Advanced Robotics, Vol. 21, No. 12, pp. 1351–1367 (2007)

Also available online - www.brill.nl/ar

Full paper

Experience-based imitation using RNNPB

RYUNOSUKE YOKOYA [1,*], TETSUYA OGATA [1], JUN TANI [2],
KAZUNORI KOMATANI [1] and HIROSHI G. OKUNO [1]

[1] *Graduate School of Informatics, Kyoto University, Yoshida-honmachi Sakyo-ku, Kyoto 606-8501, Japan*
[2] *Brain Science Institute, RIKEN, 2-1 Hirosawa Wako-shi, Saitama 351-0198, Japan*

Received 3 October 2006; accepted 6 March 2007

Abstract—Robot imitation is a useful and promising alternative to robot programming. Robot imitation involves two crucial issues. The first is how a robot can imitate a human whose physical structure and properties differ greatly from its own. The second is how the robot can generate various motions from finite programmable patterns (generalization). This paper describes a novel approach to robot imitation based on its own physical experiences. We considered the target task of moving an object on a table. For imitation, we focused on an active sensing process in which the robot acquires the relation between the object's motion and its own arm motion. For generalization, we applied the RNNPB (recurrent neural network with parametric bias) model to enable recognition/generation of imitation motions. The robot associates the arm motion which reproduces the observed object's motion presented by a human operator. Experimental results proved the generalization capability of our method, which enables the robot to imitate not only motion it has experienced, but also unknown motion through nonlinear combination of the experienced motions.

Keywords: Imitation; active sensing; humanoid robot; recurrent neural network.

1. INTRODUCTION

The final goal of this work was to develop a method that enabled robots to imitate human motion. Human adults can easily learn by watching the behavior of others and imitating them. Even infants can learn through imitating facial and hand gestures. With this significant ability, human beings can acquire new behaviors from others within an incredibly short time. From the standpoint of robot learning, any method that enables robots to imitate humans can significantly speed up the learning process [1]. The learning load is crucial to real robots because of problems with durability. It is also almost impossible to program robots manually to make every conceivable motion.

*To whom correspondence should be addressed. E-mail: yokoya@kuis.kyoto-u.ac.jp

With advances in hardware technologies, humanoid robots can now realize several kinds of motion, e.g., two-legged locomotion, running and rising. Some of them have tried to imitate human motion. Nakazawa *et al.* developed a humanoid robot that imitates dancing using a motion capture system [2]. In their study, the robot imitated the trajectories for every part of the human body. The target of their work is to control the robot's joint angles to be the same as the human's. Therefore, the motors are occasionally compelled to output extremely large torque due to differences in body dynamics. Thus, most conventional studies usually designed the recognition process as pattern clustering and the motion-generating process was isolated from the recognition process.

For robot imitation, in this work we focus on two factors: 'mirror neurons' in the brain and infant's 'body babbling'.

The mirror neurons were originally discovered in area F5 of the monkey premotor cortex, which discharge both when the monkey makes a particular action and when it observes another making a similar action [3]. The neurons suggest that both recognition and generation processes are conducted in the same structure in brain. This work uses the neural net model called the RNNPB (recurrent neural network with parametric bias) model [4] that can work as both recognition and generation functions. The detail of the model is described in Section 2.

'Body babbling' is an experiential process where infants learn what muscle movements achieve a particular goal state [5]. This process enables infants to acquire a mapping between dynamic patterns of movement and a resulting body part configuration. Based on this fact, this work introduces the active sensing process as the robot's experiential process where the robot acquires a mapping between it's own motions and target motions based on real experiences.

Our target task is moving an object on a table. In our imitation architecture, the recognition process is implemented not as the clustering of generated patterns, but as the prediction of pattern generation (forward model). Based on this approach, Ogata *et al.* proposed the active recognition model using a humanoid robot and the RNNPB model [6]. The prediction of the object's motion while manipulating enables the robot to generate the motion at the next moment (inverse model).

Section 2 describes our imitation architecture which is based on active sensing and RNNPB. Section 3 describes the implementation of the robot hardware and the neural net model. Section 4 describes the imitation experiments and the obtained results. Section 5 discusses the prediction and generalization capabilities of our architecture as an imitation model. Section 6 concludes this paper.

2. IMITATION METHOD BASED ON ACTIVE SENSING

2.1. Overview of our imitation process

We present an overview of our method, which enables a robot to imitate human behaviors by using the experience of active sensing. The imitator essentially has to

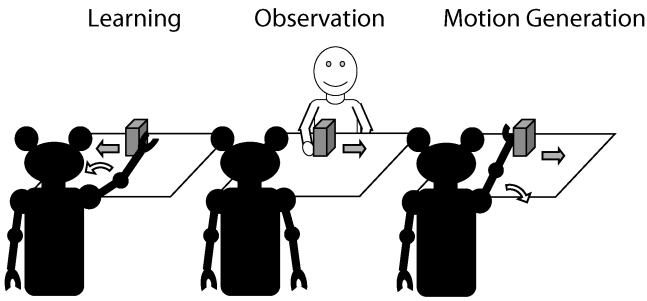


Figure 1. Imitation process.

discover a part to be imitated thorough interaction with others. For simplifying the verification of the effectiveness of the method, in this work we defined the part to be imitated as the trajectory of the object. Our imitation process consists of three phases: learning, observation and motion-generating phases (Fig. 1). We overview it as follows:

(i) *Learning (object recognition)*. The robot relates its arm motions to the object's motions while it manipulates the object (active sensing). The experience of active sensing enables the robot to predict the object's motion.

(ii) *Observation (motion planning)*. The robot observes target object manipulation generated by a human teacher focusing not on the teacher's motion, but on the object's motion. The robot planned its arm motion which can generate similar object motions.

(iii) *Motion generation (imitation)*. The robot actually generates the arm motion planned in the previous phase.

There needs to be some kind of method to acquire the relation between the robot's motion and the object's motion. The robot's motion has to be generated using only limited patterns of learnable object manipulations which are limited due to real robots having problems with durability.

The RNNPB model has advantages in that it can acquire self-organized behavioral primitives as parameter values, 'PB values'. The most significant feature of the model is its generalization capabilities. By taking advantage of the RNNPB model, in this work the robot's motion was associated with the object's motion with PB values.

2.2. Learning model

This section describes the learning model used in our method, i.e., the RNNPB model, and its learning algorithm.

2.2.1. RNNPB. The RNNPB model is the FF (forwarding forward) model [7] proposed by Tani and Ito. The RNNPB model works as a prediction system: its input data is the current sensory state $S(t)$ and its output data is the predicted sensory

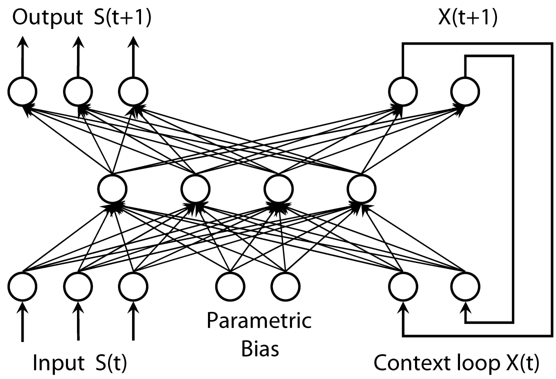


Figure 2. RNNPB.

state $S(t+1)$ in the next step. The network configuration for the RNNPB model is outlined in Fig. 2. This model has the same architecture as the conventional hierarchical neural network model, except for the context layer and the PB nodes in the input layer. Unlike the other input nodes, these PB nodes take a constant value throughout each time sequence. The context layer has a loop that inputs the current output as input data in the next step. An advantage of this layer is that the RNNPB model can use it to learn the time sequences by leveraging past contexts. After learning the time sequences, the RNNPB model self-organizes the PB values at which the specific properties of each individual time sequence are encoded.

The RNNPB model learns with a particular learning algorithm. Although the learning algorithm for the conventional hierarchical neural network is back propagation, the RNNPB model cannot learn with this algorithm because it does not have a teacher signal to the context layer. Consequently, a learning algorithm called the BPTT (back propagation through time) [8] is employed.

2.2.2. Learning PB value. The PB values are calculated during the learning process as follows, unlike a general calculation method [4]:

$$\delta\rho_t = k_{\mathrm{bp}} \cdot \sum_{0}^{T} \delta_t^{\mathrm{bp}} \tag{1}$$

$$\rho_t' = \rho_t + \delta\rho_t \tag{2}$$

$$p_t = \mathrm{sigmoid}(\rho_t'), \tag{3}$$

where k_{bp} is a constant, ρ_t is the internal value of the PB node at t, p_t is the PB value of the PB node at t, δ_t^{bp} is the δ error back-propagated from the output nodes to the PB nodes and T is the sensory sequence length. In (1), the δ errors are integrated errors in all the steps. In (3), the current PB values are obtained from the sigmoidal outputs of the internal values. In this work, one pair of PB values is calculated because our goal was to acquire specific PB values corresponding to each object manipulation. The PB values after threshold processing can also be utilized as quasi-symbols for human–robot interaction [9].

2.3. Calculation in the observation and motion-generating phases

After the RNNPB model is organized in the BPTT and the PB values are calculated in the learning phase, the RNNPB model is used in the observation and motion-generating phases. This section describes how the RNNPB model is used in the observation and motion-generating phases.

2.3.1. Method for recognizing manipulation. This section describes how the manipulation presented by the teacher in the observation phase is recognized, i.e., how the PB values in the observation phase are obtained. The PB values are calculated based on (2) and (1) by the organized RNNPB model without updating the connection weights. However, there are no arm motor data because the robot is just looking at the target and does not move, unlike in the learning phase. The initial arm motor values are then input to the motion input layer in step 0 and the outputs are calculated forward in the closed-loop mode from step 1; the outputs in the motion output layer in step $t-1$ are the input data in the motion input layer in step t (Fig. 3). To put this simply, the motion input layer plays the same role as the context layer does. In our experiments, the initial arm motor values were constant.

2.3.2. Method for generating motion. This section describes how directive motor values transferred to the robot to move its motors in the motion-generating phase are calculated (Fig. 4). The motion output of the RNNPB model is obtained in a forward

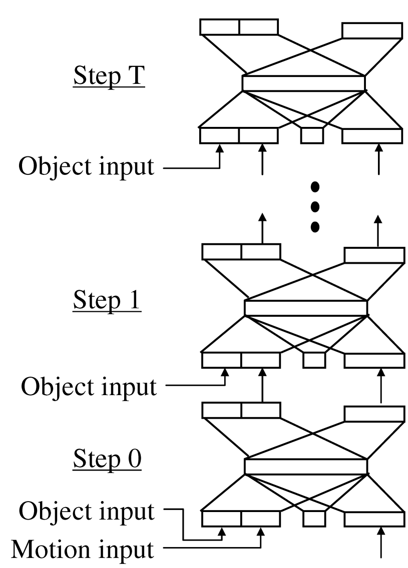


Figure 3. Forward calculation of PB values.

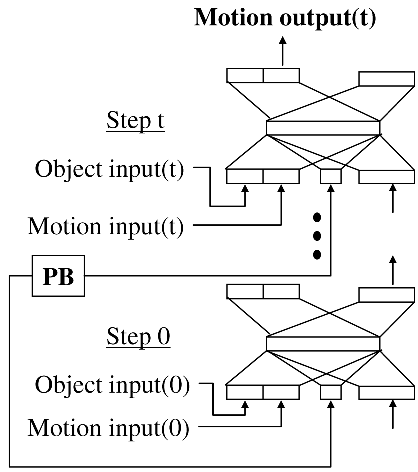


Figure 4. Motion generation.

calculation. The PB values obtained in the observation phase and each item of real input data are input in real-time to the RNNPB in each step. The motion output signal, the predicted directive motor value, of the RNNPB model in step $t-1$ is transferred to the robot as the directive motor value in the next step, step t.

3. MODEL AND SYSTEM

3.1. Humanoid robot Robovie-IIs

Our experimental platform was a humanoid robot, Robovie-IIs, a refined model of Robovie-II developed at ATR [10]. Robovie has 3 d.o.f. in its neck and 4 d.o.f. in each arm. Each motor angle value is measured with potentiometers. It also has stereoscopic CCD cameras on its head. The potentiometers and the camera collected the data required for the experiment.

3.2. Target object

The manipulation targets are a cylinder-shaped object and a box-shaped object. The cylinder-shaped object moves in parallel when the robot lays its hand on the low position and it tumbles when the robot lays the hand on the high position. The box-shaped object was moved by the robot hand. The top of the object is separated into two colors, red and blue, which enable the robot to easily detect the rotation of the object.

3.3. Experimental system

Figure 5 is the system diagram. The camera tracks the target object by controlling the neck motor keeping the centroid of the object centered on the camera. Since the

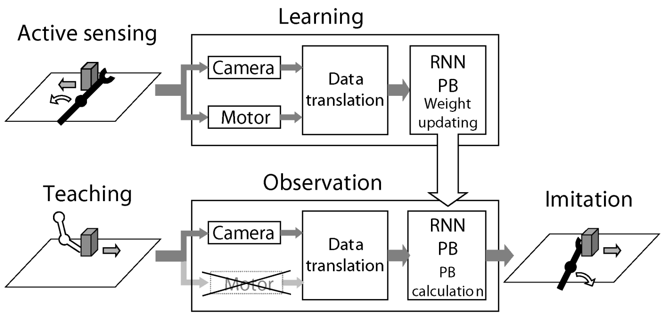


Figure 5. System diagram of imitation.

robot is required to move in real-time, the module for the moving motors has been constituted on a PC embedded in the robot, and the processes for translating data and calculating the directive motor values run on an external PC. Each trial duration was 8 s. The size of the RNNPB model and the dimensionality of the input data differed according to experiments.

The following sensory data were collected in the experiment for use in the RNNPB model.

3.3.1. Visual information. Only the left eye camera was used. The trajectory of an object was selected from the image information by a CCD camera with a resolution of 500 $\times$ 400 pixels. The center position of each colored top face, the X–Y coordinates in the camera ([0–1]), was estimated by extracting the object from the color information.

3.3.2. Motor information. The neck (2 d.o.f.: pitch and yaw axes) and the left arm were used. Note that d.o.f. of used arm motors differed according to experiments and unused motors were fixed.

Those values were synchronized between different modalities and were normalized in [0.1–0.9]. The sensory data were stored every 400 ms for each manipulation and their lengths were all 20 steps.

In the learning phase, the robot first collected the camera data and the motor data from its own neck and arm during active sensing. The connection weights for the RNNPB model were updated off-line using collected data simultaneously. In the observation phase, the robot then collected the neck motor data and the camera data. The corresponding PB values were calculated for the given sequence by the RNNPB model without updating the connection weight values. Finally, in the motion-generating phase, the robot generated its motion by inputting the PB values obtained in the observation phase into the organized RNNPB model.

4. EXPERIMENT

4.1. Imitation of known manipulation

We carried out an experiment to confirm whether the robot can associate its motions only with the object's motions.

4.1.1. Task. A target object is a cylinder-shaped object. In addition to the two neck motors, 2 d.o.f. of the arm motors, i.e., the roll axis of the shoulder and the pitch axis of the elbow, are used. In the experiment, there were two kinds of manipulation, i.e. parallel translation and tumbling (Table 1). Each manipulation has moving directions 'left to right (L→R)' and 'right to left (R→L)'. Learning 1 in Table 1 is, for example, parallel translation from the left to the right. Learning 1–4 in Table 1 represent manipulation that the robot learned in the learning phase. Observation 1–4 in Table 1 represent the manipulation that the robot observed in the observation phase.

4.1.2. Procedure. In the learning phase, the robot first conducted motions programmed to manipulate the object as listed in Table 1, i.e., learning 1–4, and collected sensory data. It manipulated the object 3 times for each learning and collected 12 sets of data. The RNNPB model was then trained with the collected data 300 000 times, which took approximately 40 min. The input layer of the RNNPB model consisted of two units for the object input, four units for the motion input, two units for the parametric bias and 20 units for the context input. The hidden layer consisted of 30 units. The output layer consisted of two units for the object output, four units for the motion output and 20 units for the context output.

In the observation phase, the robot then observed four types of manipulation, Observations 1–4 in Table 1, presented by a human teacher. The robot collected data twice for Observations 1–4; there was a total of eight patterns. With the collected sensory data, the PB values were renewed 5000 times, which took approximately 60 s.

Table 1.
Object manipulation (cylinder)

	Moving direction	Contact position
Learning		
1	L→R	low
2	L→R	high
3	R→L	low
4	R→L	high
Observation		
1	L→R	low
2	L→R	high
3	R→L	low
4	R→L	high

Finally, in the motion-generating phase, the robot generated its motion.

4.1.3. Results. Figure 6 shows examples of sequences of input and output data after the RNNPB model learned in the learning phase. The solid lines describe RNNPB output (prediction) and the broken lines describe input (real data). We confirmed that RNNPB could predict the sequences accurately.

Figure 7 shows two-dimensional PB space acquired in the learning and observation phases, which consisted of pairs of PB values. The PB values obtained in the learning phase were self-organized corresponding to the categories of object manipulations. The PB values resulting from observations of known manipulations are plotted close to the PB values resulting from the same manipulations being learned.

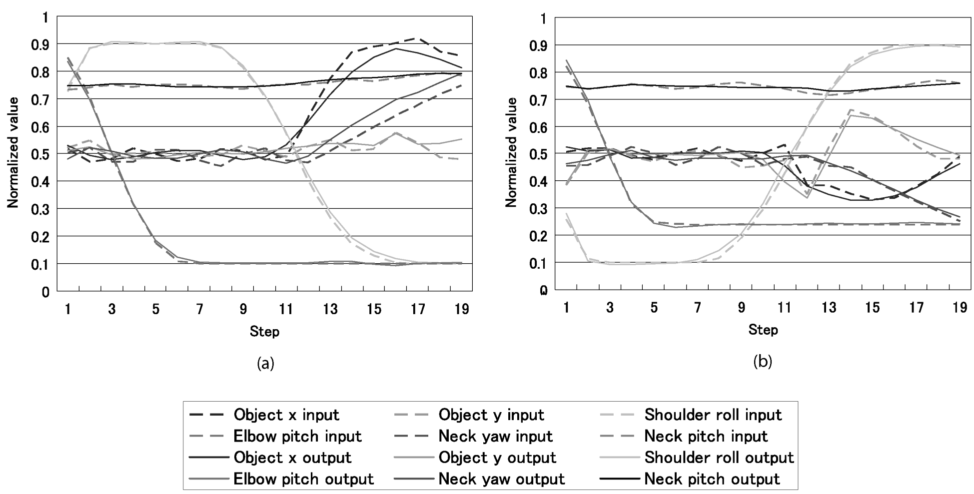


Figure 6. Prediction of time sequences. (a) L→R (low). (b) R→L (high).

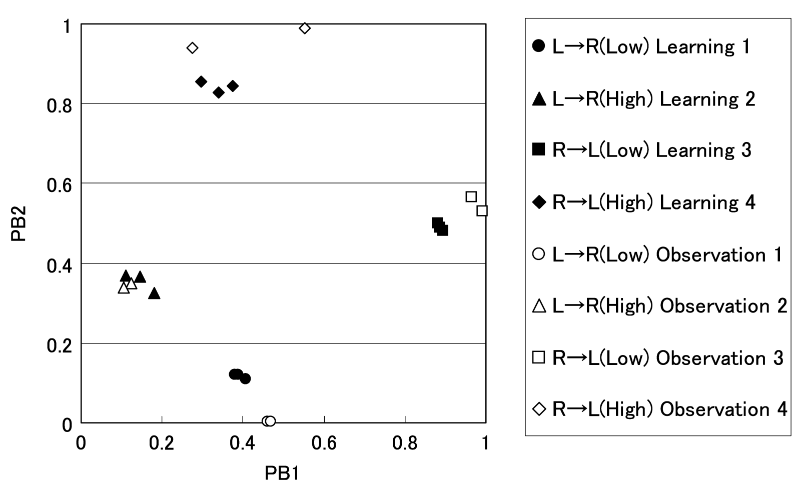


Figure 7. PB space (cylinder).

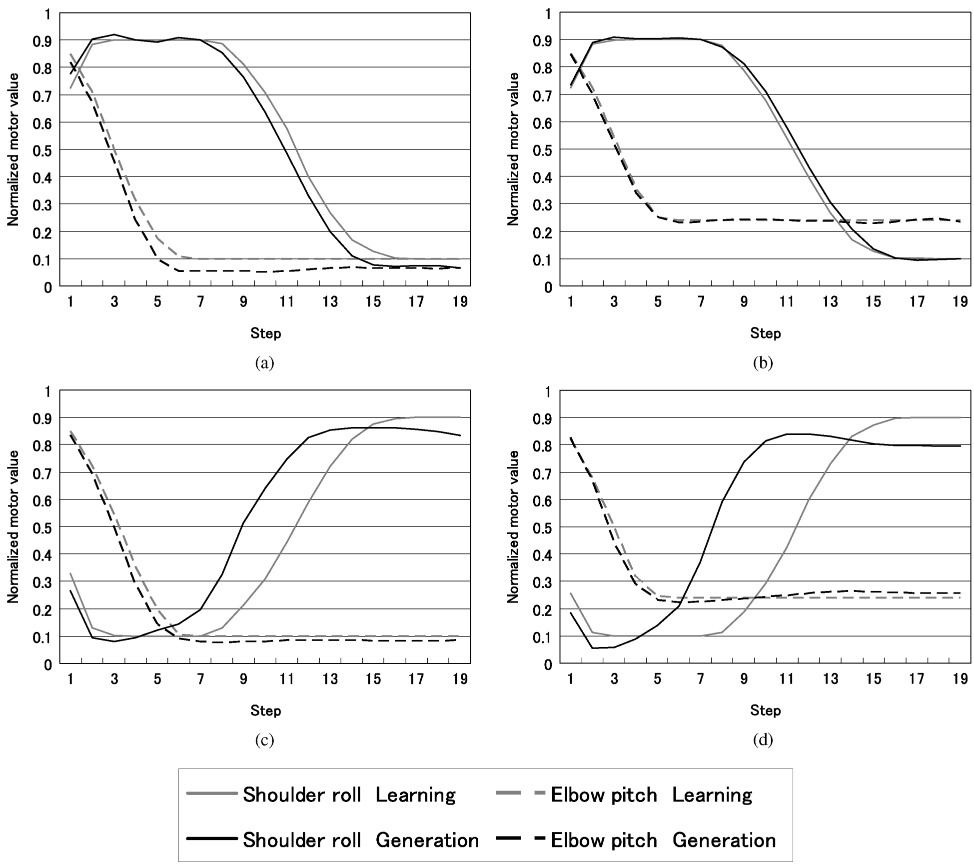


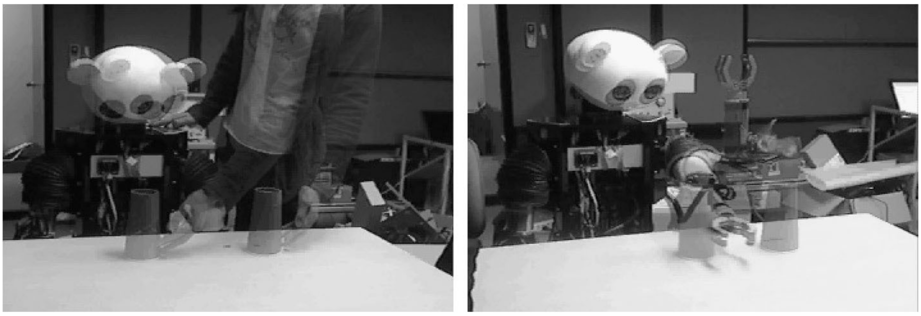
Figure 8. Motor values generated in the motion-generating phase. (a) L→R (low). (b) L→R (high). (c) R→L (low). (d) R→L (high).

Figure 8 shows motor values generated in the learning and motion-generating phases. Learned manipulations were reproduced quite accurately. In the learning phase, occlusion problems occurred when the robot generated Learning 3 and 4 (R→L): its own arm occluded part of the object while manipulating. The occlusions might cause slight inaccuracy of R→L with respect to L→R.

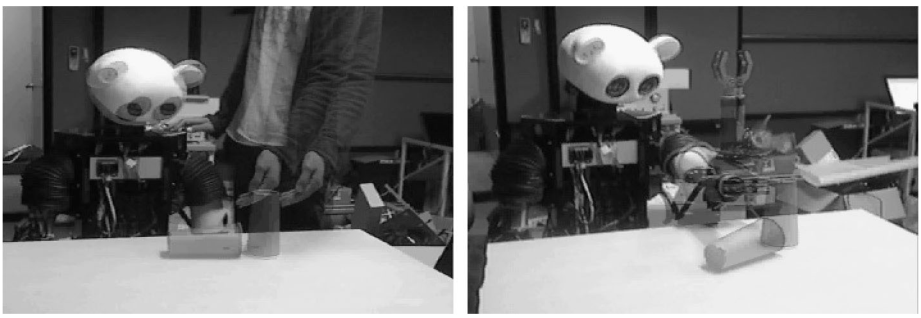
Figure 9 shows examples of sequential photographs that capture the object manipulations. The left photograph captures the human manipulating the object in the observation phase and the right one captures the robot manipulating the object in the motion-generating phase.

4.2. Imitation of unknown manipulation

We carried out another experiment testing some imitation motions involving not only the trained motions in the active sensing process, but also unknown motions.



(a)



(b)

Figure 9. Observation and motion generation (cylinder). (a) L→R (low). (b) L→R (high).

4.2.1. Task. The target object is a box-shaped object. In addition to the two neck motors, 3 d.o.f. of the arm motors, i.e., the roll and yaw axes of the shoulder and the pitch axis of the elbow, are used. In the experiment, there were two kinds of manipulation, i.e., parallel translation from the left to the right ('L→R') and rotation to the right ('Rrot'). Each manipulation was divided into three levels of moving distance: short 'S', medium 'M' and long 'L' (Table 2). Due to several levels being set for each manipulation, we expected that the robot could learn about the gradual shift in its motor value. Learning 1–6 in Table 2 represent manipulation that the robot learned in the learning phase. Observations 1–3 in Table 2 represent the manipulation that the robot observed in the observation phase. Observation 3, which is 'moving from the left to the right while rotating to the right', is manipulation unknown to the robot.

4.2.2. Procedure. In the learning phase, the robot first conducted motions programmed to manipulate the object as listed in Table 2, Learning 1–6, and collected sensory data. It manipulated the object once for each learning and collected six sets of data. The RNNPB model was then trained with the collected data 200 000 times, which took approximately 10 min. The input layer of the RNNPB model consisted of four units for the object input, five units for the motion input, two units for the parametric bias and 10 units for the context input. The

Table 2.
Object manipulation (box)

	Moving direction	Moving level
Learning		
1	L$\rightarrow$R	S
2	L$\rightarrow$R	M
3	L$\rightarrow$R	L
4	Rrot	S
5	Rrot	M
6	Rrot	L
Observation		
1	L$\rightarrow$R	L
2	Rrot	L
3[a]	Rrot + L$\rightarrow$R	L + L

[a] Unknown.

hidden layer consisted of 15 units. The output layer consisted of four units for the object output, five units for the motion output and 10 units for the context output.

In the observation phase, the robot then observed three manipulations, Observations 1–3 in Table 2, presented by a human teacher. The robot collected data once for each manipulation; there was a total of three patterns. With the collected sensory data, the PB values were renewed 5000 times, which took approximately 15 s.

Finally, in the motion-generating phase, the robot generated its motion.

4.2.3. Results. Figure 10 shows the PB space acquired in the learning and observation phases. The PB values obtained in the learning phase were self-organized corresponding to the categories of object manipulations and moving levels. The PB values resulting from observations of known manipulations are plotted close to the PB values resulting from the same manipulations being learned. However, PB values corresponding to the unknown manipulation labeled with an asterisk are plotted to the center position between ‘L$\rightarrow$R’ and ‘Rrot’.

Figure 11 plots the trajectories for the robot’s hand seen from above the table in the learning and motion-generating phases. Figure 12 presents sequential photographs that capture the object manipulations. The left photograph captures the human manipulating the object in the observation phase and the right one captures the robot manipulating the object in the motion-generating phase. The unknown manipulation was imitated as a combination of known manipulations.

5. DISCUSSION

5.1. Prediction capability

As can be seen from Fig. 6, the RNNPB model has prediction capabilities. The robot can predict what kind of object’s motion its own motion would generate. This

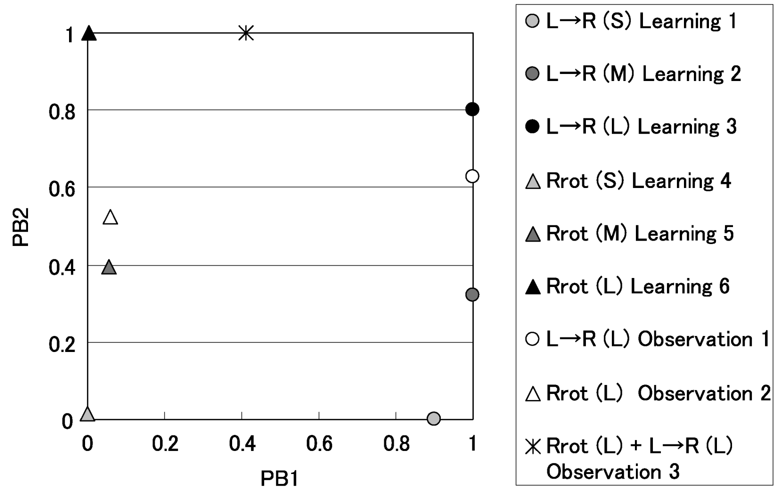


Figure 10. PB space (box).

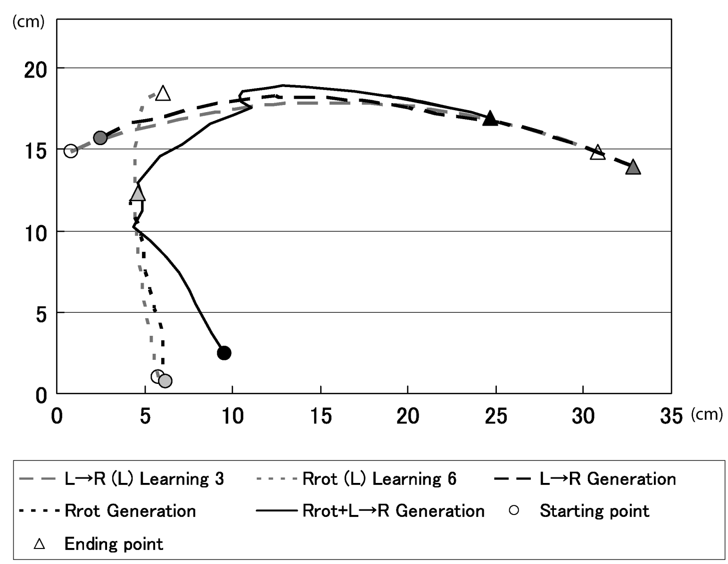
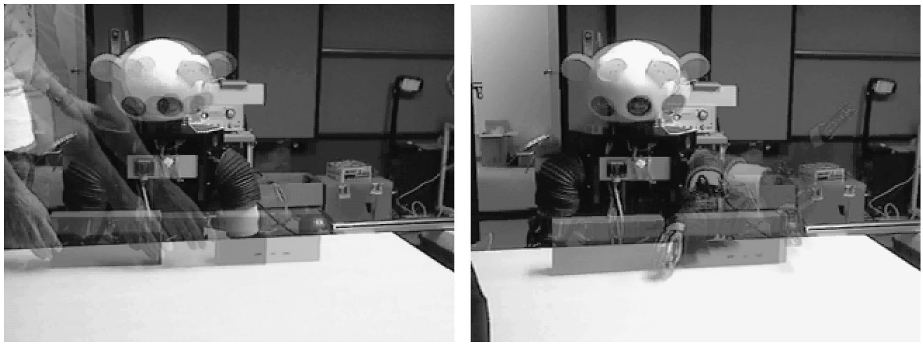
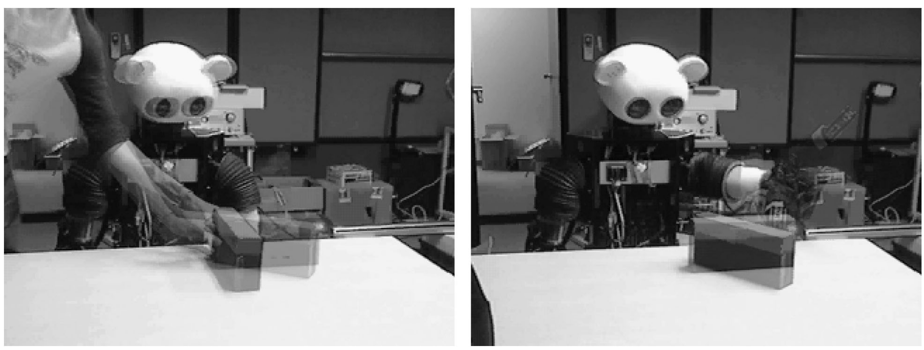


Figure 11. Robot hand trajectory.

enables the robot to associate motion with an object's motion in the observation phase. In recognizing observed manipulation, the robot predicts the motion and object sequence, and obtains the PB values that generate appropriate motion. In the motion-generating phase, the robot predicts the sequence in real-time and selects motion for the next step with the RNNPB model.



(a)



(b)

(c)

Figure 12. Observation and motion generation (box). (a) L→R (left to right). (b) Rrot (right rotation). (c) Rrot + L→R (right rotation + left to right).

5.2. Generalization capability

The robot acquired behavioral primitives implicitly through learning in the second experiment: moving the hand from the left to the right for manipulation 'L→R' and extending its arm for manipulation 'Rrot'. The unknown manipulation was recognized as a combination of the primitives. This clearly proved the generalization capabilities of the proposed method.

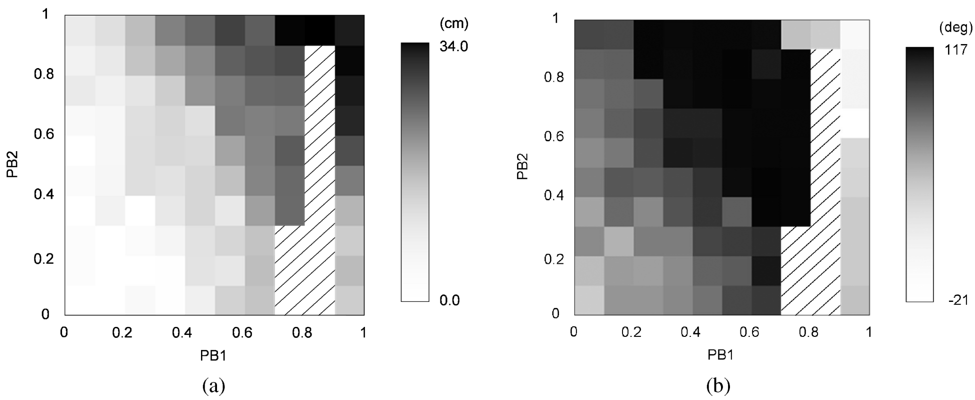


Figure 13. Analysis of PB space. (a) Moving distance. (b) Rotation angle.

5.3. Analysis of PB space

To see how the learned motions are generalized, we analyzed the PB space acquired in the experiment using the box-shaped object (Fig. 10). A total of 100 varieties of PB values were input into the RNNPB: each PB value ranged from 0.05 to 0.95. The robot generated 100 types of motions and the trajectories of the object for each manipulation were analyzed. Figure 13a and 13b shows how the PB space was self-organized upon moving distance of the object and its rotation angle, respectively. In Fig. 13b, a positive number represents that the object was rotated to the right and a negative number represents that it was rotated to the left. The hatched area on the PB space represents that its PB values generated meaningless motions such as hitting the top of the object.

The PB values on the upper half of the middle area generated motions which are combinations of known manipulations, such as parallel translation after rotation. Not all combinations of the moving distance and the rotation angle could be generated, of course, but a number of unknown combinations were generated based only on the six learned patterns. As can be seen from Fig. 13, some parts of the PB space have nonlinear properties. This result implies that the relation between the PB values and the characteristics of the generated object's motions is nonlinear. A similar result was also discussed by Tani *et al.* using the RNNPB for the mapping of PB values and robot behaviors [4].

6. CONCLUSIONS

This paper proposed a method of imitation focusing on object's motion generated while a humanoid robot was actively sensing objects. The task was moving objects on a table, the first step in object manipulation. The method consists of three phases, i.e., the learning, observation and motion-generating phases. The RNNPB model, which has generalization capabilities, was used as the learning model to reduce the learning load. By specifically taking advantage of the RNNPB model, the robot self-organized connection between its own arm motions, and the object's motions, and associated a motion with an observed object's motions. A learning system that gathered visual data and motor data during manipulations was implemented on the humanoid robot Robovie-IIs. An experiment using a cylinder-shaped object and an experiment using a box-shaped object were conducted. The first experiment demonstrated that the robot could associate its motions only with the object's motions. The second experiment demonstrated that this method enabled the robot to imitate the unknown manipulation through nonlinear combinations of the experienced manipulations.

Although the task set for the experiment was object manipulation, our method can be used for different tasks. Our method plays the role of connecting the actor's operation and the target response. If the target is a part of a body, it is also possible for robots to imitate body motions.

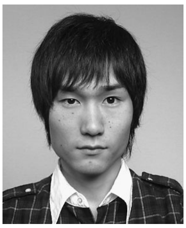
In this work, experiments were conducted with limited and few learning patterns. Acquiring a greater variety of motions requires resolution of the trade-off between generalization and differentiation of motion dynamics. Our future work will confirm the general effectiveness of the method for a variety of motions and resolve the issue stated above to develop a sophisticated method which enables robots to generate more motion patterns.

REFERENCES

1. S. Schaal, Is imitation learning the route to humanoid robots?, *Trends Cognitive Sci.* **3**, 233–242 (1999).
2. A. Nakazawa, S. Nakaoka, K. Ikeuchi and K. Yokoi, Imitating human dance motions through motion structure analysis, in: *Proc. Int. Workshop on Intelligent Robots and Systems*, Lausanne, pp. 2539–2544 (2002).
3. G. Rizzolatti and L. Craighero, The mirror-neuron system, *Ann. Rev. Neurosci.* **27**, 169–192 (2004).
4. J. Tani and M. Ito, Self-organization of behavioral primitives as multiple attractor dynamics: a robot experiment, *IEEE Trans. Syst. Man Cybernet. A* **33**, 481–488 (2003).
5. R. Rao, A. Shon and A. Meltzoff, A Bayesian model of imitation in infants and robots, in: *Imitation and Social Learning in Robots, Humans, and Animals: Behavioural, Social and Communicative Dimensions*, K. Dautenhahn and C. Nehaniv (Eds), pp. 217–247. Cambridge University Press, Cambridge (2005).
6. T. Ogata, H. Ohba, J. Tani, K. Komatani and H. G. Okuno, Extracting multi-modal motions of objects using RNNPB, in: *Proc. Int. Workshop on Intelligent Robots and Systems*, Edmonton, pp. 160–165 (2005).
7. J. Tani, Learning to generate articulated behavior by the 'forwarding forward model' network, IEICE Technical Report, *Neurocomputing* **101** (NC2001-88), 49–55 (2002).
8. D. Rumelhart, G. Hinton and R. Williams, Learning internal representation by error propagation, in: *Parallel Distributed Processing*, D. E. Rumelhart and J. L. McLelland (Eds), vol. 1, pp. 318–362. MIT Press, Cambridge, MA (1986).
9. T. Ogata, S. Sugano and J. Tani, Acquisition of motion primitives of robot in human-navigation task: towards human–robot interaction based on 'quasi-symbols', *J. Jpn. Soc. AI* **20**, 188–196 (2005).

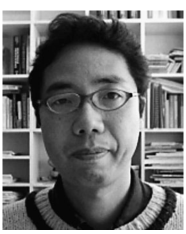
10. H. Ishiguro, T. Ono, M. Imai, T. Maeda, T. Kanda and R. Nakatsu, Robovie: an interactive humanoid robot, *Int. J. Ind. Robotics* **28**, 498–503 (2001).

ABOUT THE AUTHORS



Ryunosuke Yokoya received the BE degree from the Department of Information and Mathematical Science, Kyoto University, in 2006. He is currently a Master course student at the Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University. He received IPSJ 68th National Convention Student Award in 2006, and is a Student Member of the IPSJ, RSJ and IEEE.

Tetsuya Ogata received the BS, MS and DE in Mechanical Engineering in 1993, 1995 and 2000, respectively, from Waseda University. From 1999 to 2001, he was a Research Associate in Waseda University. From 2001 to 2003, he was a Research Scientist in the Brain Science Institute, RIKEN. Since 2003, he has been a Faculty Member in the Graduate School of Informatics, Kyoto University, where he is currently an Associate Professor. Since 2005, he has been a Visiting Associate Professor of the Humanoid Robotics Institute of Waseda University. His research interests include human–robot interaction, dynamics of human–robot mutual adaptation and inter-sensory translation in robot systems. He received the JSME Medal for the Outstanding Paper from the Japan Society of Mechanical Engineers in 2000.



Jun Tani received the BS degree in Mechanical Engineering from Waseda University, the MS degree in Electrical Engineering from Michigan University and the DE from Sophia University. He is currently a Team Leader in the Laboratory for Behavior and Dynamic Cognition, Brain Science Institute, RIKEN. His research interests include embodied cognition, complex adaptive systems and brain science.

Kazunori Komatani received the BE, MS and PhD degrees from Kyoto University, Japan, in 1998, 2000 and 2002, respectively. He is currently an Assistant Professor of the Graduate School of Informatics, Kyoto University. His research interests include spoken dialogue systems.

Hiroshi G. Okuno received the BA and PhD degrees from the University of Tokyo, in 1972 and 1996, respectively. He is currently a Professor of the Graduate School of Informatics, Kyoto University. He has received various awards including the Best Paper Awards of JSAI. His research interests include computational auditory scene analysis, music scene analysis and robot audition.