MIT Media Lab Perceptual Computing Technical Report No. 367;
Submitted, Intl. Conf. Machine Learning ML96

# Multiple-model Q-learning for Active Recognition Tasks

Trevor Darrell
MIT Media Lab
trevor@media.mit.edu

January 28, 1996

**Abstract**

When applying Q-learning or related methods for hidden-state reinforcement learning to a task involving multiple goals, such as recognition of multiple targets, the complexity of the utility-value space often becomes intractable as the number of goals increases. We propose a solution to this problem which utilizes a separate learning module for each goal, each with a distinct utility and reward value but with shared observations and actions. A Q-learning algorithm runs separately in each module, but only a single action is chosen. The action is selected from the module with highest expected reward, i.e. utility value. This coupled Q-learning algorithm can be expressed concisely as Q-learning with vector-valued utility and reward values. We apply our method in the domain of human-computer interaction, using an active recognition paradigm which assumes hidden state. In this domain the goal is successful recognition, and the system is rewarded when it signals the target is present. Empirically, we have found instance-based reinforcement learning can find optimal policies for our task given a single target, and given multiple targets when using the multiple model approach.

## 1 Introduction

In most tasks, purely passive approaches to visual recognition are impractical in unconstrained domains. To overcome the limitations of a single camera view, vision must include an *active* component [2, 4, 3]. In our research group, we are interested in developing systems which allow a user to naturally interact with an intelligent room or video wall (Figure 1); to do this requires visual recognition of body pose and gestures. However, a single camera view is of insufficient resolution to observe both the entire scene and the details needed for recognition. We have thus turned to models of action-selection for visual attention, to guide an active camera which can acquire high-resolution, foveated images. Inspired by the success of statistical methods for hidden state learning in the domain of static perception (e.g., Hidden Markov Models), for active tasks we use a hidden state learning model with both action and perception: the Partially Observable Markov Decision Process.

As we will describe in the following sections, we have formulated an active gesture recognition task using a POMDP framework, and have found the instance-based approach [14] to be a feasible method for finding a foveation policy for this task. We define a special action to signify recognition, which leads to a Q-value space that is interpretable as confidence that the target is present.

However, Q-learning fails on our task when we introduce multiple targets, as the Q-value space becomes quite complex. To solve the task under these conditions, we propose a multiple model approach, where a separate Q-learning module attempts to generate foveation actions which are appropriate under the assumption that its target is present. As we will show, this method is empirically viable, and can expressed concisely as a Q-learning system with vector-valued Q and reward functions.

First, we will briefly overview our domain and the visual routines which extract the features used in the state representation. Next, we will define the active gesture recognition task using the POMDP framework and discuss the exploration strategy we use for quick convergence. Finally, we will present our multiple model Q-learning algorithm, followed by empirical results.

## 2 Visual Attention for Recognition

Previously, we have implemented vision routines to track a user in an office setting [7, 19]. Our system can track people and identify head/hand locations as they walk about a room, using little assumed prior knowledge of the person and operating on coarse-scale images from a static camera (Figure 2). We have also previously implemented a system for real-time expression matching/tracking, using sets of view-based correlation models to represent spatio-temporal gesture patterns. We applied our model to the problem of hand gesture recognition [6] as well as for tracking facial expressions [8]. However, this expression analysis requires detailed imagery, which cannot be obtained from a single, fixed camera as the user walks about a room. To provide high resolution images we augment the wide field-of-view camera with an active, narrow-field-of-view camera. Information about head/hand location from the person tracking routines is used to drive the motor control parameters of the narrow field camera; currently the camera can be directed to autonomously track head or hands (see Figure 3). Recognition of gestures using this active camera is possible in real-time; for details see [9]. Collectively, these methods provide the feature-based person state

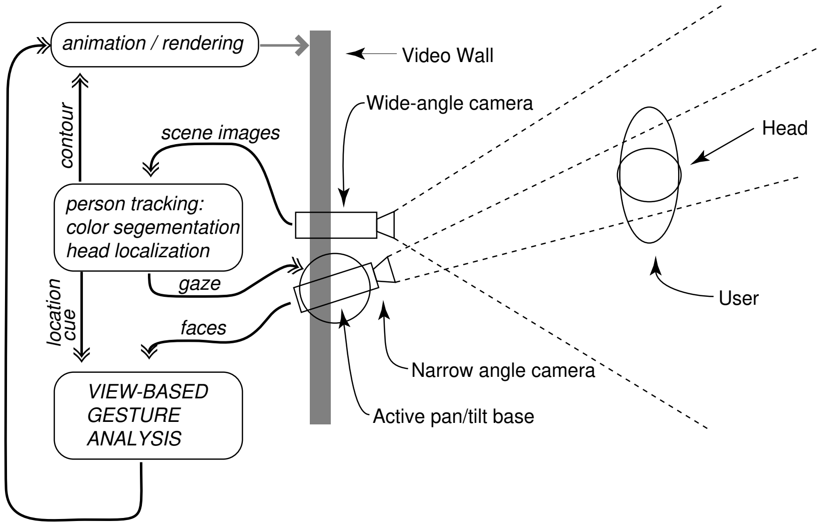



Figure 1: Overview of system combining person tracking and view-based gesture recognition. Static, wide-field-of-view, camera localizes user's head or hands, which drives gaze control of active narrow-field-of-view camera. Foveated images are used for view-based gesture analysis and recognition. With this framework animations and objects can be rendered on video wall and react to user's position, pose, and gestures.

described in Table 1.

## 2.1 The Active Gesture Recognition Problem

We define an Active Gesture Recognition (AGR) task as follows. First, we assume primitive routines exist to provide the continuous valued control and tracking of the different body parts that perform gestures. Second, we assume that body pose and hand/face state is represented as a feature set, based on the representation produced by our body tracker and view-based recognition system, and we define a gesture to be a configuration of the user's body pose and hand/face expression. Third, we assume that, in addition to there being actions for foveating all the relevant body parts, there is also a special action labeled `accept`, and that the execution of this action by the AGR system signifies detection of the gesture. Finally, the goal of the AGR task is to execute the `accept` action whenever the user is in the target gesture state, and not to perform that action when the user is in any other (e.g. distractor) state. The AGR system should use the foveation actions to optimally discriminate the target pattern from distractor patterns, even when no single high-resolution

(a) (b) (c)

Figure 2: Tracking of a user in an interactive room environment. (a) color image of user in scene. (b) segmentation result. (c) estimated part structure, including head and hand locations. From [19].

(a) (b)

Figure 3: Images acquired from wide (a) and narrow (b) field of view cameras as user moves across scene and narrow camera tracks head. From [9].

image of the user is sufficient to decide what gesture the user is performing.

An important problem in applying reinforcement learning to this task is that our perceptual observations may not provide a complete description of the user's state. Indeed, because we have a foveated image sensor we know that the user's true gestural state will be hidden whenever the user is performing a gesture and the camera is not foveated on the appropriate body part. By definition, a system for perceptual action-selection must not assume a full observation of state is available, otherwise there would be no meaningful perception taking place.

The AGR task can be considered as a Partially Observable Markov Decision Process (POMDP), which is essentially a Markov Decision Process without direct access to state [15, 13]. A POMDP consists of a set of states in the world $\mathcal{S}$, a set of observations $\mathcal{O}$, a set of actions $\mathcal{A}$, and a reward function $R$. After executing an action $a$, the likelihood of transitioning between two states $s, s'$

is given by $T(s, a, a')$, an observation $o$ is generated with probability $O(s, a, o)$. In practice, $T$ and $O$ are not easily obtainable, and we use methods which do not require them *a priori*. Rather than attempt to solve the POMDP explicitly, we look to techniques for hidden state reinforcement learning to find a solution [14, 12, 10, 1].

Our state is defined by the users pose, facial expression, and hand configurations, expressed in nine variables (see Table 1). Three of these are boolean, `person-present`, `left-arm-extended`, and `right-arm-extended`, and are provided directly by the person tracker. Three more are provided by the foveated gesture recognition system, `face, left-hand, right-hand`, and take on an integer number of values according to the number of view-based expressions/hand-poses: in our first experiments `face` can be one of `neutral`, `smile`, or `surprise`, and the hands can each be one of `neutral`, `point`, or `open`. In addition, three boolean features represent the internal state of the vision system: `head-foveated`, `left-hand-foveated`, `right-hand-foveated`.

At each time step, the world is defined by a state $s \in \mathcal{S}$, which is defined by these features. An observation, $o \in \mathcal{O}$, consists of the same feature variables, except that those provided by the foveated gesture system (e.g., head and hands) are only observable when foveated. Thus the `face` variable is hidden unless the `head-foveated` variable is set, the `left-hand` variable hidden unless the `left-hand-foveated` variable set, and similarly with the right hand. Hidden variables are set to a `undefined` value.

The set of actions, $\mathcal{A}$, available to the AGR system are 4 foveation commands: `look-body`, `look-head`, `look-left-hand`, and `look-right-hand` plus the special `accept` action. Each foveation command causes the active camera to follow the respective body part, and sets the internal foveation feature bits accordingly.

The reward function provides a unit positive reward whenever the `accept` action is performed and the user is in the target state (as defined by an oracle, external to the AGR system), and a fixed negative reward of magnitude $\alpha$ when performed and the user is in a distractor (non-target) state. (Following [18] we disproportionally penalize the Q-value of perceptually aliased states.) Zero reward is given whenever a foveation action is performed.

We wish to find a *policy*, a mapping from state (in the case of an MDP) or some function on observations (in the case of a POMDP) to action which maximizes the expected future reward, suitably discounted to bias towards timely performance. Given the reward function in the AGR task, this will correspond to a policy which successfully recognizes the target gesture.

## 2.2 Hidden-State Reinforcement Learning

To find policies for AGR tasks we have implemented an instance-based method for hidden state reinforcement learning, based on earlier work by McCallum [14]. This method performs Q-learning [16, 17], but replaces the absolute state with

| *feature* | *values* | *observability precondition* |
|---|---|---|
| `person-present` | (true, false) | (always observable) |
| `left-arm-extended` | (true, false) | (always observable) |
| `right-arm-extended` | (true, false) | (always observable) |
| `face-foveated` | (true, false) | (always observable) |
| `left-hand-foveated` | (true, false) | (always observable) |
| `right-hand-foveated` | (true, false) | (always observable) |
| `face` | (neutral, smile, surprise, ...) | `face-foveated` == true |
| `left-hand` | (neutral, point, open, ...) | `left-hand-foveated` == true |
| `right-hand` | (neutral, point, open, ...) | `right-hand-foveated` == true |

Table 1: Set of features used in POMDP formulation of Active Gesture Recognition task. This representation is computable in real-time using person tracking and (user-specific) gesture recognition routines.

a distributed memory-based state representation. Given a history of action, reward, and observation tuples, $(a[t], r[t], o[t])$, $0 \leq t \leq T$, a Q-value is also stored with each time step, $q[t]$, and Q-learning is performed by evaluating the similarity of recently observed tuples with sequences farther back in the history chain. Q-values are computed, and the Q-learning update rule applied, maintaining this distributed, memory-based representation of Q-values.

As in traditional Q-learning, at each time step the utility of each action in the current state is evaluated. If full access to the state was available and a table used to represent Q values, this would simply be a table look-up operation, but in a POMDP we do not have full access to state. Using a variation on the instance-based approach employed by McCallum's Nearest Sequence Memory (NSM) algorithm, we instead find the $K$ nearest neighbors in the history list relative to the current time point, and compute their average Q value. For each element on the history list, we compute the sequence match criteria with the current time point, $M(i,T)$, where

$$M(i,j) = S(i,j) + M(i-1, j-1) \quad \text{if } S(i,j) > 0 \text{ and } i > 0 \text{ and } j > 0$$

$$0 \quad \text{otherwise} \; .$$

We define $S(i,j)$ to be 1 if $o[i] = o[j]$ or $a[i] = a[j]$, 2 if both are equal, and 0 otherwise. Using a superscript in parentheses to denote the action index of a Q-value, we then compute

$$Q^{(a)}[T] = (1/K) \sum_{i=0}^{T} v^{(a)}[i] q[t] \; , \tag{1}$$

where $v^{(a^*)}[i]$ indicates whether the history tuple at time step $i$ votes when computing the Q-value of a new action $a^*$: $v^{(a^*)}[i]$ is set to 1 when $a[i] = a^*$

and $M(i-1,T)$ is among the $K$ largest match values for all $k$ which have $a[k]=a^*$, otherwise it is set to 0. Given Q values for each action the optimal policy is simply

$$\pi[T] = \arg\max_{a \in \mathcal{A}} Q^{(a)}[T] \ . \tag{2}$$

The new action $a[T+1]$ is chosen either according to this policy or based on an exploration strategy. In either case, the action is executed yielding an observation and reward, and a new tuple added to the history. The new Q-value is set to be the Q value of the chosen action, $q[T+1] = Q^{(a[T+1])}[T]$. The update step of Q learning is then computed, evaluating

$$U[T+1] = \max_{a \in \mathcal{A}} Q^{(a)}[T+1] \ , \tag{3}$$

$$q[i] \leftarrow (1-\beta)q[i] + \beta(r[i] + \gamma U[T+1]) \ , \tag{4}$$

for each $i$ such that $v^{(a[T+1])}[i] = 1$.

### 2.3 Deterministic Exploration Strategies

Instance-based Q-learning can find optimal policies to solve the AGR task; however, varying amounts of experience are required for the system to converge to an adequate level of performance. Furthermore, the system can become stuck in local minima if it follows a purely greedy strategy (e.g. no exploration). To learn the globally optimal policy the system must experience the global optimal path at least once; this may never happen without exploration, and may take a considerable amount of time with purely stochastic exploration.

Since there is a great deal of structure in the AGR task, we have devised deterministic exploration regimes which provide quick convergence to an optimal policy, given random presentation of target and distractors. The key insight is that the system needs to build an accurate evaluation of the utility (Q-value) of the `accept` action given different foveated observations. Depending on the complexity of the recognition task, which is determined by the target and distractor set, some number of foveated observations is ideally needed to resolve the target identity.

We define the complexity of an AGR problem to be the smallest number of foveated observations sufficient to uniquely identify the target gesture from the distractor patterns. A 0-th order AGR problem is one in which the target is uniquely identifiable without any foveated observations; a 1-st order problem requires a single foveated observation, say a facial expression; a 2-nd order problem requires foveating on two body parts, etc.[1]

[1] For example, to discriminate gesture 6 from gesture 7,8 in the example shown in Table 2 requires a foveated observation of both hands, so this is a second-order AGR task. Note that we only use first-order exploration in this example, so some of the learning is done stochastically.

In a 0-th order problem, a sufficient (and trivial) deterministic exploration regime is to execute the `accept` action under all observation conditions and compute Q-values as given above. In a higher order problem, a single positive reward is insufficient to determine policy, since the system may well be perceptually aliased. In this case we need to execute the `accept` action enough times such that consistent positive reward under the observation condition is unlikely to be accidentally obtained. To reliably interpret a positive reward as successful recognition we must counter the assumption that there is one or more distractor which is perceptually aliased with the target. To do this, we can either deterministically cycle through each observation (if we control the world state during the exploration period), or wait a length of time (if the world is stochastic) to guarantee that each state that could generate the given observation will be experienced.

In AGR problems of non-zero order a number of actions must be executed to foveate the appropriate body parts before the target can be discriminated reliably. For the system to be guaranteed to learn this, it must execute each possible foveation action (or foveation sequence for problems of 2-nd and higher order) before assessing the reliability of the accept command. Our deterministic exploration strategy is thus to exhaustively explore this foveation/accept space assuming a problem of fixed order.

Practically, in our current experiments we use a first order deterministic exploration strategy with $K * M$ accept trials for each action, where $M$ is the number of gestures (target plus distractors) and $K$ is the number of nearest neighbors used in the instance-based matching. For each foveation action, the system will execute that action and then execute `accept` to obtain a positive or negative reward, $K * M$ times. After this period of experience given by deterministic exploration, we let the system run with its optimal policy choice mediated by a fixed random exploration factor. With probability $p$, the optimal policy is disregarded and an action chosen with uniform probability.

# 3 Multiple Model Q-Learning

In general, we have found the simple, instance-based hidden state reinforcement learning described above to be an effective way to perform action-selection for foveation when the task is recognition of a single object from a set of distractors. However, we did not find that this type of system performed well when the AGR task was extended to include more than one target gesture. When multiple `accept` actions were added to enumerate the different targets, we were not able to find exploration strategies that would converge in reasonable time.

This is not unexpected, since the addition of multiple causes of positive reward makes the Q-value space considerably more complex. To remedy this problem, we propose a multiple model Q-learning system. In a multiple model approach to the AGR problem, separate learning agents model the task from

Figure 4: Four gesture patterns which require foveated images for discrimination. Top row shows wide field of view camera; bottom row shows active camera. Active camera tracks the relevant hand to identify the gesture, as described in text.

each target's perspective. Conceptually, a separate Q-learning agent exists for each target, maintains it's own Q-value and history structure, and is coupled to the other agents via shared observations. Since we can interpret the Q-value of an individual AGR agent as a confidence value that its target is present, we can mediate among the actions predicted by the different agents by selecting the action from the agent with highest Q-value (Figure 5(a)).

Formally, in our multiple model Q-learning system all agents share the same observation and selected action, but have different reward and Q-values. Thus they can be considered a single Q-learning system, but with vector reward and Q-values. Our multiple model learning system is thus obtained by rewriting Eqs. (1)-(4) with vector $q[t]$ and $r[t]$. Using a subscript $j$ to indicate the target index, we have

$$Q_j^{(a)}[T] = (1/K) \sum_{i=0}^{T} v^{(a)}[i] q_j[t] \ , \quad \pi[T] = \arg\max_{a \in \mathcal{A}} \left( \max_j Q_j^{(a)}[T] \right) \ . \quad (5)$$

Rewards are computed with: if $a[T] =$ `accept` then $r_j[T] = R(j,T)$ else $r_j[T] = 0$; $R(j,T) = 1$ if gesture $j$ was present at time $T$, else $R(j,T) = -\alpha$. Further,

$$U_j[T+1] = \max_{a \in \mathcal{A}} Q_j^{(a)}[T+1] \ , \quad (6)$$

$$q_j[i] \leftarrow (1-\beta) q_j[i] + \beta(r_j[i] + \gamma U_j[T+1]) \quad \forall \ i \text{ s.t. } v^{(a[T+1])}[i] = 1 \ . \quad (7)$$

Note that our sequence match criteria, unlike that in [14], does not depend on $r[t]$; this allows considerable computational savings in the multiple model system since $v^{(a)}$ need not depend on $j$.

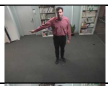
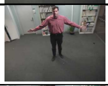
| gesture | scene | features |
|---|---|---|
| 0 |  | `person-present`, `left-arm-extended`, `left-hand-point` |
| 1 |  | `person-present`, `left-arm-extended`, `left-hand-open` |
| 2 |  | `person-present`, `right-arm-extended`, `right-hand-point` |
| 3 |  | `person-present`, `right-arm-extended`, `right-hand-open` |
| 4 |  | `person-present`, `face-smile` |
| 5 |  | `person-present`, `face-surprise` |
| 6 |  | `person-present`, `left-arm-extended`, `left-hand-open`, `right-arm-extended`, `right-hand-open` |
| 7 |  | `person-present`, `left-arm-extended`, `left-hand-open`, `right-arm-extended`, `right-hand-point` |
| 8 |  | `person-present`, `left-arm-extended`, `left-hand-point`, `right-arm-extended`, `right-hand-open` |

Table 2: The set of targets used in active gesture recognition example.

# 4 Performance

We applied our multiple model learning system on the AGR task using the types of gestures shown in Figure 4, where the body and arm features are observable in the coarse view (i.e. fully observable) but the hand and face features require a high-resolution foveated view. The full set of gesture targets we used in this experiment is shown in Table 2. We kept constant the parameters $\beta = 0.5$, $\gamma = 0.5$, $\alpha = 10$. Results summed over 2500 trials are shown in Figure 5(b), with classification error plotted against the number of nearest neighbors used in the NSM algorithm. The error rate shown is after convergence; we ran the algorithm with a period of first-order deterministic exploration before following the optimal policy with random exploration probability $p = 0.01$. As can be seen from the graph, for any non-degenerate value of $K$ reasonable performance was obtained on this task; for $K > 2$, the system performed almost perfectly.

In our present implementation, real-time performance ($> 5$Hz) on the AGR task is possible as long as the history list representation can be maintained in main memory on a contemporary workstation. (This performance is due to the

(a)

(b)

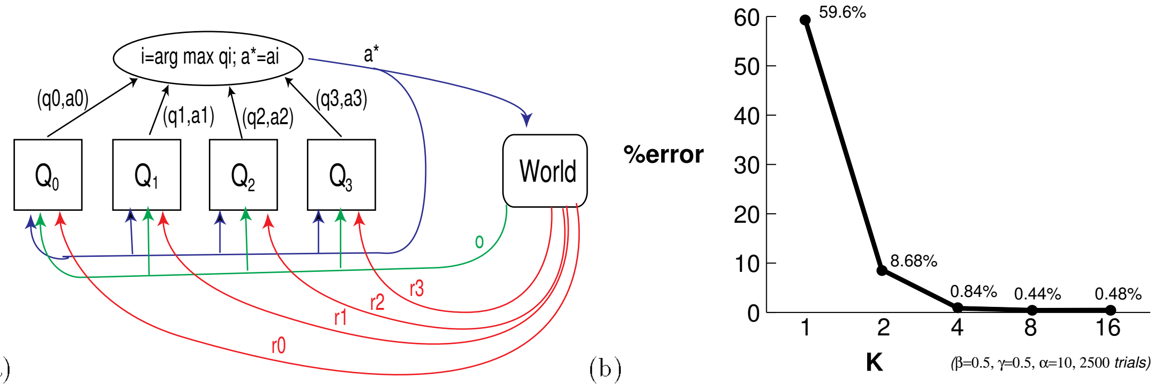



Figure 5: (a) multiple model Q-learning: one Q-learning agent for each target gesture to be recognized, with coupled observation and action but separate reward and Q-value. (b) results on recognition task with multiple gesture targets; graph shows error rate after convergence plotted as a function of number of nearest neighbors used in learning algorithm.

efficiency of the feature-based representation, which itself is computable in real-time [6, 8].) Currently, approximately ten thousand time steps can be stored in main memory, so real-time performance is possible for runs of up to 30 minutes. We are extending the system to selectively forget and/or merge portions of the history list, so that the system can run indefinitely in constant memory.

## 5 Conclusion

Gesture recognition systems often require high-resolution images of hands and faces when used in unconstrained environments. We have developed a mechanism to track and foveate salient body parts in an active gesture recognition task. We adopt a Partially Observable Markov Decision Process formalism, using an action set comprised of foveation actions as well as a special `accept` action. Execution of `accept` is rewarded based on whether the target is in the scene. We use instance-based hidden state reinforcement learning to learn a policy which models both when to execute the `accept` action and what foveation commands to execute to properly discriminate the target gesture. However, multiple targets create undesired complexity in a scalar Q-value space, so we define a multiple-model Q-learning paradigm with vector-valued Q and reward functions. With this, our system can actively recognize targets from a set of gestures in real-time.

## References


[1] A. Cassandra, L. P. Kaelbling, and M. Littman. Acting optimally in partially observable stochastic domains. In *Proc. AAAI-94*, pages 1023–1028. Morgan Kaufmann, 1994.

[2] J. Y. Aloimonos, I. Weiss, and A. Bandopadhay, Active Vision, *Intl. Journal of Computer Vision*, 1(3):333-356, 1987.

[3] R. Bajcsy, Active Perception, *Proceedings of the IEEE*, 76(8):996-1005, August 1988

[4] D. H. Ballard, Reference Frames for Animate Vision, *in Proc. 11th IJCAI*, pp. 1635-1641, August 1989.

[5] A. G. Barto, S. J. Bradtke, and S. P. Singh. Real-time learning and control using asynchronous dynamic programming. Computer Science Technical Report 91-57, University of Massachusetts, August 1991.

[6] T., Darrell, and A., Pentland, Space-Time Gestures. *Proceedings IEEE CVPR-93*, New York, IEEE Comp. Soc. Press, 1993.

[7] T. Darrell, P. Maes, B. Blumberg, and A. P. Pentland, A Novel Environment for Situated Vision and Behavior, *Proc. IEEE Workshop for Visual Behaviors*, IEEE Comp. Soc. Press, 1994

[8] T. Darrell, I. Essa, and A. P. Pentland, Correlation and Interpolation Networks for Real-time Expression Analysis/Synthesis, In *Advances in Neural Information Processing Systems 7*, MIT Press, 1995.

[9] T. Darrell and A. Pentland, A., Attention-driven Expression and Gesture Analysis in an Interactive Environment, in *Proc. Intl. Workshop on Automatic Face and Gesture Recognition (IWAFGR '95)*, Zurich, Switzerland, 1995.

[10] T. Jaakkola, S. Singh, and M. Jordan. Reinforcement Learning Algorithm for Partially Observable Markov Decision Problems. In *Advances In Neural Information Processing Systems 7*, MIT Press, 1995.

[11] T. Poggio and F. Girosi, A Theory of Networks for Approximation and Learning. MIT AI Lab TR-1140, 1989.

[12] L. Lin and T. Michell. Reinforcement learning with hidden states. In *Proc. AAAI-92.* Morgan Kaufmann, 1992.

[13] W. Lovejoy. A survey of algorithmic methods of partially observed markov decision processes. *Annals of Operation Reserach*, 28:47–66, 1991.

[14] R. A. McCallum. Instance-based State Identification for Reinforcement Learning. In *Advances In Neural Information Processing Systems 7*, MIT Press, 1995.

[15] E. J. Sondik. The optimal control of partially observable markov processes over the infinite horizon: Discounted costs. *Operations Reserach*, 26(2):282–304, 1978.

[16] R. S. Sutton. Learning to predict by the method of temporal differences. *Machine Learning*, 3:9–44, 1988.

[17] C. Watkins and P. Dayan. Q-learning. *Machine Learning*, 8:279–292, 1992.

[18] S. D. Whitehead. Active perception and reinforcement learning. *In Proc. 7th Intl. Conf. ML*, June 1990.

[19] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, Pfinder: Real-Time Tracking of the Human Body, Media Lab Per. Comp. TR-353, 1994