Using A Hierarchical Architecture for Collaborative Social Tasks to Reinforce and Recognize Action Primitives

Author 1, Author 2, Author 3
University
Address
Address
Address

Abstract

Recent work has shown that with human assistance, robots can learn to perform complex actions with limited training data. The resulting skill representations can be leveraged to replay flexible behaviors that dramatically increase the utility of the robot. Despite this, existing systems have yet to address the greater problem of cooperative action execution. For a robot to effectively cooperate with either human or robotic coworkers, the robot must be capable of modeling and recognizing its coworkers' behaviors within its environment. In response to this perceived need, we present the RAPTOR algorithm: Reinforcing Action Primitives Through Observation and Recognition. We approach the task of understanding and predicting coworker actions by providing a novel, fast algorithm allowing a robot to solve skill recognition, the inverse problem of skill execution, in real-time. Our algorithm provides a coded timeline of primitive actions generated in real-time, correcting its hypotheses as more information is presented. We demonstrate our results through the recognition of selected American Sign Language gestures. Our work is unique in that it extends the already broad applicability of Q-Learning within skill modeling to create a unified data structure capable of skill recognition in addition to skill execution. The data produced by our algorithm can then be directly used to facilitate the creation of a skill hierarchy fully decomposing a complex cooperative task into primitive actions.

Introduction

As research in recent years has demonstrated, robots operating in a collaborative environment (co-robots) need to be flexible agents, adapting to the changing needs of their daily operations. An excellent example of such a system is Robonaut, a humanoid robot designed to match the dexterity of a suited astronaut (Bluethmann et al. 2003). This platform is designed for both autonomous collaboration and teleoperative capabilities, a desirable set of operating modes for assisting human coworkers in the particularly challenging environment of outer space.

The introduction of such systems requires a substantial effort not merely to produce capable and robust robots, but also to generate a careful and planned interaction method allowing non-expert operators to contribute to the robot's training with a non-obstructive, intuitive procedure. Learning from Demonstration (LfD) (Argall et al. 2009) is a common method for teaching low-level skills required to contribute to a collaborative effort. Therefore, it is important that a co-robot not only achieve competency through direct skill training, but also that it use observations of others in its everyday behaviors to reinforce and improve existing abilities. For systems embodying these characteristics, maintaining an awareness of peer actions is critical for safety, efficiency, and general performance. Direct skill training is still, however, a vital part of a co-robot's success. Knox et. al. (2011) continue to make progress towards allowing non-experts to guide skill acquisition in such systems, facilitating the design of agents with shapeable behaviors.

We propose the Reinforcing Action Primitives Through Observation and Recognition (RAPTOR) algorithm, a real-time, novel method for online action classification that robotic agents can leverage to:

- Categorize a set of observed actions into labeled primitives and unknowns using Semi-Markov Decision Processes (SMDPs), the existing standard of skill representation in Q-Learning based systems (Sutton, Precup, and Singh 1999)
- Help achieve this awareness while simultaneously using the observed data to improve its own performance and understanding of low level actions
- Facilitate the creation of a generic skill hierarchy fully decomposing a complex cooperative task into primitive actions
- Utilize the generated skill hierarchy in a generic hierarchical learning system in order to simultaneously execute primitive actions and recognize known skills being performed by co-workers

We demonstrate our system's functionality by automating identification of selected American Sign Language gestures, while simultaneously using that observed data to improve our agent's internal representation of the underlying primitive skills involved. We then present the significance of this contribution, grounded within the rapidly expanding field of collaborative robotics.

Background and Related Work

Portable Skill Acquisition

Several methods have been proposed for automating skill acquisition in robots. Many of these rely on using expert knowledge to transfer primitive actions within specific domains. LfD, by contrast, is a technique that relies on recorded samples as a means for experts and non-experts alike to train robots on primitive actions. The automation and accessibility of such methods have increased the complexity and variety of skills that a robotic system can successfully acquire.

How such a robotic system might represent primitive actions can vary. Traditional robotic systems often use Markov Decision Processes in combination with Q-Learning to accomplish this goal. The result of such an effort is both a *Q-Table*, representing the known states in a given environment and the corresponding rewards for transitions between them, and a *policy* indicating all possible executions of a skill. Such a policy, in the simplest case, can be represented as a greedy traversal of the Markov Decision Process (MDP) until the goal state is reached.

Because a Q-Table can be viewed as an arbitrary n-dimensional representation of a given state space, it can be leveraged to classify the observed execution of known policies. For example, consider a state space representing positions of an agent's right hand, right elbow, and right shoulder in three-dimensions. A robotic system with a predefined dictionary of gestures could use standard statistical methods to decompose sensor data into timeframes labeled according to which of the robot's policies most closely matched during that interval. In fact, that is exactly what our system does.

It is important to note that gesture recognition is not the only domain in which our system is viable. Several common motor tasks, such as tool manipulation and collaborative problem solving requiring an external perspective (e.g., Trafton et al. 2005) are well-suited to utilizing a previously acquired set of action policies.

The notion of *skill transfer*, repurposing learned data to be applicable in novel contexts, is not wholly original. Konidaris et al. (2011) have successfully leveraged existing learned skills to perform manipulation of simple tools. However, our major contribution is that the skill transfer resides not in transferring knowledge across domains, but rather enhancing the utility of training data through reuse and reinforcement during object recognition. Moreover, because we utilize primitive actions learned independently during the observation stage, we can generalize the observed samples to include arbitrarily many agents that can be observed by spawning several "recognizer threads" at once.

Hierarchical Skill Architectures

In robotics, two areas of reinforcement learning research are particularly relevant to co-robot advancement: autonomously building options hierarchies and reducing the trials or information required to achieve new proficiencies. The former heavily overlaps with non-robot-centric reinforcement learning work, which aims to make high-dimensional, continuous domain problem spaces tractable. General solutions are difficult to formulate, due to the vastness of potential representations and goals for such systems. Hierarchical learning systems follow the philosophy that these highly complex problems can likely be broken into a number of simpler, tractable problems. Finding the most efficient way to discover and arrange the low-level primitive actions is an open problem, with many viable approaches (e.g., Digney 1998; Konidaris and Barto 2009; Mugan and Kuipers 2009). A major advantage of our system is that it can easily interface with any of these general solutions.

Konidaris et. al. (2011) have successfully shown that hierarchical learning can be used to enable skill transfer between environments, identifying irrelevant dimensions of state space and decoupling them from the option representation. By extending the usefulness of a skill laboriously acquired through trial and error, the ratio of the cost of the learned action to its overall utility is reduced substantially. Automatically recognizing instances where already-known actions can be transferred and applied would constitute a significant advance for collaborative robotics. This is highlighted by the increasing incorporation of "demonstration as reward signal", which is becoming more prevalent as the associated computational costs and challenges diminish.

A hierarchical skill architecture is useful for collaborative task execution. To maximally benefit the collaborative aspects of complex task execution, all agents involved must have reasonable approximations of the intent and plan of each other worker. Humans are naturally receptive to skill scaffolding, expressing a complicated task in terms of actions, sub-actions, and temporal sequencing; a hierarchical skill architecture allows the intentions of each agent to be effectively communicated in a common format. One of the greatest technical challenges of collaborative exercises between multiple agents including both humans and robots is the mutual recognition of intention and action. RAPTOR attempts to solve this problem by allowing robotic agents to interpret the actions of others through behaviors with which they are familiar.

Work by Martinson et. al. (2002) shows the benefit of using combinations of pretested behavioral assemblages to successfully accomplish a complex goal in a dynamic environment while simultaneously simplifying the process by which the operator estimates the agent's intent. They associate a list of sensor states, referenced as perceptual triggers, with constructed sets of pretrained actions with stable policies. This decision reduces the MDP search space when considering potential skills to apply, while enhancing the insight human operators have into the agent's potential actions by providing an intuitive mechanism for specifying action/reaction pairs. Mugan and Kuipers (2009) approached this problem similarly to Konidaris, in that they look to autonomously abstract portable options that can be transferred between environments from their agent's experiences. Particularly of note, the QLAP algorithm operates through qualitative representations of the continuous world (Mugan and Kuipers 2009). This provides a tolerance for incomplete input information while disregarding irrelevant dimensions of the state space. As such, QLAP learns action hierarchies while maintaining both temporal and state abstrac-

tion. From a practical standpoint, maintaining state abstraction is crucial for maintaining usable performance levels as most real-world examples cannot be directly approached due to intractably large state spaces.

Q-Learning in Robotics

Markov Decision Processes provide a convenient and efficient mechanism to create flexible, arbitrarily complex options: closed-loop policies describing action sequences (Sutton, Precup, and Singh 1999). The temporal abstraction and generality of representation make this a favorable method of internally representing knowledge about actionable skills. When designing for a non-expert audience, it is often favorable to trade complexity (and, on occasion, optimal resource allocation) for simplicity. This accessibility contributes to Q-Learning (Watkins 1989) being one of the most widely utilized reinforcement learning methods within robotics. With an environmental reward function, solving for an optimal action policy can often be accomplished autonomously and is only limited by the complexity of the state representation.

Because the exploration space of generic Q-Learning problems scale exponentially with dimensionality, heuristics that reduce the number of trials required to achieve desirable policies are actively being researched. One example heuristic uses human feedback as both an immediate and anticipatory reward signal, leveraging human foresight to effectively reduce the search depth required to learn acceptable paths through state space (Knox and Stone 2008). Another effective heuristic involves leveraging data obtained through observing demonstrations of skills to favorably influence transition probabilities within the MDP, achieving accelerated convergence to an acceptable policy (Argall et al. 2009). Our algorithm extends this capability by performing a skill classification step as well. Systems like these reduce the number of trials required by several orders of magnitude with minimal sacrifice in resulting skill quality.

For a co-robot to interact as a productive member of a team, it must be able to recognize and correctly classify peer behaviors. Our algorithm leverages Q-Learning to do this. By improving the agent's situational awareness, intelligent cooperation is facilitated, which can result in greater productivity than a sole agent executing actions independently. Furthermore, leveraging the common MDP representation of action primitives during observation allows a unified internal skill representation whether the agent is responding to coworkers or performing an action itself.

Definition of Problem Space

The agent and operating environment are represented by a continuous-state, continuous-time Semi-Markov decision process, defined by the tuple $(S, A, \mathbb{P}, \Lambda, \gamma, D, R)$, where S represents a continuous set of states, A is a set of actions, $\mathbb{P}$ is a set of policies referred to as primitive actions, Λ is a set of $\epsilon-$neighborhoods defined for readings from each input sensor comprising the state descriptor, $\gamma \in [0, 1)$ as a discount factor, D is the initial-state distribution, and $R : S \times A \to \mathbb{R}$ is an unknown state-action reward function. This data structure is representative of those commonly utilized within skill training and execution systems. We further define the state vector $\vec{v} \in S$ as $\vec{v} \in \mathbb{R}^k$ for k sensor inputs.

We assume the agent holds limited a priori knowledge of a set of primitive skills $\mathbb{P}$, each describing a policy P_π for traversing the agent-environment SMDP. Each primitive action contains its own exploration function, P_e, to guide its traversal while solving for known branches of the problem-space MDP. Additionally, each primitive $P \in \mathbb{P}$ contains metadata indicating the estimated time required for completion of the primitive, P_t. Completion of a primitive action is defined as traversing state space from an unknown initial state to within the $\epsilon-$neighborhood of a known goal state.

Each policy P_π can be described as a series of decision rules to be applied when appearing in familiar situations. These decisions may range from completely deterministic to fully stochastic, and are influenced by the exploration function P_e. Within each primitive, we introduce the notion of reward layers, which are used to influence exploration transition probabilities but not spread during reward signal propagation.

Our environment definition does not assume a uniform sensor refresh rate, as states may be sampled with stale data from a subset of its input sensors. The algorithm presented has shown robustness to lossy and noisy sampling by remaining capable of confidently classifying observations under such conditions within our proof-of-concept implementation.

Primitive Action Recognition

Given a continuous feed of sensor inputs, we seek to determine if any skills known to the agent are being demonstrated. We utilize SMDPs constructed for skill execution to estimate whether or not the incoming information can applicably benefit the agent's understanding of its known primitive actions. RAPTOR performs opportunistic inverse reinforcement learning (Ng and Russell 2000) over multiple policies through continuous observation.

RAPTOR Algorithm Specification

Initialize $P_{\text{hit}} = \{\varnothing\}$ for each $P \in \mathbb{P}$

While (True):

1. Poll each sensor to generate a composite state descriptor vector, $\vec{v} \in \mathbb{R}^k$.
2. For each $P \in \mathbb{P}$:
   (a) If the observed input state possesses non-zero action-reward values through incoming transitions originating at the immediately prior observed world state, it is added to P_{hit}. Previously unobserved states within the $\epsilon-$neighborhood of known states are given copies of incoming and outgoing action-reward transitions of these 'nearby' states, with reward values weighted inversely proportional to distance in $\mathbb{R}^k$.
   (b) If the duration d represented between $P_{\text{hit}}[0]$ and the current time is greater than a defined tolerance constant $t{*}P_t$: erase states off the front of P_{hit} until the $d \leq t{*}P_t$ seconds.

(c) If either the duration d represented within P_{hit} is too short and fails to satisfy $d \geq 1/t * P_t$ or if the density of P_{hit} is less than a predefined accuracy constant a, continue to the next primitive $P \in \mathbb{P}$ at step (2).

(d) For a previously specified neighborhood-distance relaxation constant $d_g \in [1, \infty)$, if the currently observed state is within the $(d_g * \epsilon)-$neighborhood of a known expert-trained goal state:
Compute a confidence $C_P \in [0, 1]$ representing likelihood that P_{hit} is an example of P as dictated by:

$$\begin{aligned} C_P &= \alpha_1 * \text{pathlen}_a + \alpha_2 * \text{pathlen}_o \\ &\quad + \alpha_3 * \frac{\bar{R}_{\text{actual}}}{\bar{R}_{\text{optimist}}} + \alpha_4 * \bar{R}_{\text{actual}} \end{aligned}$$

subject to the constraint

$$\sum_{i=0}^{4} \alpha_i = 1.0$$

i. Generate an 'optimistic' policy π_o. This is accomplished by adding states from P_{hit} into a set W_p with probability p_{wp}. For each state ("waypoint") in W_p, modify all incoming action-reward transitions by adding a reward bonus w to bias towards generating policies favoring transitions through waypointed states. We utilize these policies of idealized traversal for reward propagation, encouraging the agent to traverse similar paths during action execution, influencing the agent's behaviors through observations of its peers.

ii. Compute a path through MDP-space according to the optimistic policy, beginning at state $P_{\text{hit}}[0]$, terminating when within a reduced $\epsilon-$neighborhood of an expert-trained goal state in P or when no outgoing transitions remain. We define $\bar{R}_{\text{optimist}}$ as the mean transition reward sustained throughout the execution of π_o.

iii. Repeat (ii), following a policy closely resembling the observed data, π_a. We define $\bar{R}_{\text{actual}}$ as the mean transition reward sustained throughout the execution of π_a.

iv. Compute the number of states traversed following policy π_a to completion:

$$\begin{aligned} \text{pathlen}_a &= 1 - (\text{sampling rate} \\ &\quad * (\text{states traversed via } \pi_a) - P_t)/P_t \end{aligned}$$

v. Compute the number of states traversed following policy π_o to completion:

$$\begin{aligned} \text{pathlen}_o &= 1 - (\text{sampling rate} \\ &\quad * (\text{states traversed via } \pi_o) - P_t)/P_t \end{aligned}$$

(e) Apply a (C_P *score*, *label*) tuple to each frame of input data represented within P_{hit}.

(f) Given a high-confidence threshold h_c, if $C_P \geq h_c$:

i. Permanently apply additional reward coefficient $\beta_{\text{reward}} \in [1., \infty)$ to boost values of all action-reward transitions utilized in the traversal through π_a.

ii. Set $P_{\text{hit}} = \{s_i | \{s_i \in P_{\text{hit}}\} \cap \{i > |P_{\text{hit}}|/2\}\}$

Details and Analysis

Our algorithm allows a spectating agent to generate a segmented timeline of actions from those partially trained within its internal repository. The agent is then able to fully capitalize on these exemplars within its existing hierarchical framework by identifying possible expert examples to process as training data. The algorithm leverages particularly strong demonstrations to propagate what is perceived as the most beneficial reward signals while processing the identification step, contributing towards converging on an optimal execution policy. This synthesis of action recognition and inverse reinforcement learning does not demand any additional resources or training beyond what is required for basic primitive execution proficiency.

We introduce the concept of probabilistically "waypointing" states along observed paths to generate a hybrid policy allowing for increased noise and error tolerance. Waypointed states have temporary, artificially high incoming transition rewards, biasing exploration functions to choose traversals incorporating them. We combine the use of waypointed states with greedy, non-exploratory policies to create an *optimistic* assessment of how a particular observed MDP traversal can be combined with what is already known about a particular skill. This step is especially important when considering the consequences of implicitly attempting to declare which skill is being demonstrated, as the learner must be able to check if the observed decision sequence is at all relatable to its internal model of each primitive's policy. In lieu of waypointing, the confidence value is calculated by increasing α_1 and α_4 while decreasing α_2 and α_3 to zero. Additionally, algorithm steps 2(d)i and 2(d)ii are not executed, as these constitute the waypoint generation process.

As with many parameterized learning methods, choosing proper values depends on a multitude of factors including, but not limited to, sensor sensitivity, agent precision of action, overlapping primitive action state density, available computational power, and memory constraints. While there is much room for optimization within the algorithm specified, we believe it serves well as a simple, straightforward proof-of-concept for partnering a generalized action recognition system with a skill representation that is a dominant standard in human-robot interaction research.

One of the most troublesome concerns within any system dealing with high dimensional MDPs is maintaining sparsity of space representation. As the initialized state space increases, the process of intelligently connecting previously unseen states becomes computationally infeasible while simultaneously maintaining the responsiveness required for real-world systems. Within the context of our algorithm, it is suggested that a garbage collection mechanism be implemented that prunes low-reward states, prioritizing the removal of those with high connectivity.

As one of our goals is to use observation to optimize the inverse problem of execution, removing entrances to bad paths as a pre-processing step can dramatically reduce an agent's search space. Knox, et. al.'s TAMER framework has begun to address this problem by using a human in the loop as an intelligent reinforcement signal (Knox and Stone 2008). Since the intention of an observer is not as easily dis-

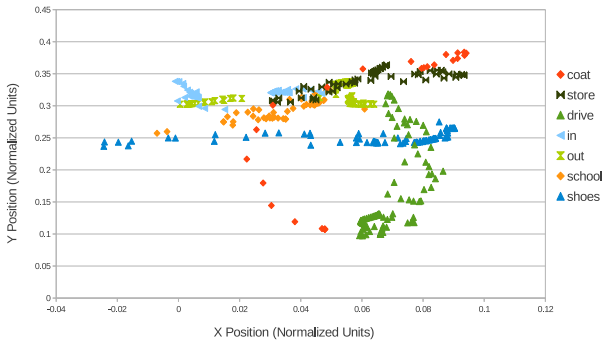

Figure 1: Overlapping right hand positions in ASL signs. This plot represents a 2D projection of 3 representative dimensions in our problem space.

cernible as that of an actor, such direct human-as-reward-signal methods are not as accessible without the introduction of an interface providing insight into the structure of each skill's internal representation as SMDP.

Experimental Setup

We chose to test our system on properly identifying and segmenting video of American Sign Language (ASL) gestures. American Sign Language provides a practical use case while testing the algorithm's tolerance to variance in training, as each training example will necessarily be unique. Our state space encompasses six dimensions, corresponding to the relative x, y, and z coordinates of each hand involved. Our system has limited *a priori* knowledge of a partially overlapping set of ASL gestures, having been given a single exemplar of training data for each known gesture. We gathered our data using a commercially-available Microsoft Kinect sensor.

We see ASL production and recognition as a complex domain containing challenges representative of those faced in collaborative robotics. Additionally, ASL requires, at a minimum, a six-dimensional state space representation. This provides sufficient complexity to prove that our skill representation can be used in environments where brute force and other naïve methods would be infeasible. Furthermore, the noise associated with human performance of the ASL symbols is large enough to ensure that our recognition techniques rely on the overall trajectory of the training data, and not merely frame-by-frame matching of exact states over time. Finally, the ASL gestures chosen overlap significantly, avoiding segmentation based on the region of the state space that a primitive action occupies. Although there are six dimensions of the state space in total, Figure 1 displays a representative sampling of the noticeable overlap along two of those dimensions.

Gesture recognition for ASL symbols is not novel. Our attempt here is not to recognize gestures with an extremely high degree of fidelity. Although an interesting problem in its own right, several researchers have already made formidable contributions in this arena (Freeman et al. 1994; Natarajan and Nevatia 2007; Darrell and Pentland 1996; Munib et al. 2007; Liang and Ouhyoung 1998; Charayaphan and Marble 1992). We use our system to perform frame-by-frame classification indicating which gesture is likely being performed, while simultaneously improving the accuracy of an internal policy representing that respective gesture.

Results

To show the effectiveness and viability of RAPTOR, we tested its ability to classify a set of seven American Sign Language gestures: coat, drive, in, out, school, shoes, and store. Due to the inherent similarity of movement within our gestures and low fidelity of our input signals, the chosen gestures had significant overlap in six-dimensional space. In particular, the gestures for 'shoes', 'school', and 'in' are extremely similar after being converted into our six dimensional reference frame. In all cases we trained our agent using a single, expert-provided example of unaltered, Kinect-recorded data consisting of centroid coordinates (x,y,z) for each hand in a frame of reference centered on the location of the demonstrator's head. We presented our system with sequences of naturally captured, noised, and lossy data.

As expected, our algorithm performed without error when tested on its training data. As increasingly lengthy gesture compositions were provided, each observed gesture became increasingly robust to both noisy data and outliers. We generate noisy data by multiplying each value within the incoming state descriptor vector by a randomly selected value $r \in [1-n/2, 1+n/2]$, where n designates the severity of the noise. We simulate lossy data by replacing large contiguous blocks of valid frames, approximately 10-30% of total sensor input, with a single repeated stale input frame. This simulates the failure of the sensor being polled by RAPTOR. The algorithm's mechanism for estimating the proper connection of previously unseen states successfully accommodated for the noise and absent states in the test data.

The complex scenario shown in Figure 2 and Figure 3 included the placement of some of the most similar gestures in close temporal proximity. We introduced additional noisy data, composited randomly from familiar states in each of the seven gestures' training data, to demonstrate our algorithm's ability to discern when there is an unrecognized action taking place. The introduction of waypointing, shown in Figure 3, boosted overall confidence values for correct classifications and smoothed the transitions between non-confident and confident classifications. One particular example of this can be seen near frame 410, as without waypointing the gesture for 'school' is confused with the end of the 'shoes' gesture preceding it.

The lack of classification of the final 'shoes' gesture is explained by the step in which after a high confidence classification is made, the window through which the algorithm searches for effective state space traversals, P_{hit}, is cut in half to prevent harmful self-reinforcement looping.

The successful application of our algorithm to this representative example of non-trivial action classification serves as a proof-of-concept that can be applied to benefit any generic hierarchical learning framework.

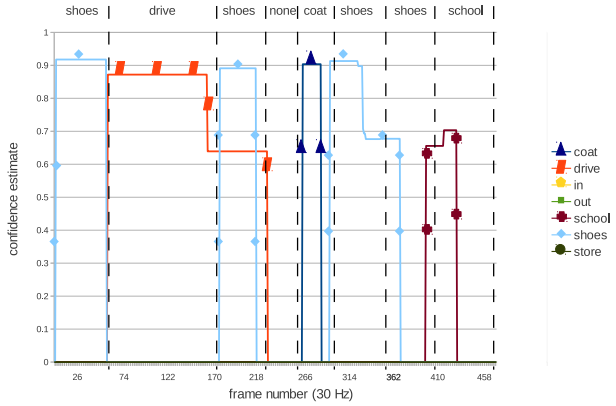

Figure 2: Primitive sequence classification without waypointing, relying solely on observed data and $\epsilon-$neighborhood transition approximations.

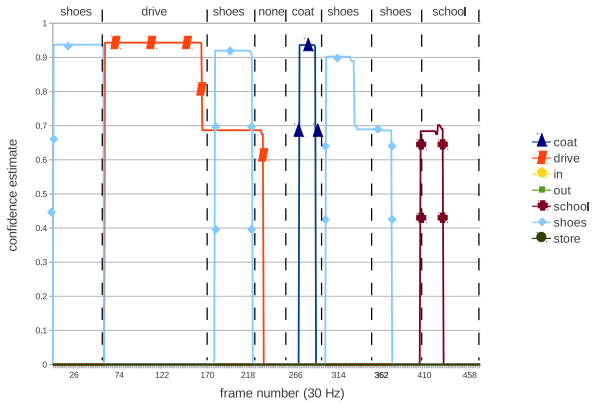

Figure 3: Primitive sequence classification with waypointing enabled. Utilizing the hybrid policy allows for higher confidence as well as fewer misclassifications.

Related and Future Work

It is important to differentiate our contribution from that of existing research in learning from demonstration. Much previous research within reinforcement learning has focused on using examples to transfer a policy representing some basic skill from teacher to pupil (e.g., Grollman and Jenkins 2008; Argall et al. 2009). Our algorithm focuses on the use of *existing knowledge* to generate an approximate decomposition of complex scene data while simultaneously using our classifications and observation to continually shape actionable policies.

While work partially overlapping our goals has been published previously (Nicolescu et al. 2008), our work differs substantially in terms of both our algorithm and the resulting simultaneous benefit to observation classification and action execution. By categorizing observations into a timeline, a robotic agent may also build a hierarchy by applying subsequence recognition to the algorithm's output. This construction provides a concise, communicable plan for any observed task. The problem of hierarchy assignment is left for other methods to more fully explore.

Furthermore, it would be greatly advantageous to study a deployed system with our algorithm in a complex, collaborative, multi-human multi-agent environment. Given a limited set of observation data from a factory floor, a robotic system using our algorithm should be able to decompose the scene into several agents performing several primitives simultaneously. Once deployed in the actual factory, the robot should have the capability to perform actions it has both learned in direct cooperation with humans as well as those actions trained through observation.

Conclusions

We have presented herein a method for action recognition that simultaneously leverages an existing representation of primitive skills and improves the accuracy with which those skills can be performed by the observer. These skills are modeled as traditional SMDPs, an existing standard for skill representation in reinforcement learning. This real-time recognition of learned actions is essential for any co-robot to be able to cooperate effectively in a multi-human multi-robot environment. In addition, the generalized recognition algorithm greatly facilitates the construction of a skill hierarchy that can represent a complex cooperative task structure. Utilizing the generated skill hierarchy, a robotic agent can simultaneously execute desired tasks and increase its cooperative potential by recognizing known primitive actions being performed by co-workers.

Our system can be easily qualified as an extension to what Ng and Abeel (2000) originally termed "inverse reinforcement learning". By opportunistically incorporating observed data as propagated rewards to the SMDP skill representation, our system converges on a stable learned primitive with fewer explicit trials. Furthermore, the bonus rewards assigned to those observations that achieve high confidence, in concert with waypointing, allow us to use specific portions of noisy, lossy, and otherwise unreliable observed data as expert training.

Other works have offered solutions to many of the challenges standing in the way of mainstreaming collaborative robotics, including automating option scaffolding through demonstration (Konidaris et al. 2011), acquiring skills utilizing human reward signals in place of (Knox and Stone 2008) or in addition to (Thomaz and Breazeal 2006) a traditional objective function, and correcting for critical omissions in non-expert demonstrations (Breazeal et al. 2006). Simplifying the human trainer/robot student interaction paradigm by shaping human guidance into usable forms is also rapidly advancing (Knox and Stone 2008; Kaplan et al. 2002; Maclin, Shavlik, and Kaelbling 1996; Thomaz and Breazeal 2006; Taylor, Suay, and Chernova 2011). These successful results strongly contribute to the widespread use of Q-Learning variants in real robotic systems and, more generally, to the feasibility of natural human-robot collaboration.

References

Argall, B. D.; Chernova, S.; Veloso, M.; and Browning, B. 2009. A survey of robot learning from demonstration. *Robotics and Autonomous Systems* 57(5):469 – 483.

Bluethmann, W.; Ambrose, R.; Diftler, M.; Askew, S.; Huber, E.; Goza, M.; Rehnmark, F.; Lovchik, C.; and Magruder, D. 2003. Robonaut: A robot designed to work with humans in space. *Autonomous Robots* 14:179–197. 10.1023/A:1022231703061.

Breazeal, C.; Berlin, M.; Brooks, A. G.; Gray, J.; and Thomaz, A. L. 2006. Using perspective taking to learn from ambiguous demonstrations. *Robotics and Autonomous Systems* 54(5):385–393.

Charayaphan, C., and Marble, A. 1992. Image processing system for interpreting motion in american sign language. *Journal of Biomedical Engineering* 14(5):419 – 425.

Darrell, T., and Pentland, A. 1996. Active gesture recognition using partially observable markov decision processes. In *Pattern Recognition, 1996., Proceedings of the 13th International Conference on*, volume 3, 984 –988 vol.3.

Digney, B. 1998. *Learning Hierarchical Control Structures for Multiple Tasks and Changing Environments*. MIT Press. 321–330.

Freeman, W. T.; Freeman, W. T.; Roth, M.; and Roth, M. 1994. Orientation histograms for hand gesture recognition. In *In International Workshop on Automatic Face and Gesture Recognition*, 296–301.

Grollman, D., and Jenkins, O. 2008. Sparse incremental learning for interactive robot control policy estimation. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, 3315 –3320.

Kaplan, F.; Oudeyer, P.-Y.; Kubinyi, E.; and Miklsi, . 2002. Robotic clicker training. *Robotics and Autonomous Systems* 38(3-4):197–206.

Knox, W., and Stone, P. 2008. Tamer: Training an agent manually via evaluative reinforcement. In *Development and Learning, 2008. ICDL 2008. 7th IEEE International Conference on*, 292 –297.

Knox, W. B., and Stone, P. 2011. Understanding human teaching modalities in reinforcement learning environments: A preliminary report. In *IJCAI 2011 Workshop on Agents Learning Interactively from Human Teachers (ALIHT)*.

Konidaris, G., and Barto, A. 2009. Efficient skill learning using abstraction selection. In *In Proceedings of the 21st International Joint Conference on Artificial Intelligence*.

Konidaris, G.; Kuindersma, S.; Grupen, R. A.; and Barto, A. G. 2011. Autonomous skill acquisition on a mobile manipulator. In Burgard, W., and Roth, D., eds., *AAAI*. AAAI Press.

Liang, R.-H., and Ouhyoung, M. 1998. A real-time continuous gesture recognition system for sign language. In *Automatic Face and Gesture Recognition, 1998. Proceedings. Third IEEE International Conference on*, 558 –567.

Maclin, R.; Shavlik, J. W.; and Kaelbling, P. 1996. Creating advice-taking reinforcement learners. In *Machine Learning*, 251–281.

Martinson, E.; Stoytchev, A.; and Arkin, R. 2002. Robot behavioral selection using qlearning. *IEEERSJ International Conference on Intelligent Robots and System* 1(October):970–977.

Mugan, J., and Kuipers, B. 2009. Autonomously learning an action hierarchy using a learned qualitative state representation. In *In Proceedings of the 21st International Joint Conference on Artificial Intelligence*.

Munib, Q.; Habeeb, M.; Takruri, B.; and Almalik, H. 2007. American sign language (asl) recognition based on hough transform and neural networks. *Expert Systems with Applications* 32(1):24–37.

Natarajan, P., and Nevatia, R. 2007. Coupled hidden semi markov models for activity recognition. In *Motion and Video Computing, 2007. WMVC '07. IEEE Workshop on*, 10.

Ng, A. Y., and Russell, S. 2000. *Algorithms for inverse reinforcement learning*. Morgan Kaufmann Publishers Inc. 663670.

Nicolescu, M.; Jenkins, O.; Olenderski, A.; and Fritzinger, E. 2008. Learning behavior fusion from demonstration. *Interaction Studies* 9(2):319–352.

Sutton, R.; Precup, D.; and Singh, S. 1999. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112:181–211.

Taylor, M. E.; Suay, H. B.; and Chernova, S. 2011. *Integrating Reinforcement Learning with Human Demonstrations of Varying Ability*. AAMAS. 617–624.

Thomaz, A. L., and Breazeal, C. 2006. Reinforcement learning with human teachers: evidence of feedback and guidance with implications for learning performance. In *Proceedings of the 21st national conference on Artificial intelligence - Volume 1*, AAAI'06, 1000–1005. AAAI Press.

Trafton, J. G.; Cassimatis, N. L.; Bugajska, M. D.; Brock, D. P.; Mintz, F. E.; and Schultz, A. C. 2005. Enabling effective human-robot interaction using perspective-taking in robots. *IEEE Transactions on Systems, Man, and Cybernetics* 35:460–470.

Watkins, C. 1989. *Learning from delayed rewards*. Ph.D. Dissertation, Kings College.