Berthouze, L., Kaplan, F., Kozima, H., Yano, H., Konczak, J., Metta, G., Nadel, J., Sandini, G., Stojanov, G. and Balkenius, C. (Eds.)
Proceedings of the Fifth International Workshop on Epigenetic Robotics: Modeling Cognitive Development in Robotic Systems
Lund University Cognitive Studies, 123. ISBN 91-974741-4-2

Covert Perceptual Capability Development

Xiao Huang and Juyang Weng
Computer Science and Engineering Department
Michigan State University
East Lansing, MI, 48824

Abstract

In this paper, we propose a model to develop robots' covert perceptual capability using reinforcement learning. Covert perceptual behavior is treated as action selected by a motivational system. We apply this model to vision-based navigation. The goal is to enable a robot to learn road boundary type. Instead of dealing with problems in controlled environments with a low-dimensional state space, we test the model on images captured in non-stationary environments. Incremental Hierarchical Discriminant Regression is used to generate states on the fly. Its coarse-to-fine tree structure guarantees real-time retrieval in high-dimensional state space. K Nearest-Neighbor strategy is adopted to further reduce training time complexity.

1. Introduction

If the behavior of a robot is not visible from the outside, we call it covert behavior. In this paper, we propose a covert perceptual capability development model for a robot. One application is to teach a robot to learn road boundary type for vision-based navigation. Instead of giving the the correct type explicitly, human teacher issues rewards and punishments according to the robot's guess. In this sense, the robot develops this covert capability via internal but non-enforceable probes. After this learning stage is mature, the robot can learn the correct heading direction for navigation in the same mode. Moreover, it gives the robot the ability to recover from error, which cannot be gained through supervised learning.

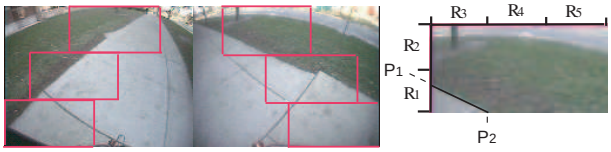
There are quite a few studies, which apply reinforcement learning to vision and attention problems. (Whitehead and Ballard, 1990)'s study is one of the earliest attempt to use reinforcement learning as a model of active perception. (Balkenius and Hulth, 2001) model attention as selection for action. They train a robot to pick out the correct focus of attention based on reinforcement learning. Experimental results are report on a simple simulator. (Minut and Mahadevan, 2001) propose a reinforcement learning model of selective visual attention. The goal is to use a fixed pan-tilt-zoom camera to find an object in a cluttered lab environment. That is, the environment is stationary. In summary we can find out that those studies test on controlled environments. The dimension of state space is low and the number of state is very limited. In order to reach the goal of real-time active vision development in non-stationary environments (outdoor navigation), we use IHDR (Incremental Hierarchical Discriminant Regression) to generate continuous state space on the fly for high dimensional visual input. Its coarse-to-fine structure guarantees online retrieval. Furthermore, we implement K Nearest Neighbor algorithm for Q-learning so that at each time instant multiple similar states can be updated, which dramatically reduces the number of rewards human teachers have to issue. We also propose a multi-layer reinforcement learning architecture (Takahashi and Asada, 2000). The first level learning modules handle covert perceptual capability development while the second level learning module deal with navigation behaviors.

In what follows, we first describe the covert capability development problem in outdoor navigation. The detailed architecture of the developmental paradigm is presented in Sec. 3. The experimental results and conclusions are reported in Sec. 4 and Sec. 5, respectively.

2. Problem Description

Our goal is to enable a robot to develop its covert perceptual capability for vision-based outdoor navigation. A good example is shown in Fig. 1 (a). Suppose the robot has two eyes (left image and right image). Instead of using the entire image as input to each first-level learning module, we divide two input images into 6 sub-windows (specified by rectangles). Suppose the size of the original image is 160×120. The dimension of the state vector is 19200, which is almost intractable. With six windows, the dimension of state vector is reduced to $80 \times 40 = 3200$. In this way, we reduce the dimension of the state space. And it turns out these six windows can cover most of road boundaries. Our goal is to teach the robot to learn the type of road boundary. Let's look at the top window on the right (Fig. 1 (b)). There is a road edge, which intersects with the window at two points (P_1, P_2). For each intersecting point, we define five

boundary types (ranges). Given p_1 as an example,there are 5 possible ranges $(R_1, R_2, R_3, R_4, R_5)$ and p_1 fits in $R1$. Suppose the image of the window is $x(t)$ (a long vector and each element corresponding to a pixel). The robot needs to learn the mapping from $x(t)$ to correct road boundary type $a = \{R_1, R_2, R_3, R_4, R_5\}$. Since each window has two intersecting points, we can generate a vector $A = (a_1, a_2, ..., a_{12})$. This vector can be the input to the second level module for heading direction learning. A major part of the developmental



(a) (b)

Figure 1: (a) A pair of images in vision based outdoor navigation with 6 attention windows; (b) Sub-image of one window and the type of road boundary (p_1's type is $R1$).

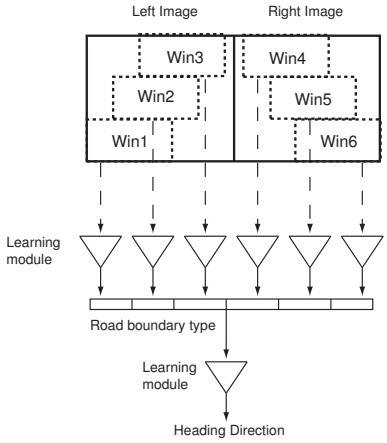


Figure 2: The system operation architecture for robotic cognitive development.

learning paradigm is Hierarchical Discriminant Regression (Hwang and Weng, 1999). Using HDR tree, the robot learns the type of road boundary in each window. The overall system operation architecture for robotic cognitive development is shown in Fig. 2. There are HDR trees at two levels. At the first level, 6 HDR trees learns the mapping from visual input to road boundary type. The second-level HDR tree maps road boundary type of all 6 windows to the correct heading direction. It is worth noting that in this study we only conduct experiments for the first-level learning modules. How the robot learns each mapping is discussed in next section.

3. System Architecture of Robotic Cognitive Development

The basic architecture for robotic cognitive development is shown in Fig. 3. The sensory input $x(t)$ is represented by a high dimensional vector. At the first level the input is visual image and the output is road edge type while at the second level the input is road boundary type vector and the output is heading direction. Each module receives reward separately.

3.1 Incremental Hierarchical Discriminant Regression

$x(t)$ is fed into the cognitive mapping module. The cognitive mapping is realized by Incremental Hierarchical Discriminant Regression (IHDR) (Hwang and Weng, 1999). IHDR clusters $x(t)$ into current state $s(t)$ and then maps the current state to the corresponding action output. Thus, IHDR generates state for high-dimensional visual input on the fly, which is a very difficult problem. That's why most of studies only deal with low-dimensional state space. In contrast, IHDR makes it possible to handle high-dimensional state space. In the testing phase, given a state $s(t)$, the IHDR finds the best matched s' associated with a list of primed contexts $(c' = (x', a', q))$, which include: primed sensations $X' = (x'_1, x'_2, ..., x'_n)$, primed actions $A' = (a'_1, a'_2, ..., a'_n)$ and corresponding Q-values $Q = (q_1, q_2, ..., q_n)$, where n is the number of different actions. In other words, the function of IHDR is g: $S \mapsto X' \times A' \times Q$. The probability to take each primed action is based on its Q-value.

3.2 Q-Learning Algorithm and Boltzmann Softmax Exploration

The basic idea of Q-learning (Watkins, 1992) is as follows. At each state s, keep a Q-value $(Q(s, c'))$ for every possible primed context c'. The action with the largest value will be selected as output and then a reward $r(t+1)$ will be received. We implemented a modified Q-learning algorithm as follows:

$$Q(s(t), c'(t)) \longleftarrow (1-\alpha)Q(s(t), c'(t)) + \alpha(r(t+1) + \gamma Q(s(t+1), c'(t+1)) \quad (1)$$

where α and γ are step-size parameter and discount-rate parameter, respectively.

The motivational system of the developmental learning paradigm works as an action selection function $v : 2^{A'} \mapsto A$ ($2^{A'}$ denotes all the possible subsets of A'), which chooses an action from a list of

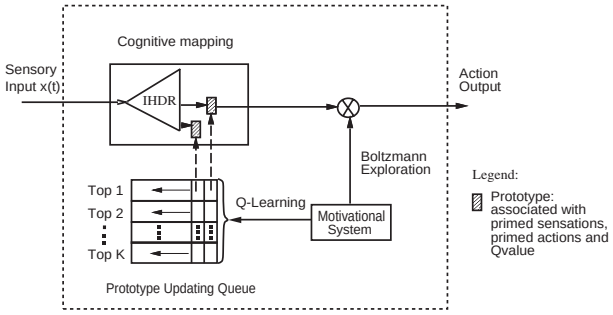


Figure 3: The system architecture of each learning module for robotic cognitive development.

primed actions. At the beginning the robot can choose random actions. Human teachers issue reward and punishment based on its actions. Through this reinforcement learning procedure the robot develops the capability to learn road type without imposed actions. In order to let the robot fully explore the action space, Boltzmann Softmax exploration is implemented. At each state (s), the robot has a list of action $A(s) = (a'_1, a'_2, ..., a'_n)$ to choose from. The probability for action a to be chosen at s is:

$$p(s, a') = \frac{e^{\frac{Q(s,a')}{\tau}}}{\Sigma_{a' \in A(s)} e^{\frac{Q(s,a')}{\tau}}} \quad (2)$$

where τ is a positive parameter called temperature. With a high temperature, all actions in $A(s)$ have almost the same probability to be chosen. When $\tau \to 0$, the Boltzmann Softmax exploration more likely chooses action a that has a high Q-value.

3.3 Prototype Updating Queue

In order to reach the real-time requirement of developmental learning, we designed the prototype updating queue in Fig. 3, which stores the addresses of formerly visited states. Thus, not only is the Q-value back-propagated, so is the primed sensation. This back-up is performed iteratively from the tail of the queue back to the head of the queue. After the entire queue is updated, the current state's address is pushed into the queue and the oldest state at the head is pushed out of the queue. Because we can limit the length of the queue, real-time updating becomes possible.

3.4 K Nearest Neighbor Q-Learning

Even though IHDR makes learning in non-stationary environments possible, the time to train the system is still not acceptable. The major reason is: as the robot explores in new environments, the state space would continue grow. In order to reduce the training time, we adopt the K-nearest neighbor strategy. For each state, its top K-nearest neighbors are saved in the prototype updating queue (Fig. 3). If a reward is issued, the system not only updates the current state-action pair but also updates the K-NN state-pairs, which dramatically reduces time complexity of the training procedure. Suppose the state space is S and the current state is $s(t)$. Let's define D as the distance between $s(t)$ and the kth -nearest neighbor of $s(t)$. The volume of the set of all states whose distance from $s(t)$ is less than D is defined as follows:

$$A(k, S, s(t)) = \frac{2D^n \pi^{n/2}}{n\Gamma(n/2)}. \quad (3)$$

Using K-NN updating rule, the updated space goes from one point $s(t)$ to A, which is a tremendous improvement.

We can also look into the training complexity issue. Suppose for each state we need to issue m rewards to get a reliable training result (Q value converges.) and the number of states is N, the time complexity of training is mN. If top K updating is applied and we assume each state has the same possibility to be chosen as a match, the time complexity is $N + \frac{(m-1)N}{K}$, where the first N means that each state has to be visited at least once while $\frac{(m-1)N}{K}$ is the number of training needed using top-K updating. The ratio of time complexity using top-1 and top-K is $\frac{mN}{N+\frac{(m-1)N}{K}} = \frac{mk}{k+m-1}$. Even though the assumption of the equal probability of each state to be chosen as top match is too strong, we still can see the potential improvement of top-K updating.

3.5 Algorithm

The algorithm of the developmental learning paradigm works in the following way:

1. Grab the new sensory input $x(t)$ and feed into IHDR tree. The IHDR tree generates a state $s(t)$ for $x(t)$.
2. Query the IHDR tree and get a matched state s' and related list of primed contexts.
3. If $s(t)$ is significantly different from s', it is considered as a new state and the IHDR tree is updated by saving $s(t)$. Otherwise, use $s(t)$ to update s' through incremental averaging.
4. Using the Boltzmann Softmax Exploration in Eq. 2 to chose an action based on the Q-value of every primed action. Execute the action.
5. Receive a reward.
6. Update the Q-value of states in PUQ using K-NN updating rule. Go to step 1.

4. Experimental Results

We train the robot to learn road boundary type for vision-based navigation and compare KNN Q-learning to traditional Q-learning.

4.1 Experiments Using Top-1 Q-Learning

Here we just show how we teach the robot to learn the road boundary type for one image (state). Totally there are 5 possible actions ($a = \{R_i | i = 0...4\}$). R_1 is the correct action. The Q-value plot of each action is shown in Fig. 4. Only information from visit No. 1 to visit No. 30 is shown. After training the Q-value of R_1 is the largest (positive) while the Q-values of other actions converge to negative values. From Fig 4, we can find out that it takes about 17 visits to converge.

4.2 Experiment Using KNN Q-Learning

In this experiment, we trained 50 consecutive images repeatedly for 20 times. Top 5 nearest-neighbors updating are used. We should notice that using top-K updating mechanism, the number of updating (n_u) is different from the number of visiting (n_v). n_v means the number of times that state s is retrieved as a top 1 match while n_u means the number of times that

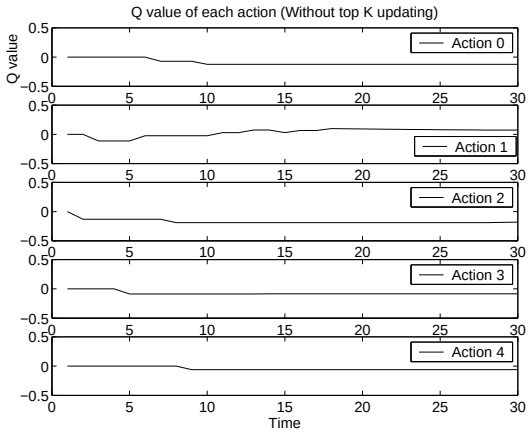


Figure 4: Q-value of each action (top-1 Q-learning).

state s is retrieved as one of the top K match. The

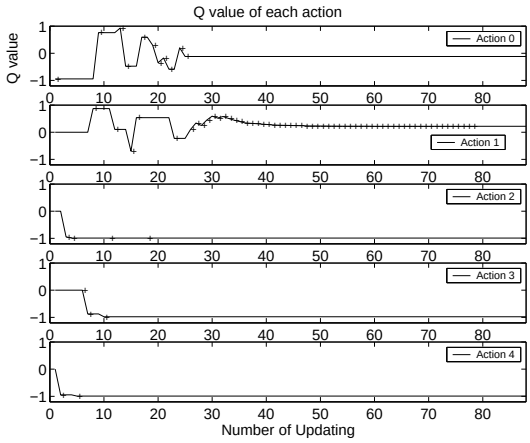


Figure 5: Q-value of each action (KNN Q-learning).

Q-value of each action versus the number of updating is shown in Fig. 5. The correct action should be action 1. After training the Q-value of action 1 is the largest (positive). "+" means that the state is chosen as top 1 match. Because of the trainer's mistake , at the beginning, action 0 got several positive rewards (Q value increases) while action 1 got a couple of negative rewards (Q value decreases). However, after 30 updatings, action 1 has the largest value and the system automatically chooses it as output. The plot also verifies that KNN-based reinforcement learning can tolerate human mistakes.

In Fig. 6, only information from visit No. 1 to visit No. 15 is shown. The first plot is the probability of each action based on its Q-value. The total probability is 1. The probabilities of action 0, 1, 2, 3 and 4 are plotted from bottom to top, respectively. The '+' denotes the random value generated by a uniform distribution. If the random value is in one range, say, the bottom range, then action 0 would be taken. As we can see, at the beginning, each action has similar probability (about 0.2). After training, action 1 is chosen for most of the time. The action sequence is

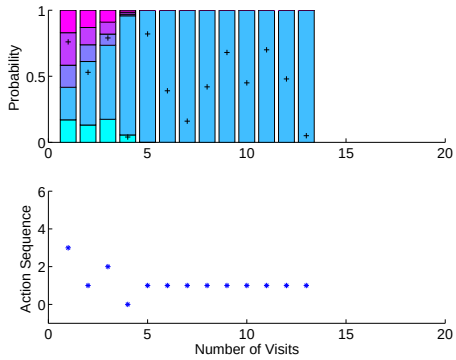


Figure 6: The first plot shows Boltzmann Softmax Exploration. The second plot shows the action sequence.

show in the second plot. When $n_v > 4$, the system only chose action 1. That is, for each attention point it takes about 5 visits to converge. This again shows the advantage of KNN-based reinforcement learning.

5. Conclusions and Future Work

In this paper, Q learning is used to model robotic covert perceptual capability development in navigation. IHDR is implemented so that learning in real-time becomes possible. And we adopt K Nearest Neighbor strategy, which dramatically reduces training time complexity in non-stationary environments. The model enables a robot to learn road boundary type through interactions with teachers. We will extend the model to develop the robot's vision-based navigation capability in the future.

References

Balkenius, C. and Hulth, N. (2001). Attention as selection-for-action: a scheme for active perception. In *Proceedings of EUROBOT '99*, pages 113–119.

Hwang, W. and Weng, J. (1999). Hierarchical discriminat regression. *IEEE Trans. on Patten Analysis and Machine Intelligence*, 22(11):1277–1293.

Minut, S. and Mahadevan, S. (2001). A reinforcement learning model of selective visual attention. In *Proceedings of the fifth international conference on Autonomous agents*, pages 457 – 464. Montreal, Quebec, Canada.

Takahashi, Y. and Asada, M. (2000). Vision-guided behavior acquisition of a mobile robot by multi-layered reinforcement learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 395–402.

Watkins, C. (1992). Q–learning. *Machine Learning*, 8:279–292.

Whitehead, S. and Ballard, D. (1990). Active perception and reinforcement learning. *Neural Computation*, 2:409–419.