自動操縦への適応のモデル化
Modeling Adaptation on Automated Vehicle Operation

森田 純哉†, 三輪 和久†, 前東 晃礼†, 寺井仁†, 小島 一晃‡ and Frank E. Ritter††
Junya Morita†, Kazuhisa Miwa†, Akihiro Maehigashi†, Hitoshi Terai†, Kazuaki Kojima‡ and Frank E. Ritter††

† 名古屋大学, ‡ 帝京大学, †† Pennsylvania State University
† Nagoya University, ‡ Teikyo University, †† Pennsylvania State University
j-morita@cmc.ss.is.nagoya-u.ac.jp

Abstract

This paper presents a cognitive model that simulates how the reliance on automation in a simple tracking task, which represents vehicle operation, changes as the success probabilities of automatic and manual mode vary. The model was developed by using ACT-R, and we also introduce three methods of reinforcement learning: the synchronization of utilities in the same mode, the summation of rewards over time, and the prediction from the past. The model performs this task through productions that manage perception and motor control. The utility values of these productions are updated based on rewards in every perception-action cycle. A run of this model simulated the overall trends of the behavioral data, suggesting some validity of the assumptions made in our model.

Keywords — Automated Operation, Reinforcement Learning, ACT-R, Semi-Markov Decision Process

1. Introduction

For sustainable industrial development, it is important to understand how humans adapt new technologies. Disuse of new technology results in less innovation, while misuse of new technology can cause serious accidents [e.g., 1, 2]. We focus here on adaptation to automated vehicles. This currently exists for speed control in many modern vehicles ("cruise control"). We examine here steering control, which is not common at the present but it is a technology being developed.

Vehicle operation is a dynamic continuous process in which the cycle of perception, judgment, and action sequentially repeats. Automation vehicle systems partially substitute such human operation. In a case where an operator can use automatic operation, s/he repeats the cycle of perception and judgment while observing that an automation system executes the overall cycle. When the operator considers manual control is suitable, he can turn off the automation to return to manual control.

Such an adaptation mechanism is possibly explained by reinforcement learning, which updates selection probabilities of actions with reward from the environment [3]. In a broader context, this paradigm has already been used to model the interaction between human and automation systems. Gao and Lee [4] proposed a computational model called Extended Decision Field Theory that simulates how operators adopt a system that automates plant operation. In their model, a selection probability of using the automation system is dynamically changed through iterated environmental feedback. Although Gao and Lee did not refer to any studies about reinforcement learning, their model is essentially a type of reinforcement learning.

However it is not simple to apply the paradigm of reinforcement learning to our problem. Generally, reinforcement learning has been applied to discrete fields called Markov Decision Process (MDP) like bandit tasks [3]. Plant operation modeled by Gao and Lee is also classified into an MDP process. Contrary to typically applied fields of reinforcement learning, vehicle operation does not directly fit into MDP. Rather it can be expressed as an SMDP (Semi-Markov Decision Process). SMDP introduces the concept of a time delay between action selection and state transition, and rewards that can be delivered at different points in time [5, 6, 7, 8, 9]. In this study, we prepared a task that extracts continuous characteristics of vehicle operation and construct a model to reveal what type of mechanisms are needed for proper adaptation on automatic vehicle operation.

2. The Task

This study extends our previous study adding new learning algorithms. Morita et al [10] simulated a psy-

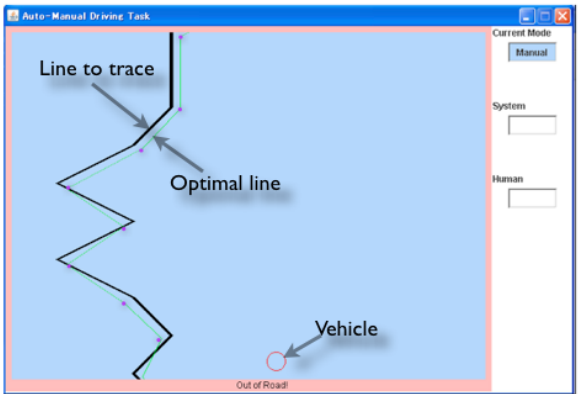

Figure 1 The Line-following task environment

chological experiment conducted by Maehigashi et al [11]. The task used in these studies was a simple tracking task, called the line-following task. Figure 1 shows the task environment. In this task, the operators are required to control the horizontal position of the vehicle (red circle) to follow the black line that scrolls down at 24 pixels per second. The screen is updated every 40 ms. If the vehicle is not on the line, a warning is presented outside of the window. The line is drawn by randomly combining 48 pixels high line patterns of varied angles (30, 45, 90, 135, and 150 degrees).

The vehicle is controlled by commands of "left", "straight", or "right". If the vehicle receives a left command, the vehicle moves 1 pixel left from the original position. The command is sampled at 48 Hz[1]. Therefore, maximally, the vehicle can move 2 pixels per one pixel scroll of the line.

An operator can chose manual or auto controls to send commands. In the manual control, operators use left and right arrow keys to send commands. If an operator's finger is put on a right arrow key, the vehicle keeps receiving a right command at every 20 ms until the key is released. In the auto control, operators monitor that the auto control moves the vehicle. The auto control tries to follow an optimal line presented as the green line in Figure 1. An optimal line is the shortest line to pass "goals" located on each corner shown as blue dots. If the center of the vehicle is off the optimal line, the auto control system sends a command to correct the vehicle position. In the experiment, the optimal line and goals are not visible to participants.

[1]If a key-press event is detected, a flag of sending commands is on. This flag is off when a key-release event is detected. Therefore, the command rate is not influenced by a key-repeat rate setting in an operating system.

In both control modes, commands are not always successfully sent to the vehicle. Failures occur at specified rates. In this study, Ca and Cm specify these rates. If Ca or Cm is low, the vehicle controlled by the corresponding mode is lagged, and it becomes hard to follow the line. To conduct the task successfully, operators need to select a suitable mode in each situation. The operators freely change between modes by pressing the space-bar.

3. Model

3.1 Architecture

Morita et al. [10] used ACT-R (Adaptive Control of Thought-Rational) [12] to construct a model for the above task. This architecture integrates several cognitive modules including a visual module, a motor module, and a production module. A visual module is used to take information from an external environment. A motor module manipulates devices like a keyboard or a mouse in an external environment. These modules have buffers to hold temporarily information called a chunk. A production module integrates the other modules by production rules, which consists of a condition/action pair that is used in sequence with other productions to perform a task. Conditions and actions in production rules are specified with buffer contents of each module.

Importantly, each event caused by the modules of ACT-R has a parameter of time approximation. For example, ACT-R production rules take 50 ms to apply. Events related visual perception and motor controls, such as eye-movement, mouse-movements and key-presses, also have time parameters. These parameters were assigned and validated by psychological studies [13][2]. By using these parameters, ACT-R makes real-time simulations possible.

ACT-R also includes sub-symbolic cognitive processes that modulate the probabilites of firing production rules. When several rules match to buffer conditions, conflict resolution of production rules is made based on utility values assigned to production rules. The learning of utility is controlled by equation 1:

$$U_i(n) = U_i(n-1) + \alpha[R_i(n) - U_i(n-1)] \qquad (1)$$

α is the learning rate; $R_i(n)$ is the reward value given to production i at time n. The learning occurs when

[2]This approach was originally developed by Card et al [14]. The perceptual-motor system of ACT-R was also developed based on EPIC architecture [15]

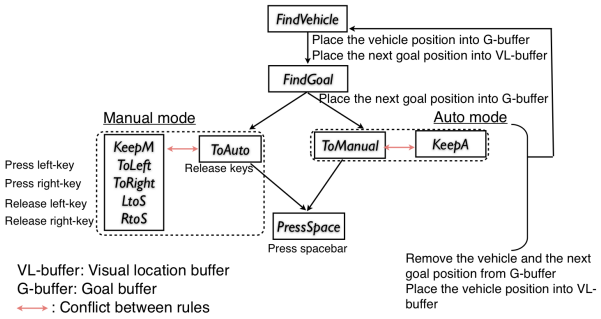

Figure 2 The basic cycle of the model.

a reward is triggered, and all productions that have fired since the last reward are updated. Though the theory of ACT-R [12] does not explicitly note it, this learning is same as the basic reinforcement learning method called Q-learning, which updates the quality of a state-action combination by recieving rewards from the environment [3].

We considered that the above characteristics (the visual and motor modules to interact with external environments, the real-time simulation, the utility apdate based on reinforcement learning) are useful for modeling an adaptation process on automatic vehicle operation.

3.2 Simulated task environment

By using the ACT-R graphical user interface (AGI) that is part of ACT-R 6 [12], we developed a simulated task environment with which the constructed model interacts. The simulated environment is same as the original environment in the keyboard layout, the screen update rates, the line scrolling speed, the vehicle size, the line width, and the screen size. The auto control mode is also implemented with Common Lisp in the simulated task environment. However, unlike the original environment, visible goal positions are set at each corner to allow the model to perceive the path.

3.3 Basic cycle of the model

Figure 2 indicates a basic cycle of the model constructed in our previous study [10]. The model uses the production, goal, vision and motor modules of ACT-R 6, and 11 production rules. These rules consist of a perceptual (the top part of the figure) and motor process (the bottom part of the figure) similar to previous driving models in ACT-R [16, 17, 18].

In the perceptual process, the model picks visual information from a visual location buffer that holds location information of objects in the environment. The *FindVehicle* rule finds the horizontal position of the vehicle, and places it into the goal buffer. The *FindGoal* rule finds the horizontal position of the nearest goal position, and places it into the goal buffer. The position information in the goal buffer is used in the subsequent motor process. After the motor process, information in the goal buffer is cleared to begin the next cycle.

The motor process depends on the current mode. In each mode, there is a rule to switch the current mode (*ToAuto* / *ToManual*). These mode-switching rules send a command to release currently pressed keys to the motor module. After finishing the key-release, the *PressSpace* rule sends a motor command pressing the space-bar.

The mode-switching rules compete with other rules in each situation. In the auto mode, the *ToManual* rule conflicts with the *KeepA* rule that just clears the goal buffer. In the manual mode, the *ToAuto* rule competes with the *KeepM*, *ToLeft*, *ToRight*, *LtoS*, and *RtoS* rules. These five rules have different conditions specifying the vehicle and the goal positions, and current move-commands (left, right, straight). The action clauses of the *ToLeft*, *ToRight*, *LtoS*, *RtoS* rules send a command to hold or release a key to the motor module[3]. The *KeepM* rule does not have any action clauses relating the motor module. This rule just clears the goal buffer.

Figure 3 presents a time flow diagram showing the relations between the environmental changes and the model cycles. The environment regularly updates the screen every 40 ms. Individual rule firings take 50 ms, but the cycle of the model is not regulated. There are delays in the visual and motor processes. The process of the visual location module itself has no delay. However, encoding the location into the goal buffer lags 10 ms. from the environment. The delay of the motor control is larger than that of the perceptual module. The ACT-R motor module needs preparation and execution time, which depends on the status of the motor module. These delays disadvantage manual control compared to automatic control.

In the previous study using this model, the fitting to the data could be improved, though the model simulated some qualitative tendencies of the psychologi-

[3]The default ACT-R implementation does not include key-press and key-release functions. We used a customised module in which the time parameter of key-punch is used.

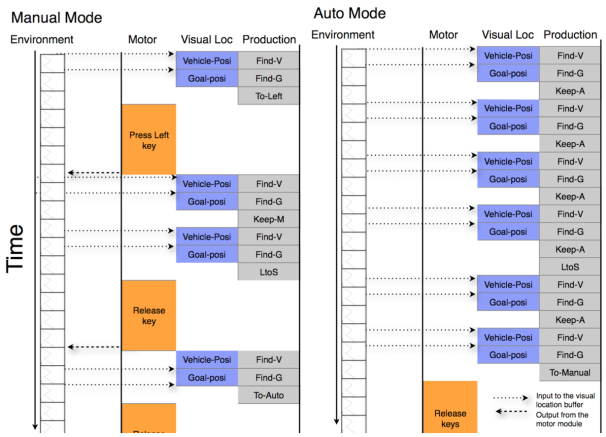

Figure 3 Time flow diagram of the ACT-R model components in the task.

cal experiment. The following section describes learning mechanisms included in the model to improve its fit to human behaivor.

3.4 Learning and mode switching

In studies that use ACT-R reinforcement learning [12, 13, 19], a reward parameter is assigned to the specific rules by a modeler. When a rule with a reward value fires, all rules that have fired since the last utility update receive the reward. Following this paradigm, we assigned a reward parameter to the *FindVehicle* rule (see Figure 2). In every cycle of the model, this rule perceives the vehicle location. Thus, it is reasonable to assume that this rule triggers rewards by checking a relative position between the vehicle and the line. However there are some difficulties for simply applying this paradigm to our study.

First, the structures of the conflicts are not the same between the manual and the auto modes. The ToAuto rule that changes the current mode to the auto mode conflicts with five rules in the manual mode (See Figure 2). On the other hand, the *ToManual* rule that changes the current mode to the manual mode conflicts only with the *KeepA* rule. To solve this asymmetry, we built a synchronization mechanism for the rules in the manual control. If any of the rules that compete with the *ToAuto* rule updates its utility, other rules also update their utilities[4]. This synchronization mechanism is not implemented in ACT-R. However, the studies of hierarchical reinforcement learning have proposed a similar mechanism that chunks together primitive actions into macro actions [6, 20].

Second, as shown in Figure 3, there are motor delays in the ACT-R architecture. Reflecting these delays, the rules in the manual mode receive less opportunity for updating rewards. This rewarding problem typically occurs in reinforcement learning in SMDP situations. To solve this, some researchers propose a procedure that sums received reward values over time [5, 7, 8]. Following these studies, we modified a rewarding approach. The reward values in this study do not directly correspond to a success at each point of utility updates, but correspond to movement distance (pixels) since the last utility update.

Third, during the cycle in Figure 2, the utilities of the two modes are not directly compared. The rules that keep the current mode receive rewards corresponding how well the vehicle moves in the current mode. Contrary, rewards for the mode-switching rules are influenced by the vehicle movement in both of the two modes because the perception-action cycle of mode-swiching bridges across the two modes (see Figure 3). Accordingly, mode switching is made only based on the utilities for the current mode without considering how well the vehicle moved in the other mode. Adaptation to the proper mode may be possible even if this problem exists. However, it is reasonably assumed that human can predict future reward by remembering past experience. Some researchers proposed a paradigm of the model-base reinforcement learning that predicts future reward based on environmental model [21, 22]. Therefore, we introduced the meta level conflict-resolution that directly compare the two modes. The details of the meta-level conflict resolution are described in Morita et al [10].

4. Model Simulation

Maehigash et al [11] conducted the task in 25 conditions where Ca (Capability of Auto) and Cm (Capability of Manual) levels were manipulated (5 levels of Ca ranging from 30% to 70% v.s. 5 levels of Cm ranging from 30% to 70%). Similarly, the model conducted the task choosing two modes of control in the 25 conditions (n = 100). In each condition, the model and the participants conducted the task for 40 seconds.

Figure 4 indicates the auto use ratio in each Ca and Cm level, which represents how long the auto mode is used during the task. Comparison of the five graphs reveals decreases of auto use ratio with increases of

[4]The Common Lisp program that controls the simulation interrupts the model process, and the utilities of rules were overwritten by the *spp* command.

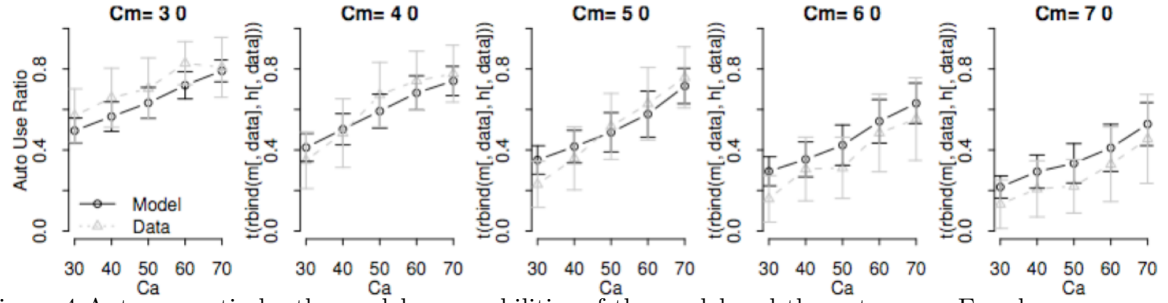

Figure 4 Auto use ratio by the model as capabilities of the model and the auto vary. Errorbars represent $\pm 1/2\,\sigma$.

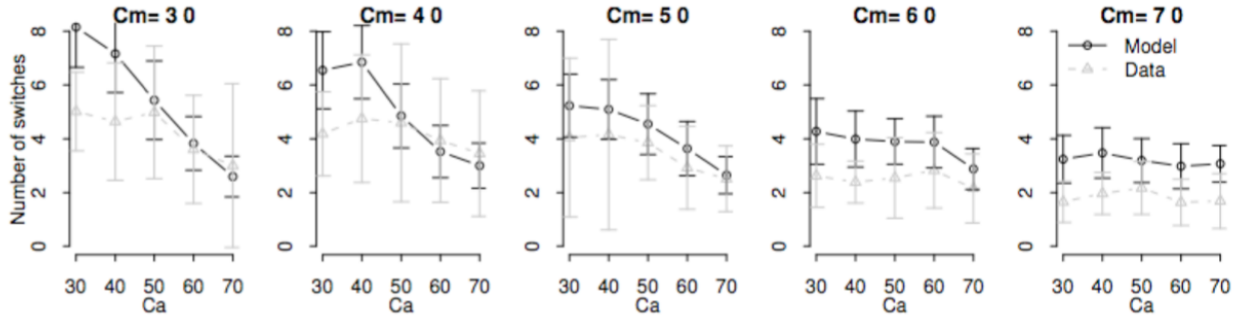

Figure 5 Number of switches during a run between auto and manual as control capabilities of the model and the auto vary. Errorbars represent $\pm 1/2\,\sigma$.

the Cm level. We can also see an increase of the auto use ratio with increases of the Cm level from each graph. The model shares these tendencies with the data. Figure 5 presents the number of switches between models in each Ca and Cm level, which represents how many switches occurred between the two modes. From this figure, we can also observe the same tendencies between the experiment and the simulation, including the decrease of the number of switches with the increase of Ca and Cm.

5. Conclusion

The present study tried to simulate adaptations on automated vehicle operation in a simple tracking task. We assumed that a paradigm of reinforcement leaning can be applied to this problem. Based on this assumption, we used the ACT-R architecture to explore mechanisms to simulate the psychological data. The results of the simulation show overall correspondence with the experimental data, suggesting validity of our assumption.

We consider that the strength of this work is in combining ACT-R with studies of reinforcement leaning in SMDP. ACT-R has so far been used to simulate many psychological experiments. Because of this history, this architecture made possible to produce a simulation that can directly compare with human data. In our study, the perceptual and motor modules of ACT-R were used to represent time constraints of the task (Figure 3).

However, to simulate the experiment, we needed to extend the reinforcement learning implemented in ACT-R. These extensions included the synchronization of utilities in the same mode, the summation of rewards over time, and the prediction from the past. Without these, we could not achieve as good a fit to the data (Figure 4 and 5).

Despite the long history of this architecture, studies using the reinforcement learning of ACT-R have not been so common. In this community, instance-based learning, which uses declarative memory, is more popular than reinforcement learning to simulate decision-making in discrete tasks [23, 24, 25]. On the other hand, reinforcement learning in SMDP has been mainly developed in the field of control engineering and robotics. The extensions of reinforcement learning made in our study are not new in these fields. Several studies have tried to apply reinforcement learning to complex and dynamic situation [6, 20, 7, 21]. These studies were not aimed to make predictions of human

behavior, though the same authors discuss theoretical connections between reinforcement learning and neural computation especially concerning dopamine release [22, 26, 27].

In sum, we consider that our study has not only the practical merit of presenting modeling techniques for adaptations on automated vehicle operation, but also the theoretical merit of combining ACT-R with reinforcement learning theories.

References

[1] L. Bainbridge. Ironies of automation. *Automatica*, Vol. 19, No. 6, pp. 775–779, 1983.

[2] R. Parasuraman and V. Riley. Humans and automation: Use, misuse, disuse, abuse. *Human Factors*, Vol. 39, No. 2, pp. 230–253, 1997.

[3] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[4] J. Gao and J. D. Lee. Extending the decision field theory to model operators' reliance on automation in supervisory control situations. *IEEE Transactions on Systems Man and Cybernetics*, Vol. 36, No. 5, pp. 943–959, 2006.

[5] M. O. Duff and S. J. Bradtke. Reinforcement learning method for continuous-time Markov Decision Problems. *Advances in Neural Information Processing Systems 7*, pp. 393–400, 1996.

[6] R. S. Sutton, D. Precup, and S. Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, Vol. 112, pp. 181–211, 1999.

[7] 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*, Vol. 110, No. 2, pp. 275–292, 1999.

[8] D. Rasmussen and C. Eliasmith. A neural reinforcement learning model for tasks with unknown time delays. In *Proceedings of the 35th Annual Conference of the Cognitive Science Society*, pp. 3257–3262, 2013.

[9] O. L. Georgeon and F. E. Ritter. An intrinsically-motivated schema mechanism to model and simulate emergent cognition. *Cognitive Systems Research*, Vol. 15-16, pp. 73–92, 2012.

[10] J. Morita, K. Miwa, A. Maehigashi, H. Terai, K. Kojima, and F. E. Ritter. Modeling decision making on the use of automation. In *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, pp. 1971–1976, 2011.

[11] A. Maehigashi, K. Miwa, H. Terai, K. Kojima, and J. Morita. Experimental investigation of calibration and resolution in human-automation system interaction. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E96-A, No. 7, pp. 1625–1636, 2013.

[12] J. R. Anderson. *How can the human mind occur in the physical universe?* Oxford University Press, New York, 2007.

[13] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, Lebiere C, and Y. Qin. An integrated theory of the mind. *Psychological Review*, Vol. 111, pp. 1036–1060, 2004.

[14] S. K. Card, T. P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, 1983.

[15] D. E. Kieras and D. E. Meyer. An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human-Computer Interaction*, Vol. 12, No. 4, pp. 391–438, 1997.

[16] D. D. Salvucci. Modeling driver behavior in a cognitive architecture. *Human Factors*, Vol. 48, pp. 362–380, 2006.

[17] D. D. Salvucci and N. A. Taatgen. *The Multitasking Mind*. Oxford University Press, New York, 2010.

[18] F. E. Ritter, U. Kukreja, and R. St. Amant. Including a model of visual processing with a cognitive architecture to model a simple teleoperation task. *Journal of Cognitive Engineering and Decision Making*, Vol. 1, No. 2, pp. 121–147, 2007.

[19] M. C. Lovett and J. R. Anderson. History of success and current context in problem solving: Combined influences on operator selection. *Cognitive Psychology*, Vol. 31, No. 2, pp. 168–217, 1996.

[20] S. Elfwing, E. Uchibe, K. Doya, and H. I. Christensen. Multi-agent reinforcement learning: using macro actions to learn a mating task. In *Proceedings of 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3164–3169, 2004.

[21] K. Doya, K. Samejima, K. Katagiri, and M. Kawato. Multiple model-based reinforcement learning. *Neural Computation*, Vol. 14, No. 6, pp. 1347–1369, 2002.

[22] H. Nakahara and O. Hikosaka. Learning to represent reward structure: A key to adapting to complex environments. *Neuroscience Research*, Vol. 74, No. 3-4, pp. 177–183, 2012.

[23] C. Lebiere, C. Gonzalez, and M. Martin. Instance-based decision making model of repeated binary choice. In *Proceedings of the 8th International Conference on Cognitive Modeling*, pp. 67–72, 2007.

[24] C. Lebiere, C. Gonzalez, and W. Warwick. Convergence and constraints revealed in a qualitative model comparison. *Journal of Cognitive Engineering and Decision Making*, Vol. 3, No. 2, pp. 131–135, 2009.

[25] J. Morita, T. Konno, and T. Hashimoto. The role of imitation in generating a shared communication system. In *Proceedings of the 34th Annual Conference of the Cognitive Science Society*, pp. 779–784, 2012.

[26] N. D. Daw, S. Kakade, and P. Dayan. Opponent interactions between serotonin and dopamine. *Neural Networks*, 2002.

[27] N. D. Daw D and K. Doya. The computational neurobiology of learning and reward. *Current Opinion in Neurobiology*, Vol. 16, No. 2, pp. 199–204, 2006.