together into working and robust reliable systems. Another challenge is enabling our systems to be transparent by being able to explain what they are doing, and apply linguistic and numerical ethical (societal and legal) constraints and rules while making decisions. Evaluation will involve testing our systems on material from educational kits and books that were kept out of the training data. Can our systems follow instructions as presented in these kits and books, construct specified processes and devices, fix broken processes and devices, and create new processes, devices, and tools to achieve
Figure 6: **Left:** The robot swinging up an inverted pendulum by moving the hand side to side. **Right:** The pendulum and hand motion during robot learning from demonstration and practice using a non-parametric model.

new task specifications?

**Prior Research**

Major emphases of this proposal are learning across multiple domains, and representing and learning about alternate strategies for doing a task. Robots need to go beyond rigid bodies and an overemphasis on geometry. We have developed expertise in deformable and workable object manipulation (bending, squeezing, stretching, pressing, stamping, shaping, drilling, cutting, machining, inserting, breaking, ...), as well as manipulation of liquids, slurries, powders and other granular and flake materials, and aggregates of many materials (Figure 2).

We will deliberately test our ideas using tasks and materials that are hard to model. Our work has focused on manipulating liquids and particulate materials (for example, pouring), and cutting objects with skins [132, 134, 144]. These tasks also allowed us to explore learning different task strategies. In the case of pouring, strategies include tipping, shaking, and tapping. In the case of cutting, one can slice, stab, or saw. We believe humans learn these type of skills from demonstrations by other humans, and we all maintain skill libraries representing alternative strategies to do the same task.

Another major emphasis of this proposal is combining methods to get the strengths of each. Figure 6 presents an example of combining model-free and model-based reinforcement learning, learning from demonstration, and learning from practice. The swing up task involves swinging an inverted pendulum up to vertical by moving the base (hand) from side to side [14, 15, 11]. This task is nonlinear, unstable, and non-minimum phase. The robot watches a human do the task once. The figure shows the human’s demonstration and the robot’s practice trials. On the first attempt the robot imitates its perception of the teacher’s movement, which fails to swing the pendulum upright. The robot then uses an updated learned model of the task dynamics to plan (using trajectory optimization) a new hand motion. The 2nd trial is better, the model is updated again, and the 3rd trial succeeds. An important goal of our work is this type of rapid learning from demonstration and practice.

In this work the robot used learned inverse models of the task (we tried both parametric and non-parametric models) to map task errors to command corrections [6, 1, 12, 80, 82, 81, 105, 103, 106, 104]. We have found that optimization is more effective than trying to track a learned reference movement, especially with non-minimum phase plants. We have found that optimization greatly speeds up this type of learning. However, this type of learning sometimes gets stuck because updating the model with new data causes only a very slow change in the policy because the planned movement is in a different area of state space from the new data. Sometimes we learn little from our mistakes. We use direct policy optimization to solve this problem. We have also implemented direct policy learning and policy selection to allow a robot to learn air hockey and a marble maze task from watching a human [24, 22, 23]. Other prior work on policy learning and optimization includes [7, 112, 115, 116]. This example convinced us that combining different approaches to intelligence is ultimately the right way to go. Recently there has been an explosion of interest in combining trajectory and policy optimization in model-based reinforcement learning [41, 59, 92, 53, 42, 44, 54, 55, 71, 73, 74, 76, 93, 120, 147, 56].

Additional relevant previous work includes: We developed a proposed framework for learning modular dynamic models and skills in [135, 136, 143]. We developed a stochastic extension of neural networks in [140] that is useful for modeling behaviors. We explored deep reinforcement learning applied to our graph-based...
approach in [139]. We developed a version of Differential Dynamic Programming (DDP) for graph-structured sequences of policies in [138]. We developed a stereo vision method to estimate liquid and particle flow during pouring [141]. We developed FingerVision, a vision system that both measures the deformation of skin on the robot fingers to measure contact locations and forces, and sees through the transparent skin to provide object and surface localization and tracking during manipulation (in this case cutting with a knife) [137]. We have shown generalization and adaptation abilities, and efficient learning based on both simulated and actual practice. Our and other’s work provides a solid foundation for the proposed work.

Proposed Research

This is an integrative proposal, which integrates many pieces of our and other’s work, our own software, and publicly available software. Page limits force us to only present selected highlights of our research, integration, and evaluation plans. We build on our extensive experience with robots learning numeric models and policies, and knowledge-based planning systems exemplified by our work for the DRC (Figures 13 and 14) [40, 16].

Initially manually build a prototype system for rigid body mechanical processes: To provide an initial system to work with, we plan to spend Year 1 manually building a prototype system with example theories, models, strategies, and policies. This example system will focus on rigid-body mechanics, so it can work with kits like the Lego Chain Reaction kit (Figures 1 and 3) [85], as well as kits involving marbles rolling down ramps, and patterns of dominos falling.

Limit the language in HRI: Robot Esperanto: In addition to manually building an initial system, we will explore expanding the system using learning from demonstration with visual learning and learning from language-based instruction from the kits and books as well as narration from teachers. We will develop a simple English-like robot interaction language, Robot Esperanto, with simple sentence structures like [adjective] [noun] [verb] [object] [adverb]. Utterances will be similar to what is found in the “Dick and Jane” readers for early grades. We may have to translate the kit instructions and books to Robot Esperanto. The goal of the proposed research is to use as simple a language as possible to explore the role of language in human-robot interaction in the proposed domains, rather than focus on sophisticated language processing.

Combine language learning with learning about physical processes: A cluster of ideas we are exploring focuses on how to communicate with users and make a system taskable using language, by linking simple language to other forms of knowledge. This will make our systems able to learn from a demonstrator, teacher, or coach. One idea is to take advantage of the pairing of language (text), diagrams, and accompanying video in educational kits and books (Figure 3). Another idea is to coordinate language learning with learning about physical processes and policies by using narrated (in Robot Esperanto) and annotated human demonstrations (Figure 7). We expect this coordinated language learning will help our systems interpret high-level and possibly vague instructions. Teachers will be accessible through a web interface so we can use crowd sourcing, or will be present in the robot’s workspace. Teachers will also “coach” the robots as the robots attempt to imitate the teachers. Many previous works use silent demonstrations [8, 37]. Automated speech recognition supports the collection of a very large amount of paired natural language, action, and visual streams in a short amount of time. We will also provide convenient ways for the teacher to annotate which objects are being referred to by using techniques like a visual cursor or laser designation, or additional “after-action” narration and annotation.
Preliminary Results: In our recent work \cite{125}, depicted in Figure \ref{fig:learnable_policies}, we use narrated visual demonstrations of pick-and-place activities from egocentric videos to learn pick-and-place policies instructible by natural language. Given natural language instructions paired with visual demonstrations we learn a visual reward detector for each such instruction and use them to guide policy learning in the simulator, replacing manually coded rewards. We focus on visual detection of natural language spatial expressions, e.g., "the bottle should be inside the wooden box". The spatial expression is parsed using a BiLSTM (bidirectional long short-term memory neural network) to localize subject, objects and relationship sub-phrases, and produce corresponding word vectors. The visual detector, given a natural language utterance, its semantic parsing and an RGB image, returns a detection score $S$ based on how well the image matches the natural language utterance, using the modular architecture depicted in Figure \ref{fig:learnable_policies}b. The detector is trained with weakly supervised metric learning, as shown in Figure \ref{fig:learnable_policies}c. Given a video sequence, our spatial expression detector effectively detects changes of spatial configurations for the depicted objects (Figure \ref{fig:learnable_policies}d). We use the learned visual detector to supply reward feedback to a learning agent that is learning a manipulation policy using deep Q learning to achieve the corresponding spatial configuration (Figure \ref{fig:learnable_policies}e). Such perceptual reward detectors can be used on real robotic platforms when instrumentation for automated reward detection is absent, and can generalize across environment variations.

**Learning about physical processes from vision of the demonstrations:** Numerous computational representations have been proposed to capture dynamics of agents and processes, ranging from assumptions of fully observed state representations comprised of object 3D shapes and poses provided by appropriate instrumentation \cite{95}, to completely unobserved ones, where standard CNNs (convolutional neural nets) encode 2D RGB frames into state vectors \cite{110, 91, 3}. Despite the simplicity of the latter, they have been found it hard to generalize to novel domains or minor environment variations \cite{61}. The question thus remains: what is the right visual representation that allows learnt dynamic models to generalize, yet can operate in more general environments than a fully instrumented motion capture studio or laboratory? We propose space and object aware visual feature representation learning, that uses structural biases regarding i) the world being 3 dimensional and ii) the world containing objects that move around. The proposed representations learn to encode RGB frame sequences, into egomotion-stabilized 3D feature maps of the depicted scene. They capture view variability due to camera motion, by unprojecting the 2D features or projecting the 3D feature map from the corresponding viewpoint. Further, they attempt to explain temporal visual changes e.g., those caused by rotation and transla-
Active geometric-semantic visual recognition (a) Our active agent sequentially accumulates visual information in a 3D geometrically-consistent feature map of the visual scene. At each frame, the agent selects a nearby camera view, conditioned on the 3D feature map of the scene thus far, and the current RGB image. It predicts a corresponding 2D depth map and foreground object mask. RGB, depth and segmentation maps are unprojected in corresponding 3D feature tensors, mapped using the relative egomotion to the coordinate frame of the cumulative 3D feature map built thus far, and update a 3D convolutional gated recurrent (GRU) memory. The output of the 3D GRU memory is then mapped via a 3D encoder-decoder network to segmentation embeddings, object category probabilities, and occupancy for every voxel in the 3D grid.

Preliminary results: We have shown that Space-Aware Visual Memory (SAVM) architectures outperform 2D LSTM networks that do not encode the geometry of 3D space, but rather operate directly on 2D RGB projections (see Figure 9). SAVM integrates visual features, stabilizing against the ego-motion of the moving robotic observer. Objects are then detected from this integrated feature map, as opposed to any single 2D view. We will train such structured state feature representations using object detection, segmentation and 3D pose prediction ground-truth in both 3D ground-truth (synthetic data) or 2D ground-truth (real images). We plan to integrate such visual feature extractor and temporal integrators with a moving robot for on-the-fly detection, while at the same time controlling the camera to choose the next best view to help visual perception (Figure 9). If successful, this research will provide drastically different paradigm in visual perception of moving robotic agents. We also show how object-centric feature representations are more effective for learning policies from visual demonstrations, as opposed to generic frame-centric ones (Figures 10). The feature representation is comprised of the local appearance features of each relevant object in the scene as well as the relative object distances in 3D. This feature representation is then used to compute the similarity of the robot’s behavior to that of a human demonstrator. This similarity serves as a reward function for the robot to learn policies that imitate the human demonstration using policy search reinforcement learning. We rely on Mask R-CNN [58], a state-of-the-art object detector, to enable the robot to detect relevant objects in the scene and to fit corresponding 2D bounding boxes. The detector network is trained using synthetic data generated from unoccluded examples of the relevant objects. During the imitation process, the learner then extracts object appearance features from each of the object boxes as well as the relative distances of the 3D centroids of the objects in the scenes. The object
appearance features are learned using self-supervised time-contrastive metric learning \cite{110} in order to capture the changes in the objects’ appearance over time. Given the learned features for the similarity reward function, the robot learns policies for performing the task in new situations using PILQR \cite{31}, a trajectory optimization method. Object-centric biases were found to be crucial for generalization beyond the training conditions.

In this proposal, we will build upon our recent work and develop 3D object-centric representations of object dynamics, and end-to-end differentiable tracking frameworks for state-of-the-art perception during manipulation. We will then use learned dynamics models for control and imitation learning of manipulation tasks, directly from visual inputs, and compare against their frame-centric 2D counterparts. We believe incorporating structure and attention on such dynamics learning would enable strong generalization across objects and environment conditions.

**Learning from play and self-guided exploration:** What would guide our agent to explore useful regions of the state space? Thus far, random exploration has only resulted in dynamics for explaining simple behaviors, such as pushing objects \cite{3}, bouncing balls \cite{45,20,30} or predicting cube stability \cite{131,72}; discovering interesting behaviors or physics rules without any guidance has not been fruitful. Our plan is to use a small set of human supplied VR demonstrations, and to augment them in a systematic manner in order to learn to generalize the demonstrated behavior across object locations, shapes, sizes. We propose systematic experimentation by careful sequencing of experiments as opposed to randomization; we believe this is a key to successfully generalizing the demonstrated policy under varying conditions, but also inferring rules of physics that the resulting configuration cannot be achieved or is systematically altered, e.g., the orange falls inside the mug as opposed to staying on top if the mug is enlarged beyond a certain point. Discovering this state change, is only possible by carefully altering the experimental conditions, else, a robotic agent could simply infer its inability to complete the task (here, put the orange on top of the mug) as opposed to the un-affordability of the goal (Figure\cite{11}).

**Finding good behavior sequences:** We find good skill sequences by building a tree of possible sequences and their effects. Initially we use coarse large time step models, and then iteratively prune the tree and search
Initial 3d object trajectory

\begin{tabular}{|c|c|c|}
\hline
 & random & structured \\
\hline
Initial & 6 & 6 \\
A & 11 & 11 \\
B & 8 & 6 \\
C & 44 & 6 \\
\hline
\end{tabular}

number of samples needed by CMA-ES

Initial 3d object trajectory

\begin{tabular}{|c|c|c|}
\hline
 & random & structured \\
\hline
Initial & 40 & 40 \\
h\times1.2 & 4 & 4 \\
h\times1.3 & 4 & 4 \\
\hline
\end{tabular}

number of samples needed by CMA-ES

Figure 11: Structured sampling of counterfactuals for efficient generalization of policies and predictive models of intuitive physics. Left: Varying object locations in the demonstrated behaviour “put the orange to the right of the cup”. Right: Varying the height of the cup. We visualize the number of samples needed by CMA-ES to achieve the desired configuration using random and systematic (structured) counterfactual experimentation.

with more accurate and smaller time step models. We also consider more possible skill choices on each iteration of the search. In preliminary results, our skill-based lookahead exploration outperforms $\epsilon-$greedy exploration, model-based RL [118] where the fitted dynamics model is used to supply (fake) experience tuples and not for exploration, as well as learning policies over coarse parameterized skills directly [57], as opposed to low-level action primitives (Figure 12).

Integrating Methods: In the prior work section we described an example of combining learning from demonstration and learning from simulated (mental) and actual practice, as well as combining model-based and model-free reinforcement learning. We propose several key ideas towards building self-aware knowledge-rich systems that employ a variety of representation and reasoning mechanisms and understand their capabilities and limitations such as when they are outside their expertise. To handle rich representations of knowledge in everyday tasks such as opening a jar, we propose combining alternative approaches along several intellectual dimensions. For example, error-driven/model-free approaches can rescue predictive/model-based approaches when modeling bias causes model-based approaches to fail or get stuck repeating the same error. This is related to how integral control reduces remaining steady state error in control systems. Modeling, planning, and learning at multiple temporal and functional scales helps avoid the pitfalls of focusing on only one scale or type of model. Combining different types of control and learning allows us to obtain the best features of each. We hope to both reconcile and take advantage of the complementary qualities of multiple robot planning and learning algorithms. The PIs have extensive experience working with each of these methods in isolation. It is time for us and the field in general to integrate components into more complete systems that make use of what we have learned so far about learning and intelligence.

Combining approaches to generating behavior: We will combine knowledge-based planning and learning systems with learned function approximators (both memory-based and parametric as described above). This will allow our systems to be taskable in the sense that they can plan out and execute actions that depend on context, go beyond pre-programmed and limited behavior, adapt their behavior to compensate for limitations and/or changing conditions, and handle novel situations. Knowledge based planners and learners use theories and models to guide plan generation and refinement, as described in the jar opening case study. Our prior work using knowledge-based planning systems is exemplified by our work for the DRC (Figures 13 and 14). Our software combined state of the art motion planning algorithms with human supervision.

Combining representations for learning: We will combine several approaches to learning function approximators. One key idea is to combine memory-based and parametric learning representations. Atkeson has a long history of working with memory-based robot learning, in which experiences are stored in memory, and
Figure 12: **Look-ahead search.** We use learned skill dynamics encoded in deep neural regressors for lookahead tree search, in order to aid effective exploration in reinforcement learning of complex manipulation tasks.

most statistical combining of experiences happens at query time using a local model that is discarded after the query is answered. Fragkiadaki has extensive experience with parametric learning in the form of deep neural networks. Memory-based learning minimizes interference as training data distributions change, can easily know how similar relevant experiences are, and can handle discontinuities by using two stage predictions where the first prediction is used as the second query to only find relevant examples on one side of a discontinuity. Deep neural networks have been shown to have excellent performance on many tasks. We propose approximating learned functions using both representations, to capture the benefits and minimize the drawbacks of both. On each query we can compare the response of a neural network with several types of local models fitting the relevant training data retrieved from the instance memory and assess if the predicted values do not agree and are likely not accurate.

**Combining approaches to learning:** We will also explore combining model-free and model-based reinforcement learning. Model-free RL attempts to learn only using actual experience \[64, 66, 122, 145\], while model-based RL uses models and their predictions in simulator-based learning (mental practice) as well as actual experience \[107, 83, 89\]. Model-based reinforcement learning is fast, data efficient, and can simultaneously learn or refine a model while improving performance. Model-free reinforcement learning can handle situations where it is difficult or expensive to learn an accurate model or model-based approaches get stuck. While model-free methods can be applied in simulation (making them model-based), when there is no model model-free methods must learn from typically a large amount of actual practice.

**Using models to verify actions:** Another key idea is to verify actions output by policies and planners using
learned models. If the predicted outcome does not match the requested outcome, more reasoning should be done and perhaps a more cautious approach should be taken by the agent. This verification combined with representations of possible outcomes will help support self-awareness, anticipate potential failures, make contingency plans, and perform online re-planning.

**Big Simulation, reflection, and learning to learn:** In perception, the availability of Big Data has supported rapid research progress and the generation of more symbolic representations from raw images and sound. We propose an analogous approach for the action side of robotics: using Big Simulation to generate large behavior libraries using our proposed representational approach. Simulation with learned probabilistic models is used to predict good sequences of tasks, features, and directions in task space to explore to enable an actual robot to efficiently learn in the real world (Learning to Learn). Clustering and feature invention and detection can be used to identify useful symbols, classes, features, parameterizations, model structures, policy structures, reasonable parameter values, and concepts from massive amounts of simulated robot data. We believe human motor learning is effective because humans identify the most important directions in command space and learn in that subspace that first. Only later are more dimensions added to learning in a principal components-like process to deal with the actual dimensionality of the task.

We do not expect simulations to predict actual robot behavior or outcomes well, but we expect the qualitative features of the simulated data to carry over to real behavior. If the simulated robot has a certain type of failure such as slipping, it is useful to try to explicitly estimate where or under what conditions that type of failure might occur with a real robot, or recognize the type of error when it occurs. We have noticed that low level commands, controller gains, and fixed nonlinear policies do not transfer well from simulation to robot, or even from robot to robot. In the DARPA Learning Locomotion program and the Robotics Challenge we worked with a set of identical robots (in the first case Little Dog and the second case Atlas I). We found that successful performance on one robot did not transfer well to a test site with a supposedly identical robot. It is even difficult to transition learned knowledge from one robot to an identical robot. What we have learned is that what transfers well from simulation or robot to another robot are higher level constructs such as task structure, task strategies, and task-specific learning algorithms, and to a lesser extent, filter, controller, and learning algorithm modes and time constants/bandwidths, and thus we should focus on “high-level” transfer. In the example that follows, we found that optimization simplified behavior generation, leading to a simple description of a complex behavior.

**Preliminary research:** In this example, we show how reflection about multiple plans can lead to discovery of underlying structure, which simplifies future planning. After generating many short optimized walking plans (Figure 14-far-right), our robot discovered that features of these plans could be approximated accurately using simple global parametric models. New optimal walking plans could now be quickly generated from quadratic “curve fits”. Features that were predicted well include the COM position and velocity of the final state of the trajectory, joint values at the start or end of each footstep, times of the start and end of each footstep, and the total cost of the trajectory. We also found we could use the interpolated optimization results as part of the cost in an A* algorithm that chose footsteps in a complex environment. We found our quadratic cost functions could rapidly estimate the cost-from-start without having to run an expensive trajectory optimization, allowing dynamic considerations to be included in footstep planning. The full details of this work are published in [63]. Related work in this area includes the approximate simple models used in many video games to reduce online computation. [62] reports on and reviews pre-computation of object dynamics, in this case cloth.
Research Plan

This research will involve two co-PIs, both with extensive experience in robot learning, including learning from demonstration. Atkeson is an expert on robot control. Fragkiadaki is an expert on robot perception and linking language to perception. The funding will also support two graduate students. We expect the students to allocate aspects of the project based on their interests. A reasonable split is that one student will specialize in learning from demonstration and play, and human-robot interaction. The other will specialize in learning from practice, and robot control.

**Year 1:** To provide an initial system to work with, we plan to spend Year 1 manually building a prototype system with example theories, models, strategies, and policies. This example system will focus on rigid-body mechanics, so it can work with kits like the Lego Chain Reaction kit (Figures 1 and 3), as well as many kits involving marbles rolling down ramps, patterns of dominos falling, and other rigid-body processes. We will focus on integration of existing and new algorithms into our system. A major emphasis will be making basic representational decisions and developing and evaluating prototype implementations of various tasks. We will implement baseline versions of the proposed approach, which will be evaluated in simulation and on our robots. We will explore alternative representations and learning algorithms. We will explore alternative control structures and termination conditions for the execution of primitives. In our preliminary work we made a set of somewhat arbitrary design choices to get something working. In the proposed work we propose a much more thorough exploration, and a more careful statistical characterization of performance. We will also begin exploring how multi-modal narrated and annotated behavior capture of humans can be used in learning from demonstration and also to define more useful component behaviors. We will develop efficient algorithms to learn and optimize temporally decomposed dynamics, including bifurcations and loops. We will develop symbolic-level reasoning to handle changes to processes, such as movement of objects, and failure of previous strategies. At the end of the first year our milestones will include both successful simulations and an implementation of our prototype system on robots for rigid body mechanical processes.

**Year 2:** In this year we will extend our system to other domains including deformable mechanical, electrical, thermal, chemical, and combustion processes. Major emphases will be on multi-domain integration, transfer learning across domains and robots, and scaling up our knowledge bases. We will evaluate the Year 1 system, and improve the components using several forms of learning and reflection. Milestones will include the ability of the system to do 100 tasks with representative tasks in all domains. We will also demonstrate learning from instruction, demonstration, coaching, practice, and reflection across a set of tasks and robots.

**Year 3:** Although we will formatively evaluate and refine our system throughout the duration of the project, Year 3 will focus on summative evaluation. We will also refine our algorithms in response to evaluation results, improving the integration of components. We will evaluate our approaches both from an experimental and a theoretical point of view. Summative evaluation is described in the section on the Evaluation Plan.
Integration Plan

Figure 15 sketches an initial architecture for our prototype system. This architecture is similar to what we did for the DARPA Robotics Challenge (DRC). The Learning module affects all other modules, so its outward arrows are not shown. Modules to the left are real time, and modules to the right are increasingly slower. The Experience Library stores a continuous stream of the robot’s experiences, as well as simulated, imagined, or hypothesized experiences, and experiences from other robots. It segments this stream in multiple ways to provide chunks of experience appropriate for different users. Items in libraries have permanent IDs in a registry, to reduce bookkeeping across modules. This supports garbage collection (forgetting) at various levels of resolution and moving data to other forms of storage without creating IDs that are not valid.

Fragkiadaki will lead development of the Perception And State Estimation module (upper left of diagram). The perception library (not shown) stores possible perception routines, which are used to define a set of features and objects in their respective libraries. The feature library (not shown) can be extended by transforming or combining existing features, as well as adding new code to the perception library. Objects are recognized by classifiers applied to features, and represented in the object library (not shown).

We will re-use the Policy Execution module (lower left of diagram) and the Policy Library of our DRC software in this system. We generalize the concept of action to include policies at several levels of detail, temporal scale (milliseconds to minutes or hours), and functional scale (instantaneous actuator commands to high level synergies). These generalized actions can be programmed, learned from observation, learned from practice, or learned from simulation (mental practice) if adequate models are available.

The Model Library and Planning module will be based on the same modules in our DRC software. The model library represents numerical models at several levels of detail, time scale, and functional scale. These models can be programmed, or learned using traditional system identification techniques (typically parametric) or more recent machine learning approaches (parametric and non-parametric). Atkeson has extensive experience in system identification as well as non-parametric memory-based locally weighted learning and parametric neural network techniques [12, 80, 82, 81, 105, 103, 106, 104]. The performance measure for this library is how well it supports model re-use, and speeds up system identification and model learning. The goal library is based on a similar module in our DRC software to make it easy to specify tasks.

The major development effort to build this system will go into the Theory Library and the Strategy Library, as described in the rest of the proposal.
Evaluation Plan

The physical testbeds we will use for evaluation will include at least two robot arms with hands. In terms of arms, we currently have available to us several Rethink Robotics Baxters and Sawyers, and expect to purchase more arms using other funds. In terms of hands, we have Rethink Robotics hands as well as a Robotiq hand. In a separate project we are building new hands which may be useful to this project (see Results from Prior NSF Support).

The physical testbeds will be capable of setting up the processes described by educational kits and books such as the Lego Chain Reaction kit (Figures 1 and 3), various marble runs, domino sets (rallies), various cooking kits and recipes, and science experiments using everyday objects. It is important to note that setting up these processes will not require fast movement from the robot. The robot can move as slowly and carefully as necessary to successfully set up and initiate the physical process. After that, the robot mostly observes what happens and then hunts down stray marbles and other pieces (yes, a robot can lose its marbles).

We will keep a random sample of projects from the educational kits and books out of the training set as an evaluation set, as well as suggested projects from the kits and books (Figure 3), and use them for summative evaluation. In terms of overall system assessment, we will measure how well our system can perform novel (test-set) projects. In addition, experimenters and naive subjects will “break” projects and see if our system can fix them. We will also use suggested projects from the kits and books as tasks to perform without detailed instructions, to achieve new task specifications. Metrics will focus on whether the constructed project achieves the desired task, as rated by independent observers (such as Mechanical Turk workers). Statistical analyses will be straightforward (N out of M test tasks were successfully completed giving a success rate of Z%). Cooking projects pose a special challenge, as taste will be involved. In this case local volunteers will provide testing results.

Part of the proposed research will be to develop more continuous metrics that measure partial performance. We will measure the role and contribution of various system components by disabling them or making them perform poorly so that we can perform controlled experiments to assess the role of various components in our system. In addition to evaluating performance, hypotheses we would like to evaluate include: H1: Symbolic knowledge in the form of theories and strategies accelerates learning and improves performance. H2: Symbolic strategy libraries, in the absence of symbolic theories, can enable robots to exhibit rich behavior, repair processes and devices, and improvise new processes and devices. H3: Such strategy libraries can be built and are practical. H4: Symbolic theories accelerate and improve performance and learning. H5: Cross domain robot learning and transfer is possible, and is accelerated by domain-independent symbolic theories and strategies. H6: Narration and/or annotation of demonstrations accelerates learning and improves performance. H7: A simplified language is adequate. H8: Learning from play accelerates learning and improves performance. H9: Space and object aware feature representations improve perception and perceptual learning. H10: Combining learning approaches helps, in the sense that one gets the union of the benefits of each approach, accelerating learning and improving performance. H11: Learning to learn accelerates learning and improves performance. H12: Planning and especially optimization often simplify behavior. H13: Models and regularities learned from reflection can help planning. H14: Learned models can be used in planning. This is difficult because optimization exploits and thus amplifies favorable modeling errors. H15: Our system supports more autonomy than a comparable system that does not use our approach. H16: Our approach reduces the cost of robot programming, rather than just changing the form of robot programming.

Results from Prior NSF Support

The most relevant recent award for Atkeson is: (a) NSF award: IIS-1717066 (PI: Atkeson); amount: $440,000; period: 8/1/17 - 7/31/20. (b) Title: RI: Small: Optical Skin For Robots: Tactile Sensing and Whole Body Vision (c) Summary of Results: This recent grant is supporting work on developing optical approaches for tactile sensing as well as whole body vision (eyeballs all over the body). We will develop robot hands that
complement the robot skin.

**Intellectual Merit:** This project will enable robots to feel what they touch. The key idea is to put cameras inside the body of the robot, looking outward at the robot skin as it deforms, and also through the robot skin to see nearby objects as they are contacted or avoided. This approach addresses several challenges: 1) achieving close to human resolution (a million biological sensors) using millions of pixels, 2) reducing occlusion during grasping and manipulation, and detecting obstacles before impact, and 3) protecting expensive electronics and wiring while allowing replacement of worn out or damaged inexpensive skin. Technical goals for the project include first building and then installing on a robot a network of about 100 off-the-shelf small cameras (less than 1 cubic centimeter) that is capable of collecting information, deciding what video streams to pay attention to, and processing the video streams in real time to estimate forces, slip, and object shape. A transformative idea is to aggressively distribute high resolution imaging over the entire robot body. This reduces occlusion, a major issue in perception for manipulation. Given the low cost of imaging sensors, there is no longer a need to restrict optical sensing to infrared range finders (single pixel depth cameras), line cameras, or low resolution area cameras. Building a camera network of hundreds of cameras on a mobile skin, and building a multi-modal sensing skin, will be highly synergistic with developing the proposed system.

**Broader Impacts:** Robots with better sensing can more safely help people. **Development of Human Resources:** The project involves one graduate student. We have weekly individual meetings and weekly lab meetings. The graduate student is performing research, making presentations to our group, and will give conference presentations and lectures in courses. We will put the graduate student in a position to be a success in academia and industry.

(d) **Publications resulting from this NSF award:** [142]. (e) **Other research products:** We have made instructions on how to build our tactile sensors available on the web. (f) **Renewed support.** This proposal is not for renewed support.

There is no prior NSF support for Katerina Fragkiadaki.

**Broader Impacts of the Proposed Work**

We expect to make programming robots much cheaper. Humans directly authoring behavior and humans teleoperating behavior are the wrong robot programming models. Endowing robots with the ability to solve simple problems, learn, and ask for help when needed is more effective. Another practical goal is to make robots more useful, particularly for domestic and care robots supporting everyday life, and worker and explorer robots. We will push the limits of what robots can do autonomously in multiple domains. In addition to curriculum development inspired by the proposed research, we have several outreach efforts.

This work, focusing on “everyday” physical processes, will eventually also be applicable to robots on the surface and oceans of Earth, in orbit, on the Moon and other bodies in space. Limited resources that could encourage robot autonomy and improvisation include long communication latencies and low bandwidth communication, and the expense of human supervision. Underwater repair is a domain which often requires a great deal of improvisation as initial diagnoses are often wrong, materials and tools break, weather interferes, the ability to access or apply forces to do a repair may be limited, and it is difficult to get new resources to a site quickly. Nuclear plant decommissioning also requires improvisation as the actual structure may be damaged or just different from plans, materials may have degraded, and doors and other fixtures may no longer work as desired. On the Moon it would be useful to use lunar materials for large scale construction of solar panels, and variation in lunar materials and conditions may require improvisation as well.

**Impact on the research community:** We expect to continue to make demonstration and robot data available on the web. We have had great success making public most data collected in our Motion Capture Lab (mocap.cs.cmu.edu and kitchen.cs.cmu.edu) and Panoptic Studio (domedb.perception.cs.cmu.edu). Data made available so far has been acknowledged in several hundred papers, mostly from the computer graphics, animation, and vision communities worldwide. This form of usage is freely available to all, including those from non-Ph.D. and/or minority-serving institutions. We will host visitors who wish to use
our facilities, as we do now. Our technologies are being shared by being published, and papers and software will be available electronically. We will maintain a public website to freely share our demonstrations and robot data with additional video material. We will present our work at conferences and publish it in journals, and will use these vehicles to advertise our work to potential collaborators in science and industry. For a more complete description of our Dissemination Plan, please see our Data Management Plan.

Outreach: A major outreach initiative led by Atkeson is the creation of a physical and virtual Robot Museum. So far we have created physical exhibits on juggling robots, robot actuation (gears vs. direct drive), mobile robots, soft robots, Steve Jacobsen and Sarcos, robots in literature, legged robots, computer graphics (Ivan Sutherland), and AI (Newell and Simon). Our next major initiatives are 1) to develop cell phone apps that trigger off augmented reality (AR) tags and robot pictures in halls to provide a self-guided tour of the Robotics Institute, and 2) use virtual reality (VR) to provide access to our collection from anywhere in the world. We want anyone to be able to design, build, debug, evaluate, and repair a historical robot in virtual reality. The impact of our outreach will be increased by a new Disney TV show based on the characters from the Disney movie Big Hero 6, including the inflatable medical robot Baymax inspired by Atkeson’s work on inflatable robots. We will publicize the proposed work in synergy with this TV show by emphasizing we are “Building Baymax’s Brain”. We have coordinated our outreach activities with the larger outreach efforts of CMU’s Robotics Institute to scale up reach and effectiveness.

Participation of Underrepresented Groups: In terms of more general outreach to under-served populations, we will make use of ongoing efforts in the Robotics Institute and CMU-wide. These efforts include supporting minority visits to CMU, recruiting at various conferences and educational institutions, and providing minority fellowships. As the Robotics Institute PhD admissions chair in 2016, Atkeson led a process which resulted in 31% of acceptancesgoing to female applicants. As a member of the Robotics Institute faculty hiring committee in 2017, Atkeson participated in a process that led to approximately half the interviewees being female. Half of the faculty hired were women. As the head of Robotics Institute hiring in 2018, Atkeson led a process in which again approximately half the interviewees were female, and 3 out of the 4 hires were female. Atkeson is assisting efforts at CMU to raise money for fellowships for students who will help us in our efforts to serve diverse populations and communities, including our own. Fragkiadaki organizes the CMU chapter of the AI4ALL national program, with the first version presented in July 2018: a three week program for 20 high school students from disadvantaged local schools, to expose them to the excitement of AI, its potential societal impact, and what people do in college, graduate school, and beyond. The first instantiation went very well, and both the participants and the organizers learned a lot. Preliminary results of the proposed research were presented to the students, who were very enthusiastic.

Technology Transfer: The best way to transfer technology is by having students go to industry. Three recent students work at Boston Dynamics transferring our work in robotics to commercial applications, one recent student and recent postdoc work on self-driving cars at Uber, one recent student works on self-driving cars at Apple, and one recent student works on humanoid robotics at the Toyota Research Institute. An older former student is the CTO of the Amazon drone effort. Several older former students work at Google. We are thrilled that we and our students are part of the robotics revolution.

Curriculum Development Activities: We will develop course material on robot learning and reasoning, which will directly be influenced by the planned activities of this proposal and freely available on the web. The PIs currently teach several courses that will benefit from this material. For example, 10-703: Deep Reinforcement Learning and Control, 10-898: Language Grounding to Vision and Control, and 16-745: Dynamic Optimization directly address the research areas in which this proposal is embedded. We also teach a course designed to attract undergraduates into the field, 16-264: Humanoids.
Appendix: Parts of Previous Proposal: Exploring Hybrid Problem Solvers

Summary of reviews of this previous proposal

The main novelty in this proposal, in the view of some panelists, is the use of behavior graphs to “glue” symbolic and model representations.

The cooking setting at the home institution gives credibility to the capability of the team to perform realistic evaluations on this general area of research.

Integrated AI has been tried multiple times before including large DARPA projects such as Integrated Learning. Those projects didn’t get as far as expected. Panelists didn’t see any compelling reason to believe this project would do better.

- Furthermore, those were large teams and multi-million dollar efforts. This raises the concern that the scope of this proposal is too large compared to the budget asked in this project.
- The project feels like a laundry list of topics that doesn’t integrate well together. This project seems to be consistent of independent sub-projects.
- The PIs are very strong but the proposal is not cohesive and needs to be more specific about how they plan to combine the techniques.

Please study projects related to integrative AI (see one of the reviews for pointers). Where did they fail? what can you do to avoid those pitfalls?

Problem has been studied multiple times with mixed results. Unclear what will make a difference in this proposal.

Review 1 excerpts:

Strengths:
+ Broad research program, touching on practically every issue in robotics and machine learning.
+ Behavior graphs are an promising way to use symbolic reasoning in an RL framework.
+ PIs have a strong history of building real world robust robots.

Weaknesses:
- Proposal reads like a brain-dump of every possible method in reasoning and robotics.
- Kitchen domain is indeed rich, but not particularly original – fine to use, but no points for introducing it.
- Cites lots of work, but no real description of how the proposed work builds up specific prior work – just for one example, manipulation of non-rigid objects.

Team is strong and proposed work is broad, but proposal is little more than a laundry list of ideas and challenges; hard to understand what will lead to specific reproducible scientific advances.

Review 2 excerpts:

A system that combines many heterogenous methods and techniques could easily become too complex and even more brittle than the individual
components
- No clear evaluation metrics are given

The research proposed is interesting and novel, however the proposed plan is very ambitious and not enough detail is given to be able to evaluate how likely it is to succeed.

Review 3 excerpts:

The combination of the many techniques is called hybrid learning and reasoning and the expectation of this proposed work is that the combination of these many approaches will lead to robust behavior of the IPS.

- The later point also raises negative points about this proposal: first, I had a grasp how individual tasks are going to be addressed: the system computes the estimate of the state where the agent wants to be and constantly contrast it with the actual state. With the aim at reducing the error: expected - actual state. Behavior graphs are used to represent complex behavior (plans). Real-time variables are used to reason about things such as trajectories. Each of this individual pieces sound feasible but it is not clear to me how this all will be integrated together. Stepping back at a higher level, the multiple techniques might "pull" the agent into different directions. At a lower level, I don’t see the "glue" putting all of this together.

Review 4 excerpts:

- The proposed research plan is not clear enough, and does not contain any new insights into how integration will be achieved. Integrated AI systems have been attempted many times over the past decades (starting with the classic Pandemonium system from the late 50s, and including more modern systems such as those resulting from the CALO, POIROT and GILA DARPA projects). Although all of those systems showed some interesting benefits from integration, none of them fully succeeded in the task. I did not see any new idea in this proposal that shows that the PIs can succeed where previous work failed.

- The PIs plan to hire 1 PhD student and 1 postdoc, but it seems completely unfeasible to develop a project of this magnitude with just 1 student and one postdoc over 3 years.

While this proposal focuses on very ambitious and intriguing goals, insufficient details on the key ideas the PIs plan to explore to achieve those goals are provided. The proposal does not contain an explanation of the plan they have to succeed in the task of integration of multiple approaches, when a large amount of previous approaches did not fully succeed.
Exploring Hybrid Problem Solvers

Summary

The key feature of the proposed work is combining different approaches in one smart and autonomous system. The system will combine memory-based and knowledge-based components, model-based and model-free reinforcement learning algorithms, numeric and symbolic learning, different levels of modeling, planning, and learning, and also feedforward and feedback control with error monitoring and failure detection. We want to enable robot workers and explorers to make simple plans and solve minor problems autonomously, and be able to attain a safe state and ask for help when major errors or problems occur.

Keywords: robot learning; learning from observation, reinforcement learning; memory-based learning

Intellectual Merit: We believe that hybrid approaches that combine learning methods can be robust and effective. Our ideological commitments are to use a library-based approach, represent both models and policies, use both numerical and symbolic reasoning, support both planning and reacting, represent alternative strategies and behaviors, model at the task level and in the context of a particular task strategy, and focus on optimization as the key reasoning tool. Key intellectual issues are: How does a robot efficiently learn to select appropriate behaviors? How should a robot use simulation to learn to learn (become better at future learning tasks)? How does Robotics go beyond rigid objects to handle deformable and workable objects and liquids and granular material? How do robots learn to use tools? How do robots invent new tools?

Broader Impacts: We expect to make programming robots much cheaper, and make robots more useful, particularly for care robots supporting everyday life activities. This work initially focuses on an everyday task, cooking. We believe this work will eventually also be applicable to robots on the surface and oceans of Earth, in orbit, on the Moon and other bodies in space. These robots could explore and do science, prepare for humans, or perform useful tasks such as map, mine, construction and assembly, supervision, maintenance, and repair, even with long communication latencies and low bandwidth communication. Underwater repair is a domain which often requires a great deal of improvisation as initial diagnoses are often wrong, materials and tools break, weather interferes, and the ability to access or apply forces to do a repair may be limited. Nuclear plant decommissioning also requires improvisation as the actual structure may be damaged or just different from plans, materials may have degraded, and doors and other fixtures may no longer work as desired. On the Moon it would be useful to use lunar materials for large scale construction of solar panels, and variation in lunar materials and conditions may require improvisation as well.
Exploring Hybrid Problem Solvers

We propose to build a smart and autonomous problem solving system by combining alternative approaches along several intellectual dimensions. We hope to both reconcile and take advantage of the complementary qualities of multiple robot planning and learning algorithms in one system. The PI has decades of experience of working with each of these methods in isolation. It is time for us and the field in general to build prototype systems on a larger scale that make use of what we have learned about learning and intelligence.

Combine memory-based and knowledge-based reasoning: ...

Combine model-free and model-based reinforcement learning: ...

Combine numeric and symbolic learning: In order to improvise if things go wrong, we believe robots need to reason both numerically and symbolically. Accurate numerical simulation is often computationally expensive and slow, and we want robots to rapidly reason about physical phenomena. Robots need to abstract, reason with symbols, and communicate symbolically. We will create symbols and abstractions from analysis of large amounts of simulation, practice, and execution data. Some symbols and concepts will correspond to parts of speech: nouns, adjectives, verbs, and adverbs. Based on human labelling of concepts, features, and objects in the libraries, our system will understand and generate simple sentences.

Combine different levels of modeling, planning, and learning: We will model, plan, and learn at a number of temporal and functional scales, including the millisecond time scale of a robot force controller, the time scale of seconds for many behaviors such as a foot step, and the functional level of a sub-task or task such as open a door. Dealing with smaller time and functional scales supports greater compositional and generalizability, but is more vulnerable to prediction and control errors accumulating with time. Dealing with larger time and functional scales limits generalizability, but greatly reduces the effect of modeling and policy errors. We will also make models, plans, and policies at multiple levels. We have found that modeling complex physics in the context of both a task and a particular robot strategy can be more effective than trying to learn general models. For example, modeling shaking salt from a dispenser from first principles is very difficult. Learning models of granular materials in general or for all possible salt dispensers is very difficult. However, learning models of the outcome of a particular shaking strategy applied to a particular salt dispenser is much easier and more accurate. We propose using learned task models at several levels in hierarchical modeling: qualitative/symbolic models that represent different physics and task phases such as in-contact/unconstrained or sliding/stuck; task-level models that represent continuous outcomes of tasks such as how much salt will come out when a particular salt shaker is shaken in a particular way; and detailed process models such as how granular materials will flow or how much friction there will be. This will allow the system to plan and reflect at different levels: more abstract levels support fast reasoning and more generalization, while detailed specific models may be more accurate but are often computationally expensive and slow.

Combine feedforward and feedback control with error monitoring and failure detection: Another aspect we will emphasize is execution monitoring and behavior switching to handle large errors, and a complementary learning approach of remembering planning “bugs”. Explicit monitoring for errors makes it easy to learn by storing planning failures (bugs) and their corrections. Execution monitoring and behavior switching was our (WPI-CMU team) successful strategy in the DRC, where our robot avoided falling or needing human rescue by detecting errors early, transitioning to safe states, and asking for help from the remote mission controllers (Figure 14).

Why is this proposal relevant to the Smart and Autonomous Systems program? Our goal is to develop a robot that operates for extended periods of time with minimal or no supervision by human operators. Our robot will go beyond the capabilities of present-day physical systems. It will be able to learn from experience, as well as use learned models to introspectively examine its own actions and assess the effects of those actions in the environments in which it operates, leading to greater autonomy and range of operations. Learned models will support awareness of the system’s capabilities and limitations. We will use cooking as a rich set of tasks our robot can perform that require planning at many levels, evaluating alternative strategies at many levels, improvisation to handle missing ingredients and tools, awareness of the capabilities of various agents (self, other robot, and assisting humans), and handling errors (Figure 2). Our robot will explore and reflect, finding
better ways of doing things and supporting transfer from other experiences, robots, and humans. Our robot will be cognizant, aware of human needs, desires, patterns, intent as well as system capabilities and limitations; and taskable, able to interpret high-level vague instructions, translating them into concrete actions that are dependent on the particular context (recipe, step, subtask, ...) in which the robot is operating. Our robot will be able to learn from its own experiences and those of other entities, such as other robots or humans, and from instruction, coaching, and observation; and will certainly exhibit self-optimizing capabilities as well as internal models of itself that supports self-awareness. Our knowledge-rich robot will employ a variety of representation and reasoning mechanisms, such as semantic, probabilistic and commonsense reasoning; will be cognitively plausible and able to explain its reasoning; reason about uncertainty and the intentions of other entities in decision making. Our robot will be ethical in that it will adhere to a system of societal and legal rules, taking those rules into account when making decisions.

Some broader impacts:

We believe this work, focusing on cooking, an everyday task that is important for domestic and care robots, will eventually also be applicable to robots on the surface and oceans of Earth, in orbit, on the Moon and other bodies in space ...

Motivation

We believe that hybrid approaches can be more robust. Often the blind spots or weak points of one type of problem solver complement those of an alternative type of problem solver. That leads to our interest in combining memory-rich/knowledge-poor and memory-poor/knowledge-rich approaches to form a memory-rich/knowledge-rich approach. Error-driven/model-free approaches can rescue predictive/model-based approaches when modeling biases causes model-based approaches to fail or get stuck repeating the same error. This is related to how integral control reduces remaining steady state error in control systems. Modeling, planning, and learning at multiple temporal and functional scales helps avoid the pitfalls of focusing on only one scale or type of model. Combining different types of control allows us to obtain the best features of each.

We have chosen cooking as an initial domain because it challenges model-based reinforcement learning approaches. Cooking includes perceiving and manipulating deformable objects, liquids, and granular materials that are difficult to model. Cooking involves different types of processes that are also difficult to model, including cutting, grating, and mashing; thermal (heating and cooling); and chemical processes. Cooking is an interesting domain for other reasons as well. Cooking involves tools (utensils, containers, special-purpose devices, mixers, ovens, stoves, refrigerators, etc.). Cooking is well suited to a library-based approach to knowledge representation, including collections of high level plans (cookbooks full of recipes). There is enough data in the form of how-to and instructional videos and web pages to support data-driven learning approaches. Examples of related tasks involving deformable objects include clearing debris and rubble at a disaster site, many manufacturing processes, and assisting humans with dressing and undressing.

Proposed Research

We are taking the ideological stance that a rich diversity of representational tools is useful, rather than placing all our bets on a more monolithic approach. We expect to develop new representational tools as this and other projects progress. A key challenge in this research is facilitating synergistic interactions between different approaches. Page limits force us to only present selected highlights of our proposed work addressing these issues.

Test domain: cooking: We have chosen a difficult initial domain to develop our ideas and framework: manipulating deformable objects, liquids, and granular materials (Figure 2). Our prior work in this area provides evidence that we can tackle the challenge of combining many methods in one system. Figure 16 shows an example of learning parameters for a primitive, and learning which primitive to use. Goals (reward functions), and sub-goals (rewards at intermediate steps) are shown by demonstrating the task. We developed the proposed framework for learning modular dynamic models and skills in. We developed a stochastic extension of neural networks in that is useful for modeling primitives. We explored deep
reinforcement learning applied to our graph-based approach in [139]. We developed Differential Dynamic Programming (DDP) for graph-structured sequences of primitives in [138]. We developed a stereo vision primitive to estimate liquid and particle flow during pouring [141]. We developed FingerVision, a vision system that both measures the deformation of skin on the robot fingers to measure contact locations and forces, and sees through the transparent skin to provide object and surface localization and tracking during manipulation (in this case cutting with a knife) [137]. We have shown generalization and adaptation abilities, and efficient learning based on both simulated and actual practice. Our and other’s work provides an excellent foundation for the proposed work. Related research on robot cooking includes making pancakes [68] and baking cookies [29]. There is extensive commercial work on robot cooking and serving food [133]. Although there are successful results, the robotic behaviors do not generalize widely, and many systems are proprietary, and thus do not contribute much to the community.

Many forms of knowledge represented in libraries: Figure [17] shows some of the various forms of knowledge we plan to use as our starting point. The question mark emphasizes that this diagram is incomplete. Year 1 will largely focus on manually initializing these components. Year 2 will largely focus on improving the components using several forms of learning and reflection. Year 3 will largely focus on improving the interaction of the components. Although we will iteratively evaluate and refine our system throughout the duration of the project, Year 3 will also focus on finding weaknesses or poor performance and improving the system.

The central element of Figure [17], experience memory, stores data structures (such as frames [101]) that link items from the surrounding modules. The experience memory stores a continuous stream of a robot’s experiences, as well as imagined or hypothesized experiences, and experiences from other robots.

We intend to use off the shelf components for the perception components of the knowledge base (blue background in Figure [17]). The perception library itself stores perception routines, which are used to define a set of features and objects in their respective libraries. For this proposed project the perception library will come from publicly available libraries such as OpenCV and software under development by colleagues at CMU and elsewhere. Features and objects can be hand programmed as well as learned (again using existing software).

The feature library can be extended by transforming or combining existing features, as well as adding new code to the perception library. Objects are recognized by a classifier applied to features, and represented in the object library. New objects can be learned using established approaches. Perception research is not our central focus, so we will use publicly available software and manual input for this part of the system (as we have done in the preliminary research).

We explicitly represent goals, subgoals, goal sets, constraints, optimization criteria, and reward functions that this system or similar systems have had in the past or are currently trying to achieve in a goal library. Goals can be explicitly programmed, taught, or coached by a human supervisor. Goals will also be learned using inverse reinforcement learning. This explicit representation of goals supports generalizing knowledge between tasks with similar goals as well as reflection to improve performance using mental practice. We will explore how well ethics can be manually programmed into the goal library to shape and limit behavior.
The abstraction library represents heuristic information about possible relationships between features and objects as well as qualitative models. It represents heuristic ontologies and analogies. It helps guide search as to what features and objects might be relevant to achieving a particular goal. This library will suggest starting points for feature, model structure, and policy structure selection by the system. It is important to note that this library stores heuristic knowledge, that is often but not always correct. The performance measure to be applied to this library is how much it speeds up feature selection and generating new strategies in more formal search guided by optimization.

The model library represents numerical models at several levels of detail, time scale, and functional scale. These models can be programmed, or learned using traditional system identification techniques (typically parametric) or more recent machine learning approaches (parametric and non-parametric). The PI has extensive experience in system identification as well as non-parametric memory-based locally weighted learning and parametric neural network techniques [12, 80, 82, 81, 105, 103, 106, 104]. The performance measure for this library is how well it supports model re-use, and speeds up system identification and model learning.

The action library: We generalize the concept of action to include policies, and plans at several levels of detail, temporal scale (milliseconds to minutes or hours), and functional scale (instantaneous actuator commands to high level synergies) in the action library. These generalized actions can be programmed, learned from observation, learned from practice, or learned from simulation (mental practice) if adequate models are available. We represent generalized symbolic and numeric actions and policies as combinations of primitives (also known as options, macros, chunks, schemas, basic behaviors, subroutines, operators, ...) which take input states and parameters and generate a behavior which leads to an output state [9, 18, 101, 109]. This library is key to the entire effort, and will be a major focus of our research. We will be guided by related work on behavior libraries: In computer animation, the use and adaptation of databases of human poses and motion are common [75, 102]. In artificial intelligence, the idea of storing, re-using, and adapting plans has a long history [84, 117]. A number of efforts have been made to use collections of stored trajectories to
represent policies, including [69, 51, 52, 17, 108]. Library-based approaches in robot planning and control include [10, 13, 47, 46, 96, 114, 112, 113, 79, 35, 87, 121, 98, 25].

**Reasoning processes:** Here we briefly list some of the reasoning processes we plan to start with. Goal selection provides a way for operators to set or shape the robot’s task. A planning process (actually a set of competing planning processes) attempts to continually find reasonable plans. State estimation is used to update system-wide estimates of the current situation. Execution uses the currently selected policy to generate actions. Prediction processes are constantly estimating what will happen next on a variety of time scales, while error monitoring detects errors that require behavior switching or more severe safety or recovery behaviors. Ethics and safety monitoring provides an independent estimate of when the system is violating whatever “Laws of Robotics” are currently operational. Reflection (which can actually be performed both online and offline in the cloud) sets virtual goals and attempts to achieve them in simulation (mental practice), and suggests new features, objects, subgoals, task models, and policy and other primitives for learning modules to use in the future (learning to learn).

**Symbolic policy and model representation: behavior graphs:** We will represent a framework for a set of skills involved in a specific robot task or computation using a directed graph we call a behavior graph (Figure 5). Nodes of a behavior graph represent generalized states. Edges represent generalized actions. Behavior graphs enable us to modularize learning, and more easily interpret and generate a linguistic interaction with a human user. We will initially manually design behavior graphs to initialize our systems. We will then explore learning from demonstration methods to extract graph structures from human demonstrations (e.g. [86]). Previous work on semantic action representation includes [49, 130, 21], and some of them are used to reason about robot behaviors [49, 130]. In the framework of human behavior analysis (activity recognition), segmented and semantically analyzed behavior representations are created from videos of human demonstrations (for example [2, 5, 65, 97]).

**Representing alternative strategies:** A major research issue is handling situations where there are multiple strategies available to perform tasks, and it is useful to be able to consider many of them in problem solving. We will extend behavior graphs with a standardized selection primitive that allows the graph to bifurcate to represent alternative strategies, and what the robot has learned so far to guide future choices between those strategies. We will explore problem solving methods that take advantage of precalculating or learning multiple possible plans (often from simulation). We present one approach here. The novel feature of our approach is that we store and provide multiple solutions at every training point. We hypothesize that this will allow our solution method to develop multiple solutions and then pick the best one for a specific problem, rather than commit too early to one possible solution or strategy.

In the field of function optimization having many comparable but significantly different local minima is referred to as “multimodal” optimization. Evolutionary algorithms maintain a population of possible solutions at all times [119, 4, 33]. Typically terms are added to the optimization criterion to reward “diversity” in the population of solutions. Evolutionary algorithms are more general than genetic algorithms, in that any representation and update operators can be used, rather than just a linear “genetic code” and update operators modeled on biology such as mutation and crossover. In our experience evolutionary algorithms can find good solutions, but are not fast enough for online use.

We describe a memory-based approach to this problem, using the inverse kinematics necessary to move debris and perform assembly as a simple but realistic example task. The first issue is defining the index that will be used to store and index plans. For this presentation we will focus on long objects such as pipes, planks, or bars (this was true for the DRC, and simplifies the presentation here). We can describe each piece with the 3D location of one end, its orientation, and a value describing the length of the debris. This is a seven dimensional index vector that describes that component of the task.

The second issue is to define the database or memory format. Our plan to use the simplest possible representation initially. For practical database sizes, the most effective implementation of nearest neighbor lookup may simply be to store the indices in a list or array, and exhaustively search through the entire database on every query. If this proves too slow, we will take advantage of our previous work in memory-based learning [12, 105, 103, 106, 104]. Tree data structures such as kd-trees can be adapted to store and access multiple
values at a point, using nearest neighbor techniques to do access. We can probably use approximate nearest neighbor algorithms, which are much faster than exact nearest neighbor algorithms [129].

The third issue is where the training data will come from. What we need is to have the training data reflect the distribution of tasks the robot will actually have to do. We will initially use human teleoperation of a robot to characterize the task distribution. We will also construct a behavior generator, which can generate new behavior similar to the known behavior, greatly increasing the amount of training data and allowing deep learning techniques to be used.

The fourth issue is developing a system that can use the precalculated results and generate actual solutions fast enough to be used in real time on a robot. Stored plans must be accessed, adapted for the current query, and then used to generate a complete behavior. In the debris example this includes approach, envelop, grasp, lift, move, release, and return phases. We will use the gradient optimization techniques to adapt a plan for the current situation. It is an empirical question as to which gradient algorithm is more effective (first vs. second order gradient, for example).

The fifth issue is keeping the final behavior library small. We will only store “surprises”. If similar solutions are already available for the query point, we will not store new data.

We are not aware of any prior precalculation or memory-based learning system that stores multiple alternative answers for each point, and provides multiple different appropriate answers on each access, so there will be much to do in designing and tuning these algorithm components. We will explore a range of indexing schemes, training data generation, memory architectures and algorithms, and run-time systems. [12, 105, 103, 106, 104] describe our prior work on optimizing distance metrics for this type of memory. Cross validation approaches from machine learning will be used to optimize algorithm parameters.

After initially developing a simple version of this system, we will explore using the multimodal memory to store alternative higher level task strategies, such as whether to move to the left or right of the robot, or whether to use one or two hands to grasp. In this case we will develop offline planners that generate solutions for each possible strategy, and then store them. An interesting research question is whether we should prune strategies that don’t work well for a particular task index, or store them anyway so the system does not “forget” a failed strategy and then have to re-invent it repeatedly in that region of the task space.

There is recent work in machine learning on multimodal learning [100]. In work on grasping, here are some representative memory-based approaches where sets of grasps are stored and used to guide grasp planning [26, 75, 36]. An example of an evolutionary algorithms/multimodal optimization approach to six degree of freedom inverse kinematics is [119]. A Bayesian approach to multimodal learning represents multiple answers to a query by representing a probability density function and choosing local maximums or sampling from the density [94, 50]. Qin and colleagues present a particle filter-based approach to inverse kinematics [94]. Grochow and colleagues present a memory-based approach to inverse kinematics [50]. What is represented in the memory is a probability distribution of human figure poses. The representation used is Scaled Gaussian Process Latent Variable Model (SGPLVM) [70]. This representation automatically reduces the dimensionality of what is represented (typically to 3 dimensions). Any representation that learns densities can be used, such as mixture of Gaussians, but automatic dimensionality reduction makes generating a likely pose easier. To generate a likely pose, candidate solutions are generated and then optimized using gradient-based inverse kinematics, as described in this proposal. The major difference between this approach and our approach is that this approach indexes by poses \( q \) and we index by indices related to the task. We will experiment using aspects of this system from computer animation, such as the dimensionality reduction in SGPLVM for our task indices, and see how well it applies to robotics.

**Facilitate re-sequencing and mutating primitives:** Our initial approach will be to represent input and output parameter sets using associative lists that are sets of triples (label, data-type, value), as is done in ROS messages, so during learning the information in the nodes can be used in many ways and redirected, connecting nodes to new primitives (Figure 5). The use of associative lists allows parameter sets to be unordered, and outputs of one primitive to be rerouted to inputs of another primitive by just matching data-types, or matching labels as well (as in ROS). Elements can be duplicated or dropped to handle different numbers of outputs and
inputs. Various default or random values could be used if no output matched a particular input data-type or label. The dimensionality of these lists and their labels and contents depend on the current active node(s).

**Formalizing the representation:** We need to develop a formal representation that supports selection among alternative strategies. We will sketch out our anticipated initial path to formalize this representation and the reasoning algorithms that use it. We propose representing discrete and continuous components of generalized states and actions separately. We represent task and computational states $s$ with the triplet $(n, z, x)$ where $n$ is the current node in the behavior graph, $z$ is an associative list of the other discrete components of the state, and $x$ is an associative list of the continuous components of the state (the traditional state vector and belief state in control theory). We represent actions and computations $a$ with the pair $(v, u)$ where $v$ is an associative list of the discrete components of the action, and $u$ is an associative list of the continuous components of the action (the traditional action vector in control theory). To denote that the dimensionalities and even definitions of $z$, $x$, $v$, and $u$ depend on the current node in the graph, all of the variables can be subscripted by $n$: $s_n$, $z_n$, $x_n$, $a_n$, $v_n$, and $u_n$. We will suppress this part of the notation for clarity.

The deterministic dynamics are given by $s' = f(s, a)$ which expands into $f(n, z, x, v, u)$. The apostrophe $'$ indicates an output or changed state. A one step reward (or cost in optimal control) is $r(s, a) = r(n, z, x, v, u)$. Note that $r()$ is the total reward of running a particular primitive or algorithm for a particular time interval, and may be the sum of more “atomic” rewards or costs over that interval of time.

**Formalizing the reasoning algorithms:** Because this notation becomes confusing very quickly (part of the proposed research will be trying to find a better way to express these ideas), we will present the deterministic case first. We will also suppress the dependence on the “sampling time” generated by applying the concepts at the task level. The dependence of the reward of a particular primitive or algorithm on the length of time it is applied can be made explicit, at the cost of further notational complexity, which we will avoid. If the optimality principle holds, in the deterministic case the $Q$ function (lifetime reward for choosing an action in a state) can be computed from the reward $r()$ and value function $V()$: $Q(n, z, x, v, u) = r(n, z, x, v, u) + V(f(n, z, x, v, u))$ and the value function is given by $V(s) = \max_a Q(n, z, x, a)$, which expands into

$$V(n, z, x) = \max_v \max_u Q(n, z, x, v, u).$$

The maximization with respect to $u$ can be performed using some form of gradient ascent with the first order gradient $\frac{\partial Q}{\partial u}$. This gradient is the sum of $\frac{\partial r}{\partial u}$ and $\frac{\partial V}{\partial x} \frac{\partial x}{\partial u}$ where $x'$ is the continuous component of the next state and $u$ is not at a discontinuity of the dynamics and does not cause a change in the next node or next discrete state variables. Separating out discrete and continuous variables allows us to represent gradients for gradient-based optimization algorithms. The maximization with respect to $v$ is discrete and must be done using some discrete optimization algorithm. In our preliminary work [138], we have used a multi-start gradient-based optimization where we maintain multiple search points that covers the discrete action space of $v$. Each of search points have different $u$ that are optimized with a gradient based optimization ADADELTA [146].

We propose using continuous trajectory optimization, in the form of Differential Dynamic Programming (DDP), combined with discrete optimization applied to a planned task execution to optimize robot policies $a_n(s_n)$, by performing gradient ascent on the value function computed above at each step of the behavior sequence. After a trajectory (a sequence of states interleaved with primitives) is computed, DDP computes the derivative of the $Q$ function at each state $\frac{\partial Q(s, a)}{\partial u}$ to optimize (gradient ascent) the corresponding action with respect to the continuous action variables $u$. Taking into account future consequences of the action is done using the chain rule.

Using the subscript notation for derivatives $Q_u = \frac{\partial Q(s, a)}{\partial u}$ we can sketch out the backwards (in time) recursion of DDP to compute $Q_u$: $Q_u = r_u + V_x' f_u$ and $V_x = r_x + V'_x f_x$, where primes $'$ indicate the execution of the next primitive. Situations where there is an output discontinuity or a change in discrete output for a primitive in the sequence along the current trajectory complicate this computation, which is one reason why we attempt to smooth out such discontinuities and replace discrete outputs with continuous outputs in what follows.

The stochastic case is more complex because expectations over the random variables need to be taken. The reality is that much of our reasoning and learning boils down to repeated application of the chain rule (as is true for neural networks and deep learning as well). The proposed research will try to find ways to apply this
approach to generalized states and actions in perception, reasoning, and learning algorithms. We can avoid the curse of dimensionality using DDP, at the cost of only finding locally optimal perception, reasoning, and learning algorithm settings. The computational cost of DDP is only proportional to the dimensionality of $x$ cubed, times the number of discrete action sequences considered, rather than exponential in the dimensionality of $x$. DDP can be applied to single task executions to optimize perception and reasoning algorithms, and sequences of task executions to optimize learning algorithms. We believe some combination of Monte Carlo and gradient-based optimization will be able to find good perception, reasoning, and learning algorithm selections and settings for realistic problems.

How can we use this approach to efficiently learn and feature selection and generate new strategies? We will explore using first and second order gradient descent in continuous spaces, using derivative estimates to speed up search. Searching over discrete variables is much more expensive. The success of deep learning suggests that converting discrete selection search (features, model and policy structures, choice of primitives, and orderings) to a continuous search (gradient descent on neural network parameters) is helpful. We are trying to apply that insight more generally and cleanly. This technique is known as embedding, relaxation, and to some extend continuation in applied mathematics. We will explore using several approaches to turn discrete optimization problems into continuous optimization problems, with the hope of speeding up search and finding better answers. We can do this if the discrete variables are binary choices, or can be put in some kind of order (even an artificial order that seems unrelated to the task). Counts and numbers of items (both integers) can be ordered, for example. We note that scaling of variables to avoid making the problems too numerically “stiff” will be important.

**Developing context-dependent high level representations:** A major problem we did not resolve in the DRC Challenge and which we will attack in the proposed research is how to modify more abstract representations when a plan failed at the lower levels, such as due to a self-collision, for reasons that cannot be represented at the higher level (since the higher level does not represent the body shape, for example). A hack we explored is to “poison” parts of the high level abstract space, so that future plans avoid this area. This is a crude and worst case approach. We will explore better ways to solve this problem, such as adding rich representations of context to the abstract levels, to force alternative plans to be developed or express preferences for the use of alternative planning techniques or planners in certain situations where the default planning approach fails.

**Enable programmed and learned primitives:** Primitives can be manually designed, and many will be engineered to initialize our system. This is an excellent area for outreach in the form of students across the country programming primitives and trying them out in a web-based simulation environment. Primitives can be represented by parametric structures like deep neural networks, and trained or learned using some appropriate source of data. For example, the output of an expensive engineered algorithm like an optimization might be used to train a neural network. The neural network provides an approximate answer to the optimization much faster than the original algorithm.

**Developing continuous feature selection and dimensionality reduction:** We believe human motor learning is effective because humans identify the most important direction in command space and learn that first. Only later are more dimensions added to learning in a principal components-like process to deal with the true dimensionality of the task. We can reduce the dimensionality of a primitive by restricting its input to a subspace. The most extreme case would restrict the input to a scalar or along a line. This can greatly speed up learning if the direction of the line is well chosen. An interesting question is how to choose the most useful direction. One way is to use simulation with learned probabilistic models to evaluate learning in a particular search direction, a form of Monte Carlo evaluation of search directions. It may be useful to choose good dimensionality reductions by preserving the input directions that have high reward function curvature. Alternative approaches to feature selection include the forwards and backwards selection algorithms used in statistics. Principal Components Analysis, Independent Component Analysis, Canonical Correlation Analysis, Fisher’s Linear Discriminant, Topic Models, Latent Dirichlet Allocation, and Factor Analysis are all alternative approaches to feature selection and dimensionality reduction we will explore, compare to, and use to initialize our search processes. We will develop algorithms to explicitly search for sub-goals or intermediate goals, as well as sim-
plifications of tasks that enhance learning (shaping). We can use deliberate or random modifications (mutations) of existing primitives to generate new primitives. The research challenge is to make the mutated primitives likely to be useful. We can add $\alpha$-like “slack variables” for primitives applied to different possible feature sets, to do continuous optimization-based search for the most useful features. Related work in neural-network-based (deep) learning includes \cite{41, 59, 92, 53, 42, 44, 54, 55, 71, 74, 76, 93, 120, 147, 56}.

**Explore different ways to mix methods:** We will explore different ways to mix methods such as model-based and model-free reinforcement learning. One hybrid method is a simple succession. First model-based methods are used, each time retraining the model after an optimized behavior is tried out on the real system. When the model stops changing, then apply a model-free method. An alternating hybrid method would continue this sequence of model-based, model-free, model-based, ... A blending hybrid method might use a weighted combination of the outputs of both methods.

**Use big simulation to invent new primitives and heuristics:** We will explore automatic generation of primitives, heuristics, and abstractions based on *Big Simulation*. Previous work on discovery of primitives includes \cite{43, 60, 77, 19}. The data we have to operate with is simulations of trajectories, along with all the internal states generated by the various system modules. There are several heuristics that can be used to guide this search. Methods to identify primitives, heuristics, and abstractions include: clustering and data compression techniques, searching for attractors in the state space, matching templates, isolating regions with a simple behavior, physical reasoning, and using critical events to segment component behaviors. There are certain kinds of behaviors that occur often in many tasks, and one can use templates to identify them. An example is pumping, where energy is stored in an oscillatory mode of the system and then used to attain a higher energy state (for example in the swing up task, where a pendulum might be started swinging to make swing up to the inverted state easier). It may be possible to segment out regions of the behavior that are simple to describe (being either zero, constant, or a linear function of state), which suggest a behavioral primitive. Critical events, generally discontinuities in one or more dimensions of the trajectory flow, are generally a useful guide to behavior boundaries. The critical events often occur on a lower dimensional manifold in state space, and this manifold can be explicitly identified. Physical reasoning, especially about crude forms of causality, may be useful to identify sub-goals. Reasoning about support against gravity, and which items in the environment could possibly be providing forces to drive a trajectory, may be useful, for example. We will explicitly explore different state representations (absolute vs. relative positions, for example), to see which best support generalization in simulated learning.
References


[34] C. Dawes. Twist and shout. [www.packagingnews.co.uk/design](http://www.packagingnews.co.uk/design). 2009.


[126] Pretty Good Videos. 3 easiest ways to a open stubborn jar — easy life hack. www.youtube.com/watch?v=OL0mHgGWd9w accessed 2018.


