Proceedings of 2004 IEEE/RSJ International Conference on
Intelligent Robots and Systems
September 28 - October 2, 2004, Sendai, Japan

Multi-Agent Reinforcement Learning: Using Macro Actions to Learn a Mating Task

Stefan Elfwing[1,2,3.4], Eiji Uchibe[2,3,4], Kenji Doya[2,3,4] and Henrik I. Christensen[1]
[1] Centre for Autonomous Systems, Numerical Analysis and Computer Science,
Royal Institute of Technology (KTH), S-10044 Stockholm, Sweden
[2] Dept. of Computational Neurobiology, ATR Computational Neuroscience Labs.
[3] CREST, Japan Science and Technology Agency
2-2-2 Hikaridai, "Keihanna Science City", Kyoto 619-0288 Japan
[4] Neural Computation Project, Initial Research Project, Okinawa Institute of Science and Technology, JST
12-22 Suzaki, Gushikawa, Okinawa 904-2234, Japan

***Abstract*— Standard reinforcement learning methods are inefficient and often inadequate for learning cooperative multi-agent tasks. For these kinds of tasks the behavior of one agent strongly depends on dynamic interaction with other agents, not only with the interaction with a static environment as in standard reinforcement learning. The success of the learning is therefore coupled to the agents' ability to predict the other agents' behaviors. In this study we try to overcome this problem by adding a few simple *macro actions*, actions that are extended in time for more than one time step. The macro actions improve the learning by making search of the state space more effective and thereby making the behavior more predictable for the other agent. In this study we have considered a cooperative mating task, which is the first step towards our aim to perform embodied evolution, where the evolutionary selection process is an integrated part of the task. We show, in simulation and hardware, that in the case of learning without macro actions, the agents fail to learn a meaningful behavior. In contrast, for the learning with macro action the agents learn a good mating behavior in reasonable time, in both simulation and hardware.**

I. Introduction

The goal of our research is to realize embodied evolution of multiple adaptive agents [11]. In conventional genetic algorithms, the process of selection and reproduction was centralized. On the other hand, in embodied evolution, as in the case of real animals, each individual must select a partner to mate and perform physical intercourse of exchanging genes. The agents should therefore learn how to find a suitable mating partner and exchange gene material with that partner. This means that the selection process will be an integrated part of the task, and the fitness of an individual will be calculated as the survival potential of the robot, i.e. a combination of how well the agent performs both foraging and mating. Our aim is to use reinforcement learning (RL) [8] to solve the tasks and to use the evolutionary process to tune the meta-aspects of the learning, such as meta-parameters and action primitives.

The first step towards this goal is to study the learning of mating behavior between two agents. Mating behavior is a relatively difficult task, requiring cooperation between two agents. Although mating is generally regarded as an innate behavior, it actually requires adaptive sensory-motor coordination by taking into account inter-individual differences in the sensorimotor system and the variability in the environment. The successful learning of mating behavior strongly depends on the interaction with the other agent, not only on the interaction with a static environment as in standard RL.

Among the theoretical studies performed in the field of multi-agent RL Littman's [6] early study is especially prominent. He introduced a framework, using game theory, for 2-player zero-sum stochastic games for multi-agent RL. This work was then extended by Hu *et al.* [4] to general-sum stochastic games.

The ability to predict the behavior of other agents is an very important issue in multi-agent RL. Nagayuki *et al.* [7] proposed an method for cooperative behaviors, where one agent estimates the other agent's actions based on an internal model of the other agent, given by observations.

There are several interesting studies using the RoboCup simulation environment as a testbed for multi-agent RL methods. Asada *et al.* [1] proposed a vision-based RL method that acquires cooperative behaviors in a dynamic environment, where the learning schedule makes the learning processes stable. Kuhlmann *et al.* [5] used SMDP Sarsa(λ) with tile-coding function approximation to successfully learn the soccer agents to play 3 vs. 2 keepaway in a realistic environmental setting.

The main reason for the difficulty to learn cooperative multi-agent tasks relates to fact that it is hard for an autonomous agent to predict the actions of another autonomous agent, which makes the outcome of the actions, new states and reward signals, very uncertain. This is especially a problem in the beginning of the learning process when the agents have not yet learned a firm policy. During this phase the agents perform a lot of exploration to acquire knowledge about the task and the environment, making their action selection very unpredictable for the mating partner.

In this paper we present a method to overcome this problem by adding simple macro actions according to Sutton's option framework [9]. The macro actions force the agent to execute the same primitive action for more than one time step, making the learning more stable and the action selection more predictable for the mating partner.

0-7803-8463-6/04/$20.00 ©2004 IEEE

II. Cyber Rodent Robot

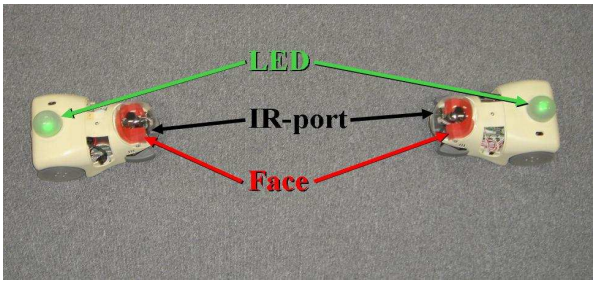


Fig. 1. The Cyber Rodent robot

This study has been performed within the Cyber Rodent (CR) project [2] [3] [10]. The main goal of the CR project is to study the adaptive mechanisms of artificial agents under the same fundamental constraints as biological agents, namely self-preservation and self-reproduction. The Cyber Rodent, shown in Figure 1, has two main features: the ability to exchange data and programs via IR-communications, for self-reproduction, and to capture and recharge from battery packs in the environment, for self-preservation.

The CR is a two-wheel mobile robot equipped with an omni-directional vision system, eight distance sensors, color LEDs for visual signaling, an audio speaker and microphones.

III. Mating Task, State Space, Actions and Reward function

In this work, we considered the simple case where one CR is predefined as sender (hereafter sender CR) in the communication and one CR is predefined as receiver (hereafter receiver CR) in the communication. The IR-port is located in front of the CR, approximately 25 mm to right of the center of the robot, as shown in figure 1. As the IR-port is directed forward, the two CRs have to face each other in a relatively small angle range for successful mating.

The input to the learning consists of the angle and the blob size to two kinds of color features extracted from the vision system. The first color feature represents the LED, green color, and is used for detection of the other agent. The second color feature represents the red area located in the front of the other CR (hereafter face) and is used for positioning the agent for mating.

If the face of the other agent is visible the state is the angle and blob size of the color feature for the face. Otherwise if the LED of the other agent is visible the state is the angle and blob size of the color feature for the LED. If the agent loses all visual information no learning is applied, instead a simple hand coded search behavior is executed.

The state space is two-dimensional and the continuous state values are normalized to the interval $[0, 1]$. The visual angle range of the CR is approximately $[-\pi/2, \pi/2]$ and the angle values are linearly mapped to the normalized interval. The relation between the blob size and the distance to the color feature is non-linear and is mapped to the normalized interval using an exponential function approximator tuned for each robot.

The agent has 9 basic actions, pairs of left and right wheel velocities, according to Table I. For the sender CR action 1 is the *sending action*. If this action is selected, the agent has got visual information from the face of the mating partner and the angle is within a reasonable range, $[-\pi/4, \pi/4]$, the sender CR performs an attempt to mate.

TABLE I
Wheel velocities for the actions

Action	Wheel velocities (mm/s)	Movement
1	(0,0)	stop/sending
2	(75,-75)	rotate left
3	(100,200)	curve right
4	(150,150)	straight ahead
5	(200,100)	curve left
6	(-75,75)	rotate right
7	(-100,-200)	back left
8	(-150,-150)	straight back
9	(-200,-100)	back right

IV. Method

A. Gradient Decent Sarsa(λ)-learning

The states values in our experiments are continuous and we therefore need to approximate the value function. We have used a normalized radial basis function (RBF) network to approximate the action value function, which is a linear function approximator. The action value function, $Q(s, a)$, for state s and action a is calculated as

$$Q(s,a) = \sum_{i=1}^{n} \phi(i,s)\theta(i,a), \tag{1}$$

where $\phi(i, s)$ is the value of the ith basis function for state s and $\theta(i, a)$ is the weight for basis function i and action a. The normalized RBFs are defined as

$$\phi(i,s) = \frac{e^{-\frac{\|s-c_i\|^2}{2\sigma_i^2}}}{\sum_{j=1}^{n} e^{-\frac{\|s-c_j\|^2}{2\sigma_j^2}}}, \tag{2}$$

where n is the total number of basis functions.

During the learning the weight matrix, $\vec{\theta}$ is updated according to *gradient decent Sarsa(λ)-learning* algorithm [8] as

$$\vec{\theta}_{t+1} = \vec{\theta}_t + \alpha\delta_t\vec{e}_t, \tag{3}$$

where α is the learning rate. δ_t is the temporal difference error defined as

$$\delta_t = r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t), \tag{4}$$

where γ is the discount factor for future rewards and A is the set of all actions. $\vec{e}_t$ is the eligible traces matrix, which is updated as

$$\vec{e}_t = \gamma\lambda\vec{e}_{t-1} + \Delta_{\vec{\theta}_t} Q_t(s_t, a_t), \tag{5}$$

where λ is the decrease factor of the traces.

The action selection is controlled by a softmax function, using a Boltzmann distribution. The probability to select action a in state s is defined as

$$P(a|s) = \frac{e^{\beta Q_t(a,s)}}{\sum_{i=1}^{k} e^{\beta Q_t(a_i,s)}}, \quad (6)$$

where β is the inverse temperature controlling the extent of exploration of the environment. If β is close to zero, $\lim_{\beta \to 0}$, the action selection becomes random. As β gets larger the actions are selected proportional to their Q-values. In the limit, $\lim_{\beta \to \infty}$, becomes greedy and the action with the largest Q-value is selected. It is preferable that the agent performs a lot of exploration in the beginning of the learning and explores less when the learned policy becomes better. Therefore, we use the following scheme for increasing β over time:

$$\beta_t = (1 + C_\beta \cdot n_{trial}(s)), \quad (7)$$

where C_β is a constant controlling the speed of the increase and $n_{trial}(s)$ is the number of times the agent has visit a certain type of state. Because we are using continuous state values, each state space dimension is divided into $k_{\beta,s}$ equidistant intervals. To ensure that there always is a small probability for exploration, we set an upper bound to β, β_{max}.

B. Options

The option framework by Sutton *et al.* [9] allows the agent to take *macro actions*, which are extended in time, i.e. executed for more than one time step. An option consists of three components: a pre-designed policy π, a termination condition and a initiation set $\mathcal{I}$, which is a subset of the state space.

A macro action can be selected if and only if $s_t \in \mathcal{I}$. If a macro action is chosen, primitive actions are selected according to the pre-designed policy until the termination condition is fulfilled. During the time a macro action is executed, the action value functions of the primitive actions are updated, in the usual way, each time step according to equation 3. The weight matrix for the macro action is updated when the macro action terminates. If a macro action is selected at time step t and is active for T time steps, the temporal difference error δ, used in equation 3, is calculated as

$$\delta = R + \gamma^T \max_{i \in A} Q_t(s_{t+k}, a_i) - Q_t(s_t, a_t), \quad (8)$$

where R is discounted cumulative reward received during the execution of the macro action: $R = \sum_{i=1}^{T} \gamma^{i-1} r_{t+i}$.

In the mating experiment we have used three simple options. For all of three the initiation set $\mathcal{I}$ is equal to the state space, and therefore all options can be selected in all states. For all options a timeout, maximum number of time steps a macro action is allowed to be executed, are applied, which is set to 5 time steps.

- **Rotate-to-target** rotates the CR towards a target, LED of face, using the action rotate left or rotate right, depending on if the angle to the target are negative or positive. The termination condition is that the angle to the target is within the range $[-\pi/10, \pi/10]$.
- **Move-back** moves the CR backwards, using the action straight back. The termination condition is that state value corresponding to blob size is greater than 0.35.
- **Move-forward** moves the CR forward, using the action straight ahead. The termination condition is that state value corresponding to blob size is less than 0.6.

V. Experiments and Results

A. Experimental Setup

The experimental procedure for the reported experiments is as follows. The CRs perform one mating trial, which ends when the two CRs successfully mate and both receives a $+1$ reward. A mating is considered successful if:

1) the sender CR executes the sending action and thereby performs an attempt to communicate with the receiver CR.
2) the sender CR successfully communicates with the receiver CR.
3) the receiver CR verifies that the communication works in both directions, by successfully communicate with the sender CR.

Before a new mating trial begins the CRs perform random movements to place the robots in different positions. If a CR, during learning, loses all visual state information, it receives a -1 reward and a simple pre-designed search behavior is executed. The search behavior rotates the CR in random direction until the robot receives visual state information again. For the reported results the elapsed time, for each robot, is only calculated during learning.

We have used the same parameters for all reported experiments, shown in table II. The specific values have been selected during a trial and error process in simulation and hardware.

TABLE II
Reinforcement Learning Parameters

α	0.125
γ	0.95
λ	0.4
C_β	0.05
β_{max}	100
$k_{\beta,s}$	5
no. of RBFs	18
time step	200 ms
trials	150
computational delay	240 ms

One problem of using function approximation in real hardware experiments is that the updating of the weight matrix takes considerable time. This gives a computational delay, i.e. the time between when the agent receives a new state and when the agent uses the new state information to

execute a new action. This limitation is the reason for the relatively small number of RBFs, shown in table II.

B. Simulation Learning

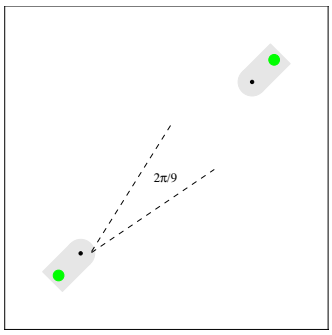


Fig. 2. The Cyber Rodent simulation environment. The figure also shows the angle range, $[-\pi/9, \pi/9]$ of the IR-communication for successful mating in the simulator.

The simulation experiments have been performed in a MATLAB simulator developed for CR project. A snapshot of the simulation environment is shown in figure 2. The figure also shows the angle range of IR-communication in the simulator, $[-\pi/9, \pi/9]$, which is approximately the same as for the hardware. The maximum distance of the IR-communication is approximately 900 mm.

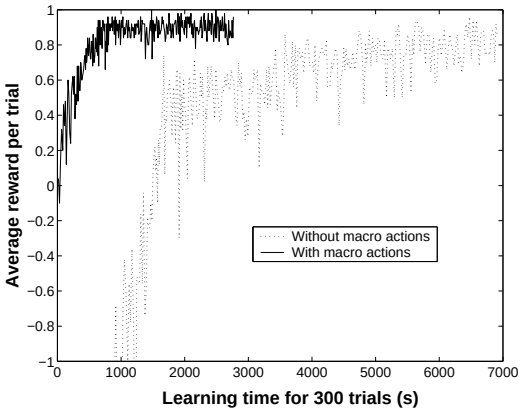
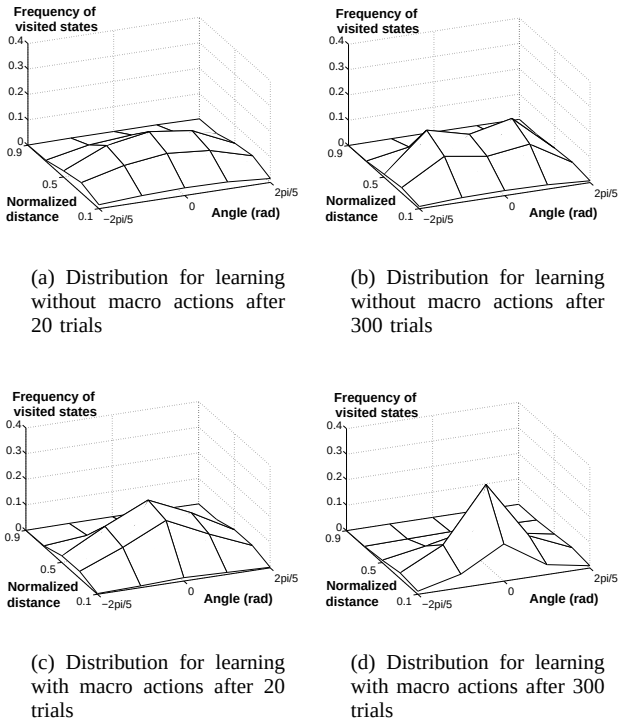


Fig. 3. Learning with and without macro actions in simulation for the sender CR. The figure shows the average reward received per trial plotted over the learning time for 300 trials. Note that the graph for the learning without macro actions is cut. During the first 1000 s the average reward per trial is constantly less than -1.

Figure 3 shows the large difference in performance for learning with and without macro actions. The figure shows the average reward per trial for 50 simulation experiments plotted as function of learning time for 300 trials. In the figure only the result for the sender CR is shown, because in simulation there was no significant difference between the sender CR and the receiver CR, neither for the characteristics of the reward curve nor the amount of reward received over the learning time. For the learning with macro actions the mating task is relatively easy and the CRs learn a good mating behavior in approximately



(a) Distribution for learning without macro actions after 20 trials

(b) Distribution for learning without macro actions after 300 trials

(c) Distribution for learning with macro actions after 20 trials

(d) Distribution for learning with macro actions after 300 trials

Fig. 4. The figure shows the normalized distribution of the types of states the sender CR visited when it had visual state information of the face of the receiver CR, in simulation. The state space is discretisized into five equidistant intervals along the two state dimensions, i.e. for the angle dimension: $[-\pi/2, -3\pi/10, -\pi/10, \pi/10, 3\pi/10, \pi/2]$. Note that the angle range of the IR-communication is $[-\pi/9\ \pi/9]$, and therefore the sender CR can almost only perform a successful mating attempt if it is visiting a state type corresponding to the angle interval $[-\pi/10\ \pi/10]$.

75 trails, 700 s. The learning converges to an average received reward per trial of approximately 0.9 and for the learned behavior the agents need in average approximately 20 actions to successfully complete a mating attempt.

Figure 4 shows the distribution of visited types of states in the beginning of learning, i.e. after 20 trials, and at the end of learning, for the sender CR when the face of the receiver CR is visible. In the early stages of the learning process it is natural that the distribution is relatively flat, see figures 4(c) and 4(a), as the agent is mostly relying on random actions. Although, it is evident that early in learning process for the learning with macro actions, see figure 4(c), the search of the state space is guided towards the most interesting area in the angle dimension, i.e. $[-\pi/10\ \pi/10]$. At end of learning, see figure 4(d), the distribution has a single large peak, which means that the sender CR has learned a good mating behavior, which can quickly move the agent to a good mating position, with respect to the position and movements of the receiver CR. It is important to notice that the performance of the sender CR is coupled to the behavior of the receiver CR i.e. the ability to predict the behavior of the receiver CR. For the learning with macro actions the search of state space becomes more directed over time. This increases the agents' ability to predict the behavior of the other agent,

which increases the learning performance.

The learning without macro actions never converges to a stable reward level, i.e. the average reward per trial is still slowly increasing after 300 learning trials. The matter of fact is that the agents never learn a good mating behavior, instead the agents are trapped in a sub-optimal behavior, where they increase the reward over time by slowly learning not lose visual information, i.e. avoiding receiving negative rewards. Even in late stages of the learning process the agents are relying a lot on random actions to successfully perform a mating attempt. This conclusion comes from

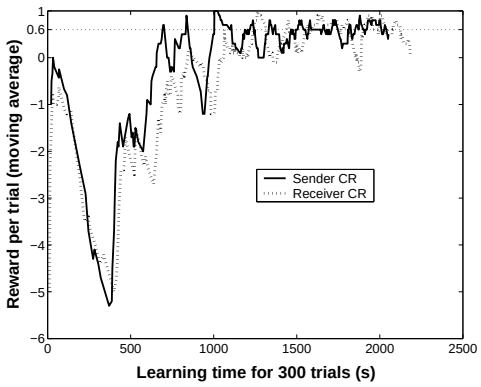
- the fact that the average number of actions that is needed to perform a mating in the second half of the learning process is remaining on relatively constant level of approximately 50 actions, even though the average reward is increasing.
- that the distribution of visited states remains relatively flat over learning process, see figures 4(a) and 4(b). This means that the sender CR has not learned a policy for moving the sender CR to a good mating position, which corresponds to a peak in the distribution for angles in the interval $[-\pi/10\ \pi/10]$.
- that there is two small peaks in final distribution, see figure 4(b), corresponding to the angle intervals $[-3\pi/10\ -\pi/10]$ and $[\pi/10\ 3\pi/10]$. Since the probability to successfully mate from these type of states are very small, the peaks are not related to the learning of the mating behavior.

C. Real Robot Learning

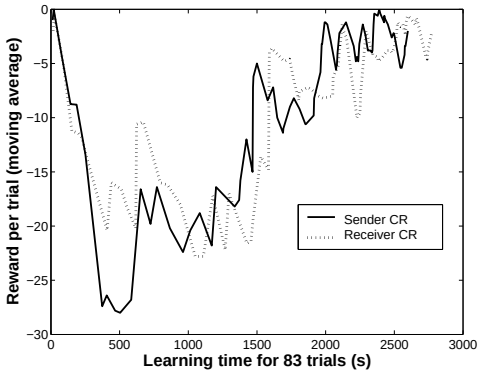
The main differences between the the hardware environment and the simulation environment is that

- the CRs lose visual state information much easier in the hardware setting, depending on that the simulated vision system is greatly simplified compared with the real hardware.
- in the simulator the learning for the two agents is synchronous, e.g. the agents execute their new actions simultaneously. In the real hardware the learning for the agents is of course asynchronous, as the learning algorithm is running on two different physical robots.
- the simulated IR-communication is greatly simplified compared with real hardware. In the simulation environment the success of the IR-communication is deterministic, i.e. if the CRs' IR-ports are inside the angle and distance ranges the IR-communication always works. In the hardware the IR-communication is somewhat stochastic, i.e. in certain situations the IR-communication works after repeated communication attempts.

For the learning with macro action, see figure 5(a), it took approximately 1000 s, corresponding to 100 trials, to learn a good mating behavior for both sender CR and receiver CR. The reward, shown as a moving average over five trials, reaches a relatively stable level of approximately 0.6. The lower received reward compared with the simulated



(a) Learning with macro actions over 300 trials

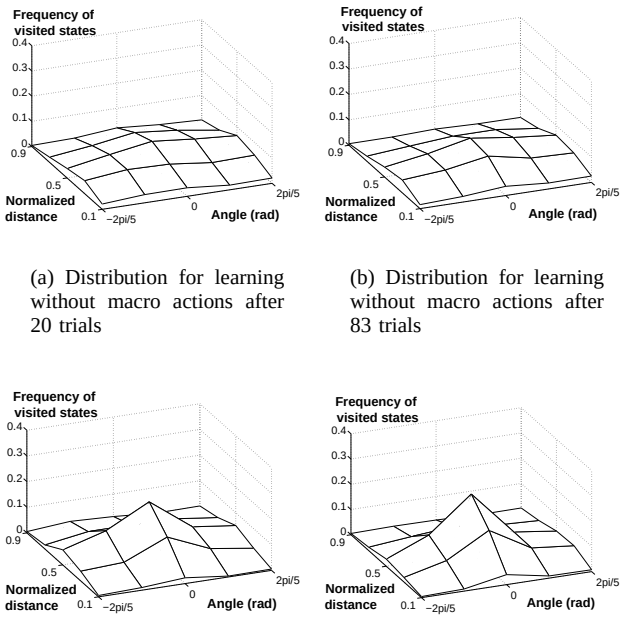


(b) Learning without macro actions over 83 trials

Fig. 5. Learning with and without macro actions in hardware for for the sender CR and the receiver CR. The figures show the reward received per trial as a moving average over 5 trials, plotted over the learning time. Note the difference in scale between the figures, for both received reward and elapsed time.

experiments is explained by the fact that the agents lose visual information much easier in the hardware. Notable is that the agents complete the 300 trials considerable faster in the hardware environment and for the learned behavior the agents need only approximately 10 actions to perform a successful mating. The distribution of visited states show that early in the learning process, see figure 6(c), the search of the state space is already directed towards angles in the interval $[-\pi/10\ \pi/10]$. This tendency is strengthened over the learning time, as shown in figure 6(d).

For the learning without macro actions the learning process was stopped after 83 trials, approximately 2600 s, as it became evident that the agents were not learning to mate. The only observable behavioral improvement was that the agents lost visual information less often, which is illustrated by the increased reward per trial over time in figure 5(b). The inability to learn to mate is illustrated by the that flat distribution of visited states, which remains almost identical over the whole learning time, see figures 6(a)



(a) Distribution for learning without macro actions after 20 trials

(b) Distribution for learning without macro actions after 83 trials

(c) Distribution for learning with macro actions after 20 trials

(d) Distribution for learning with macro actions after 300 trials

Fig. 6. The figure shows the normalized distribution of the types of states the sender CR visited when it had visual state information of the face of the receiver CR, in hardware.

and 6(b). In accordance with the simulation experiments the number of actions the agents need to perform a mating reaches relatively stable level of approximately 50 actions.

The results for the hardware experiments, as seen in figure 5 and figure 6 confirm the conclusions from the simulation experiments. The main differences in the results are related to differences between the hardware condition and the simulation condition. The agents more often receive negative rewards in the hardware, which is related to the differences between the vision system in the simulator and the real hardware. It is easier for the agents to perform the mating procedure in hardware, because the IR-communication can work successfully in both direction for a wider angle range in the hardware. This conclusion comes from that

- the learned behavior with macro actions in hardware needs half the number of actions to successfully perform a mating, compared with the behavior learned in simulation.
- in the beginning of learning in hardware both the learning with and without macro actions receives relatively high rewards, see figure 5. At this stage of learning the behavior is almost completely random and that a nearly random behavior can achieve a comparatively high performance indicates that it is relatively easy to successfully perform the mating procedure.

VI. Discussion

In this paper we have shown, in simulation and hardware, that standard RL fails to learn a cooperative mating task. Our proposed method is to add a few macro actions according Sutton's option framework, which stabilizes the learning process. We shown that for learning with macro actions the agents ability to learn the mating task increases drastically. The agents obtained a good mating behavior in reasonable time, in both simulation and hardware.

This study is the first step towards our goal to perform embodied evolution experiments on the CR robots. The successful learning of mating behavior, especially in the hardware setting, is very encouraging for our future work.

Acknowledgment

This research was conducted as part of "Research on Human Communication"; with funding from the Telecommunications Advancement Organization of Japan

Stefan Elfwing's part of this research has also been sponsored by a shared grant from the Swedish Foundation for Internationalization of Research and Education (STINT) and the Swedish Foundation for Strategic Research (SSF). The funding is gratefully acknowledged.

References

[1] M. Asada, E. Uchibe and K. Hosoda. "Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development.", *Artificial Intelligence*, 110:275–292, 1999.

[2] S. Elfwing, E. Uchibe, and K. Doya. "An Evolutionary Approach to Automatic Construction of the Structure in Hierarchical Reinforcement Learning." In *Proc. of the Genetic and Evolutionary Computaion Conference*, 2003.

[3] A. Eriksson, G. Capi, and K. Doya. "Evolution of Meta-parameters in Reinforcement Learning Algorithm.", In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[4] J. Hu and M. P. Wellman. "Multiagent Reinforcement Learning: Theoretical Framework and an Algorithm", In *Proc. of the Fifteenth International Conference on Machine Learning*, 242–250, 1998.

[5] G. Kuhlmann and P. Stone. "Progress in Learning 3 vs. 2 Keepaway.", In *RoboCup-2003: Robot Soccer World Cup VII*, Springer Verlag, Berlin, 2004.

[6] M. L. Littman. "Markov games as a framework for multi-agent reinforcement learning", In *Proc. of the Eleventh International Conference on Machine Learning*, 157–163, 1994.

[7] Y. Nagayuki, S. Ishii and K. Doya. "Multi-Agent Reinforcement Learning: An Approach Based on the Other Agent's Internal Model", In *Proc. of the Fourth International Conference on Multi-Agent Systems*, 215–221, 2000.

[8] R. S. Sutton and A. Barto, *Reinforcement Learning: An introduction*, MIT Press, Cambridge, MA, 1998.

[9] R. S. Sutton, D. Precup, and S. Singh. "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning." *Artificial intelligence*, 112:181-211, 1999.

[10] E. Uchibe and K. Doya. "Competitive-Cooperative-Concurrent Reinforcement Learning with Importance Sampling", In *Proc. of International Conference on Simulation of Adaptive Behavior: From Animals and Animats*, 2004.

[11] R. A. Watson, S. G. Ficici, and J. B. Pollack. "Embodied Evolution: Distributing an evolutionary algorithm in a population of robots." *Robotics and Autonomous Systems*, 39:1–18, 2002.