21

Obtaining Humanoid Robot Controller Using Reinforcement Learning

Masayoshi Kanoh[1] and Hidenori Itoh[2]
[1]*Chukyo University,* [2]*Nagoya Institute of Technology*
Japan

1. Introduction

Demand for robots is shifting from their use in industrial applications to their use in domestic situations, where they "live" and interact with humans. Such robots require sophisticated body designs and interfaces to do this. Humanoid robots that have multi-degrees-of-freedom (MDOF) have been developed, and they are capable of working with humans using a body design similar to humans. However, it is very difficult to intricately control robots with human generated, preprogrammed, learned behavior. Learned behavior should be acquired by the robots themselves in a human-like way, not programmed manually. Humans learn actions by trial and error or by emulating someone else's actions. We therefore apply reinforcement learning for the control of humanoid robots because this process resembles a human's trial and error learning process.

Many existing methods of reinforcement learning for control tasks involve discrediting state space using BOXES (Michie & Chambers, 1968; Sutton & Barto, 1998) or CMAC (Albus, 1981) to approximate a value function that specifies what is advantageous in the long run. However, these methods are not effective for doing generalization and cause perceptual aliasing. Other methods use basis function networks for treating continuous state space and actions.

Networks with sigmoid functions have the problem of catastrophic interference. They are suitable for off-line learning, but are not adequate for on-line learning such as that needed for learning motion (Boyan & Moore, 1995; Schaal & Atkeson, 1996). On the contrary, networks with radial basis functions are suitable for on-line learning. However, learning using these functions requires a large number of units in the hidden layer, because they cannot ensure sufficient generalization. To avoid this problem, methods of incremental allocation of basis functions and adaptive state space formation were proposed (Morimoto & Doya, 1998; Samejima & Omori, 1998; Takahashi et al., 1996; Moore & Atkeson, 1995).

In this chapter, we propose a dynamic allocation method of basis functions called Allocation/Elimination Gaussian Softmax Basis Function Network (AE-GSBFN), that is used in reinforcement learning to treat continuous high-dimensional state spaces. AE-GSBFN is a kind of actor-critic method that uses basis functions and it has allocation and elimination processes. In this method, if a basis function is required for learning, it is allocated dynamically. On the other hand, if an allocated basis function becomes redundant, the function is eliminated. This method can treat continuous high-dimensional state spaces

Source: Humanoid Robots, New Developments, Book edited by: Armando Carlos de Pina Filho
ISBN 978-3-902613-02-8, pp.582, I-Tech, Vienna, Austria, June 2007

because the allocation and elimination processes reduce the number of basis functions required for evaluation of the state space.

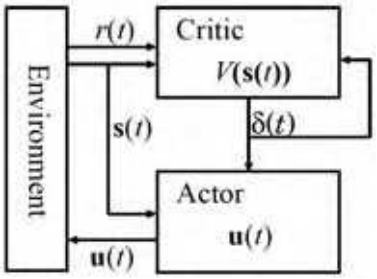


Fig. 1. Actor-critic architecture.

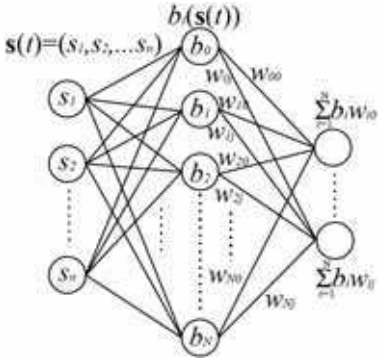


Fig. 2. Basis function network.

To confirm the effectiveness of our method, we used computer simulation to show how a humanoid robot learned two motions: a standing-up motion from a seated position on chair and a foot-stamping motion.

2. Actor-Critic Method

In this section, we describe an actor-critic method using basis functions, and we apply it to our method.

Actor-critic methods are temporal difference (TD) methods that have a separate memory structure to explicitly represent the policy independent of the value function (Sutton & Barto, 1998). Actor-critic methods are constructed by an actor and a critic, as depicted in Figure 1. The policy structure is known as the actor because it is used to select actions, and the estimated value function is known as the critic because it criticizes the actions made by the actor.

The actor and the critic each have a basis function network for learning of continuous state spaces. Basis function networks have a three-layer structure as shown in Figure 2, and basis functions are placed in middle-layer units. Repeating the following procedure, in an actor-critic method using basis function networks, the critic correctly estimates the value function $V(\mathbf{s})$, and then the actor acquires actions that maximize $V(\mathbf{s})$.

1. When state $\mathbf{s}(t)$ is observed in the environment, the actor calculates the j-th value $u_j(t)$ of the action $\boldsymbol{u}(t)$ as follows (Gullapalli, 1990):

$$u_j(t) = u_j^{\max} g\left(\sum_i^N \omega_{ij} b_i(\mathbf{s}(t)) + n_j(t) \right), \quad (1)$$

where $u_j^{\max}$ is a maximal control value, N is the number of basis functions, $b_i(\mathbf{s}(t))$ is a basis function, ω_{ij} is a weight, $n_j(t)$ is a noise function, and $g()$ is a logistic sigmoid activation function whose outputs lie in the range (−1, 1). The output value of actions is saturated into $u_j^{\max}$ by $g()$.

2. The critic receives the reward $r(t)$, and then observes the resulting next state $\mathbf{s}(t+1)$. The critic provides the TD-error $\delta(t)$ as follows:

$$\delta(t) = r(t) + \gamma V(\mathbf{s}(t+1)) - V(\mathbf{s}(t)), \quad (2)$$

where γ is a discount factor, and $V(\mathbf{s})$ is an estimated value function. Here, $V(\mathbf{s}(t))$ is calculated as follows:

$$V(\mathbf{s}(t)) = \sum_i^N v_i b_i(\mathbf{s}(t)), \quad (3)$$

where v_i is a weight.

3. The actor updates weight ω_{ij} using TD-error:

$$\omega_{ij} \leftarrow \omega_{ij} + \beta \delta(t) n_j(t) b_i(\mathbf{s}(t)), \quad (4)$$

where β is a learning rate.

4. The critic updates weight v_i:

$$v_i \leftarrow v_i + \alpha \delta(t) e_i, \quad (5)$$

where α is a learning rate, and e_i is an eligibility trace. Here, e_i is calculated as follows:

$$e_i \leftarrow \gamma \lambda e_i + b(\mathbf{s}(t)), \quad (6)$$

where λ is a trace-decay parameter.

5. Time is updated.

$$t \leftarrow t + \Delta t. \quad (7)$$

Note that Δt is 1 in general, but we used the description of Δt for the control interval of the humanoid robots.

3. Dynamic Allocation of Basis Functions

In this chapter, we propose a dynamic allocation method of basis functions. This method is an extended application of the Adaptive Gaussian Softmax Basis Function Network (A-GSBFN) (Morimoto & Doya, 1998, 1999). A-GSBFN only allocates basis functions, whereas our method both allocates and eliminates them. In this section, we first briefly describe A-GSBFN in Section 3.1; then we propose our method, Allocation/Elimination Gaussian Softmax Basis Function Network (AE-GSBFN), in Section 3.2.

3.1 A-GSBFN

Networks with sigmoid functions have the problem of catastrophic interference. They are suitable for off-line learning, but not adequate for on-line learning. In contrast, networks with radial basis functions (Figure 3) are suitable for on-line learning, but learning using these functions requires a large number of units, because they cannot ensure sufficient generalization. The Gaussian softmax functions (Figure 4) have the features of both sigmoid

functions and radial basis functions. Networks with the Gaussian softmax functions can therefore assess state space locally and globally, and enable learning motions of humanoid robots.

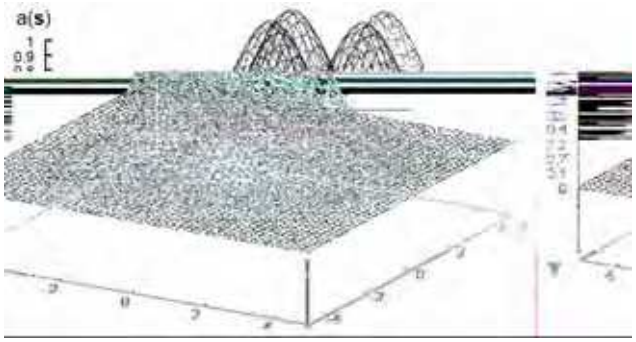


Fig. 3. Shape of radial basis functions. Four radial basis functions are visible here, but it is clear that the amount of generalization done is insufficient.

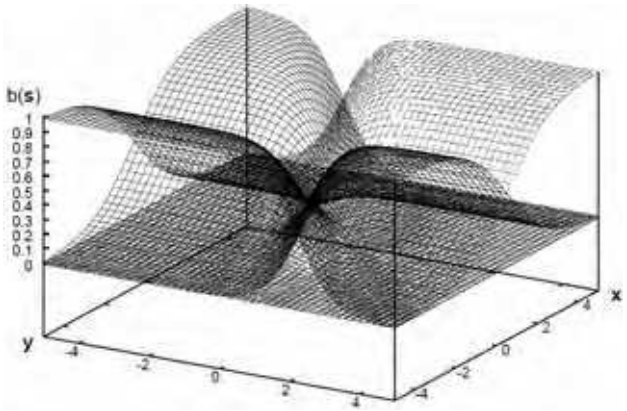


Fig. 4. Shape of Gaussian softmax basis functions. Similar to Figure 3, there are four basis functions. Using Gaussian softmax basis functions, global generalization is done, such as using sigmoid functions.

The Gaussian softmax basis function is used in A-GSBFN and is given by the following equation:

$$b_i(\mathbf{s}(t)) = \frac{a_i(\mathbf{s}(t))}{\sum_k^N a_k(\mathbf{s}(t))}, \tag{8}$$

where $a_i(\mathbf{s}(t))$ is a radial basis function, and N is the number of radial basis functions. Radial basis function $a_i(\mathbf{s}(t))$ in the i-th unit is calculated by the following equation:

$$a_i(\mathbf{s}(t)) = \exp\left(-\frac{1}{2}\left\|M(\mathbf{s}(t)-\mathbf{c}_i)\right\|^2\right), \tag{9}$$

where $\mathbf{c}_i$ is the center of the i-th basis function, and M is a matrix that determines the shape of the basis function.

In A-GSBFN, a new unit is allocated if the error is larger than threshold $\delta_{\max}$ and the activation of all existing units is smaller than threshold $a_{\min}$:

$$\left|h(t)\right| > \delta_{\max} \quad \text{and} \quad \max_i a_i(\mathbf{s}(t)) < a_{\min}, \tag{10}$$

where $h(t)$ is defined as $h(t) = \delta(t) n_j(t)$ at the actor, and $h(t) = \delta(t)$ at the critic. The new unit is initialized with $\mathbf{c}_i = \mathbf{s}$, and $\omega_i = 0$.

3.2 Allocation/Elimination GSBFN

To perform allocation and elimination of basis functions, we introduce three criteria into A-GSBFN: trace ε_i of activation of radial basis functions, additional control time η, and existing time τ_i of radial basis functions. The criteria ε_i and τ_i are prepared for all basis functions, and η is prepared for both actor and critic networks. A learning agent can gather further information on its own states by using these criteria.
We now define the condition of allocation of basis functions.

Definition 1 - Allocation
A new unit is allocated at $\mathbf{c}_i = \mathbf{s}(t)$ if the following condition is satisfied at the actor or critic networks:

$$\left|h(t)\right| > \delta_{\max} \quad \text{and} \quad \max_i a_i(\mathbf{s}(t)) < a_{\min} \quad \text{and} \quad \eta > T_{\text{add}}, \tag{11}$$

where T_{add} is a threshold.

■

Let us consider using condition (10) for allocation. This condition is only considered for allocation, but it is not considered as a process after a function is eliminated. Therefore, when a basis function is eliminated, another basis function is immediately allocated at the near state of the eliminated function. To prevent immediate allocation, we introduced additional control time η into the condition of allocation. The value of η monitors the length of time that has elapsed since a basis function was eliminated. Note that η is initialized at 0, when a basis function is eliminated.
We then define the condition of elimination using ε_i and τ_i.

Definition 2 - Elimination
The basis function $b_i(\mathbf{s}(t))$ is eliminated if the following condition is satisfied in the actor or critic networks.

$$\varepsilon_i > \varepsilon_{\max} \quad \text{and} \quad \tau_i > T_{\text{erase}}, \tag{12}$$

where $\varepsilon_{\max}$ and T_{erase} are thresholds.

□

The trace ε_i of the activation of radial basis functions is updated at each step in the following manner:

$$\varepsilon_i \leftarrow \kappa \varepsilon_i + a_i(\mathbf{s}(t)), \tag{13}$$

where κ is a discount rate. Using ε_i, the learning agent can sense states that it has recently taken. The value of ε_i takes a high value if the agent stays in almost the same state. This

situation is assumed when the learning falls into a local minimum. Using the value of ε_i, we consider how to avoid the local minimum. Moreover, using τ_i, we consider how to inhibit a basis function from immediate elimination after it is allocated. We therefore defined the condition of elimination using ε_i and τ_i.

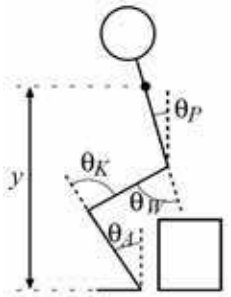


Fig. 5. Learning motion; standing up from a chair.

4. Experiments

4.1 Standing-up motion learning

In this section, as an example of learning of continuous high-dimensional state spaces, AE-GSBFN is applied to a humanoid robot learning to stand up from a chair (Figure 5). The learning was simulated using the virtual body of the humanoid robot HOAP-1 made by Fujitsu Automation Ltd. Figure 6 shows HOAP-1. The robot is 48 centimeters tall, weighs 6 kilograms, has 20 DOFs, and has 4 pressure sensors each on the soles of its feet. Additionally, angular rate and acceleration sensors are mounted in its chest. To simulate learning, we used the Open Dynamics Engine (Smith).

Fig. 6. HOAP-1 (Humanoid for Open Architecture Platform).

The robot is able to observe the following vector $\mathbf{s}(t)$ as its own state:

$$\mathbf{s}(t) = \left(\theta_W, \dot{\theta}_W, \theta_K, \dot{\theta}_K, \theta_A, \dot{\theta}_A, \theta_P, \dot{\theta}_P\right), \tag{14}$$

where θ_W, θ_K, and θ_A are waist, knee, and ankle angles respectively, and θ_P is the pitch of its body (see Figure 5). Action $\mathbf{u}(t)$ of the robot is determined as follows:

$$\mathbf{u}(t) = \left(\dot{\theta}_W, \dot{\theta}_K, \dot{\theta}_A\right), \tag{15}$$

One trial ended when the robot fell down or time exceeded t_{total} = 10 [s]. Rewards $r(t)$ were determined by height y [cm] of the robot's chest:

$$r(t) = \begin{cases} -20\left|\dfrac{l_{\text{stand}} - y}{l_{\text{stand}} - l_{\text{down}}}\right| & (\textit{during trial}) \\ -20\left|t_{\text{total}} - t\right| & (\textit{on failure}) \end{cases}, \tag{16}$$

where l_{stand} = 35 [cm] is the position of the robot's chest in an upright posture, and l_{down} = 20 [cm] is its center in a falling-down posture. We used $u_j^{\max} = \pi / 36$, $\gamma = 0.9$, $\beta = 0.1$, $\alpha = 0.02$, $\lambda = 0.6$, and $\Delta t = 0.01$ [s] for parameters in Section 2, M=(1.0, 0.57, 1.0, 0.57, 1.0, 0.57, 1.0, 0.57), $\delta_{\max} = 0.5$, and $a_{\min} = 0.4$ in Section 3.1, and $T_{\text{add}} = 1$ [s], $\kappa = 0.9$, $\varepsilon_{\max} = 5.0$, and $T_{\text{erase}} = 3$ [s] in Section 3.2.

Figure 7 shows the learning results. First, the robot learned to fall down backward, as shown in i). Second, the robot intended to stand up from a chair, but fell forward, as shown in ii), because it could not yet fully control its balance. Finally, the robot stood up while maintaining its balance, as shown in iii). The number of basis functions in the 2922nd trial was 72 in both actor and critic networks. Figure 8 shows the experimental result with the humanoid robot HOAP-1. The result shows that HOAP-1 was able to stand up from a chair, as in the simulation.

We then compared the number of basis functions in AE-GSBFN with the number of basis functions in A-GSBFN. Figure 9 shows the number of basis functions of the actor, averaged over 20 repetitions. In these experiments, motion learning with both AE-GSBFN and A-GSBFN was successful, but the figure indicates that the number of basis functions required by AE-GSBFN was fewer than that by A-GSBFN. That is, high dimensional learning is possible using AE-GSBFN. Finally, we plotted the height of the robot's chest in successful experiments in Figures 10 and 11. In the figures, circles denote a successful stand-up. The results show that motion learning with both AE-GSBFN and A-GSBFN was successful.

4.2 Stamping motion learning

In Section 4.1, we described our experiment with learning of transitional motion. In this section, we describe our experiment with periodic motion learning. We use a stamping motion as a periodic motion (Figure 12). Periodic motions, such as locomotion, are difficult to learn only through reinforcement learning, so in many cases, a Central Pattern Generator (CPG), etc., is used in addition to reinforcement learning (e.g., Mori et al., 2004). In this experiment, we use inverse kinematics and AE-GSBFN to obtain a stamping motion.

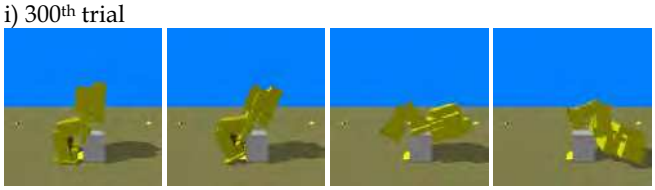
Inverse kinematics calculates the amount $\dot{\theta}$ of change of each joint angle from the amount $\dot{P}$ of change of the coordinates of a link model:

$$\dot{\theta} = J^{-1}(\theta)\dot{P} , \tag{17}$$

where $J(\theta)$ is the Jacobian matrix. Generally, since the dimension of $\dot{\theta}$ differs from the dimension of $\dot{P}$, $J(\theta)$ does not become a regular matrix, and its inverse matrix cannot be calculated. Moreover, even if it could be calculated, a motion resolution by $J^{-1}(\theta)$ cannot be performed in the neighborhood of singular points, which are given by θ around $\det J(\theta) = 0$. To solve these problems, we used the following function (Nakamura & Hanafusa, 1984) in this section:

$$\dot{\theta} = J^T (JJ^T + k_s I)^{-1} \dot{P} , \tag{18}$$

i) 300th trial



ii) 1031st trial

iii) 1564th trial

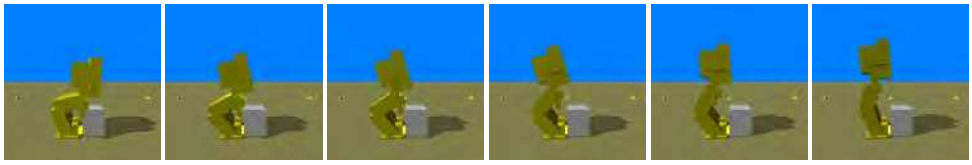


Fig. 7. Learning results.

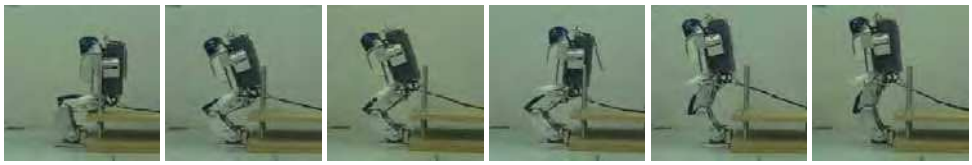


Fig. 8. Experimental result with HOAP-1.

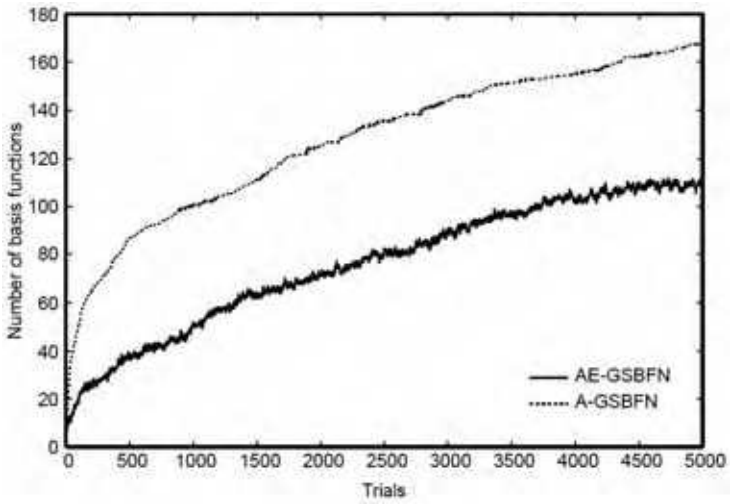


Fig. 9. Number of basis functions in the actor network (averaged over 20 repetitions).

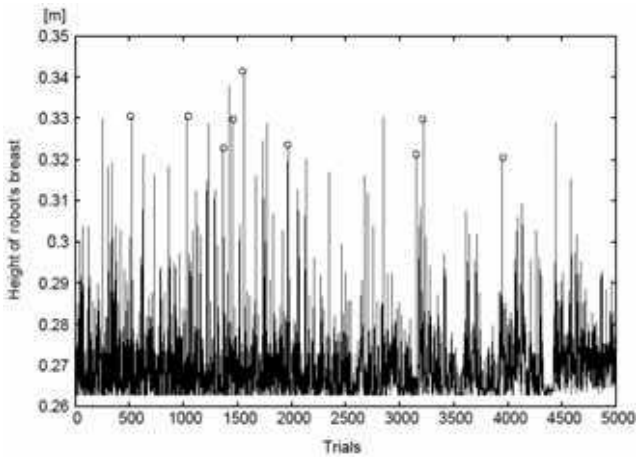


Fig. 10. Height of robot's chest with AE-GSBFN. Circles denote successful stand-ups.

where k_s is a scalar function with which it becomes a positive value near singular points and becomes 0 otherwise:

$$k_s = \begin{cases} k_0\left(1-\frac{w}{w_0}\right)^2 & (w < w_0) \\ 0 & (otherwise) \end{cases}, \tag{19}$$

where k_0 is a positive parameter, w_0 is a threshold that divides around singular points from the others, and w is given by $w = \sqrt{\det J(\theta) J^T(\theta)}$.

In this experiment, the coordinate of the end of the legs is given by inverse kinematics (i.e., up-down motion of the legs is given), and motion of the horizontal direction of the waist is learned by AE-GSBFN. The coordinate value was acquired from the locomotion data of HOAP-1. Concretely, motion is generated by solving inverse kinematics from p_w to the idling leg, and from the supporting leg to p_w (Figure 12 (a)). The change of supporting and idling legs is also acquired from HOAP-1's locomotion data.

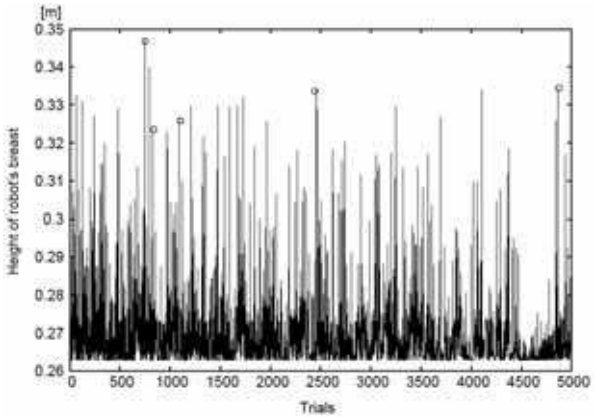


Fig. 11. Height of robot's chest with A-GSBFN. Circles denote successful stand-ups.

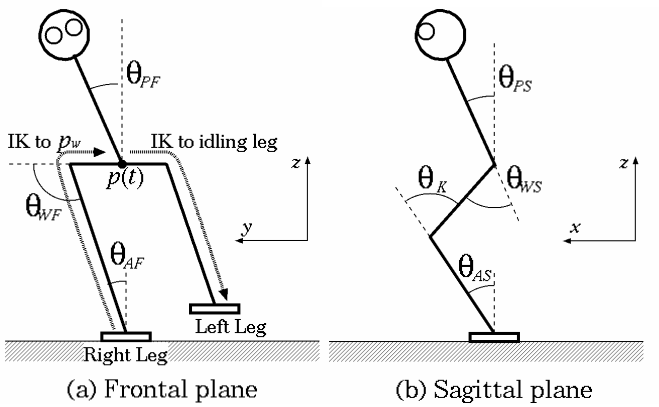


Fig. 12. Learning motion; stamping the foot.

The robot is able to observe the following vector $\mathbf{s}(t)$ as its own state:

$$\begin{aligned}\mathbf{s}(t) = (&\theta_{WF}^{(R)}, \dot{\theta}_{WF}^{(R)}, \theta_{WF}^{(L)}, \dot{\theta}_{WF}^{(L)}, \theta_{AF}^{(R)}, \dot{\theta}_{AF}^{(R)}, \theta_{AF}^{(L)}, \dot{\theta}_{AF}^{(L)}, \theta_{PF}, \dot{\theta}_{PF}, \\ &\theta_{WS}^{(R)}, \dot{\theta}_{WS}^{(R)}, \theta_{WS}^{(L)}, \dot{\theta}_{WS}^{(L)}, \theta_{K}^{(R)}, \dot{\theta}_{K}^{(R)}, \theta_{K}^{(L)}, \dot{\theta}_{K}^{(L)}, \theta_{PS}, \dot{\theta}_{PS}),\end{aligned} \tag{20}$$

where $\theta_{WF}^{(\cdot)}$ (right leg: $\theta_{WF}^{(R)}$, left leg: $\theta_{WF}^{(L)}$) and $\theta_{AF}^{(\cdot)}$ (right leg: $\theta_{AF}^{(R)}$, left leg: $\theta_{AF}^{(L)}$) are angles of the waist and ankle about the roll axis, respectively, and θ_{PF} is the pitch of its body about the roll axis. Also $\theta_{WS}^{(\cdot)}$ (right leg: $\theta_{WS}^{(R)}$, left leg: $\theta_{WS}^{(L)}$) and $\theta_{K}^{(\cdot)}$ (right leg: $\theta_{K}^{(R)}$, left leg: $\theta_{K}^{(L)}$) are angles of the waist and knee about the pitch axis, respectively, and θ_{PS} is the pitch of its body about the pitch axis. Note that the angle of the ankle of each leg about the pitch axis was controlled to be parallel to the ground.

Action $\mathbf{u}(t)$ of the robot is determined as follows:

$$\mathbf{u}(t) = \dot{p}(t), \tag{21}$$

where $\dot{p}(t)$ is the amount of change of $p(t)$ which is the position of the center of the robot's waist. Note that the value of $p(t)$ is a y-coordinate value, and does not include x- or z-coordinate values.

One trial terminated when the robot fell down or time exceeded t_{total} = 17.9 [s]. Rewards $r(t)$ were determined by the following equation:

$$r(t) = \begin{cases} -20\left|p(t) - p(0)\right| & (\textit{during trial}) \\ -20\left|t_{\text{total}} - t\right| & (\textit{on failure}) \end{cases}. \tag{22}$$

We can use the value of the difference between its supporting leg and $p(t)$ as rewards, but these rewards may represent the ideal position of $p(t)$ because of the use of inverse kinematics. Therefore, we used the above equation. Using the equation (22), the closer $p(t)$ is to $p(0)$, the more the rewards increases. Intuitively, it is unsuitable for rewards of stamping motion learning, but acquiring a stamping motion only brings more rewards, because an up-down motion of the leg is given forcibly by inverse kinematics, and it is necessary to change $p(t)$ quite a lot to make the robot stay upright without falling down.

We used $u_j^{\max} = 1.0 \times 10^{-4}$, $\gamma = 0.9$, $\beta = 0.1$, $\alpha = 0.02$, $\lambda = 0.6$, and $\Delta t = 0.01$ [s] for parameters in Section 2, M=diag(2.0, 1.1, 2.0, 1.1, 2.0, 1.1, 2.0, 1.1, 2.0, 1.1, 2.0, 1.1, 2.0, 1.1, 2.0, 1.1, 2.0, 1.1), $\delta_{\max} = 0.5$, and $a_{\min} = 0.135$ in Section 3.1, and $T_{\text{add}} = 1.0$ [s], $\kappa = 0.9$, $\varepsilon_{\max} = 5.0$, and $T_{\text{erase}} = 2.0$ [s] in Section 3.2. We also used k_0 = 0.01 and w_0=0.003 for the parameters of inverse kinematics.

Figure 13 shows the learning results. The robot can control its balance by moving its waist right and left. Figure 14 plots the amount of time taken to fall down. You can see that the time increases as the learning progresses. Figure 15 shows the value of $p(t)$ in the 986th trial. It is clear that $p(t)$ changes periodically. These results indicate that a stamping motion was acquired, but the robot's idling leg does not rise perfectly when we look at the photos in Figure 13. We assume that the first reason for these results is that it is difficult to control the angle of ankle using inverse kinematics (since inverse kinematics cannot control $\theta_{AF}^{(R)}$ and $\theta_{AF}^{(L)}$ to be parallel to the ground). The second reason is that we only used y-coordinate values of the waist for learning, and the third is because we used equation (22) for rewards. To solve the second issue, we can use its z-coordinate value. Using equation (22), the third reason, a small periodic motion is obtained (Figure 16). To solve this problem, we should consider another reward function for this experiment. We will explore these areas in our future research.

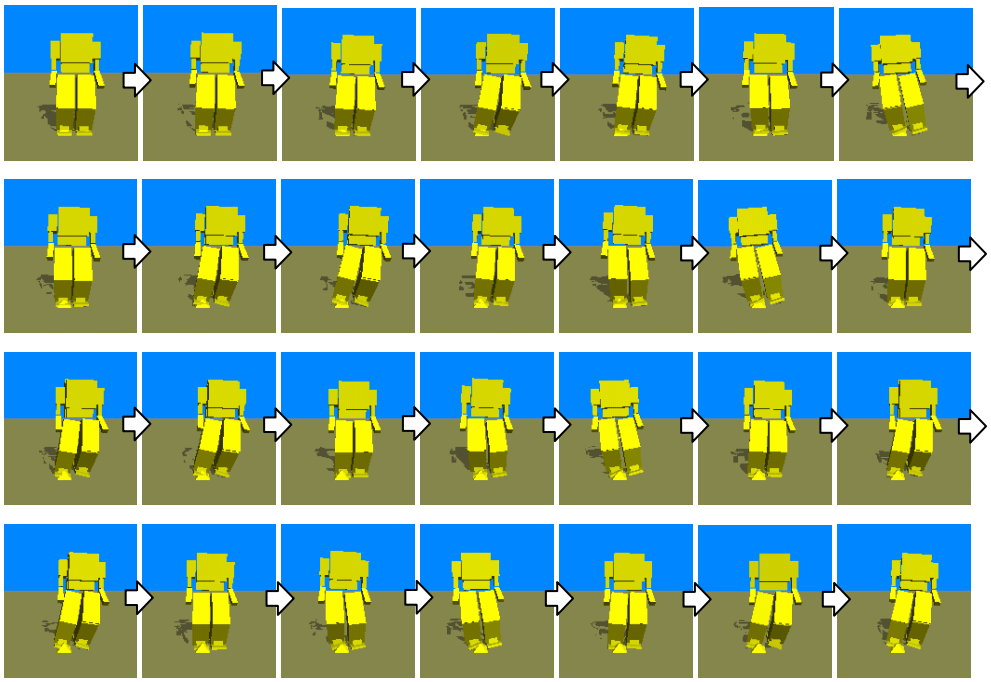


Fig. 13. Simulation result (986th trial).

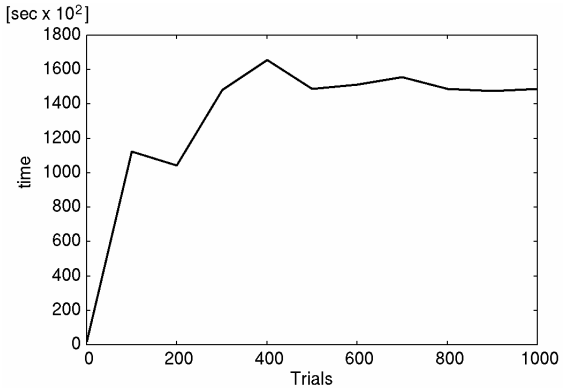


Fig. 14. Time during trial (averaged over 100 repetitions). If the value of a vertical axis is large, the stamping motion extends for a long time.

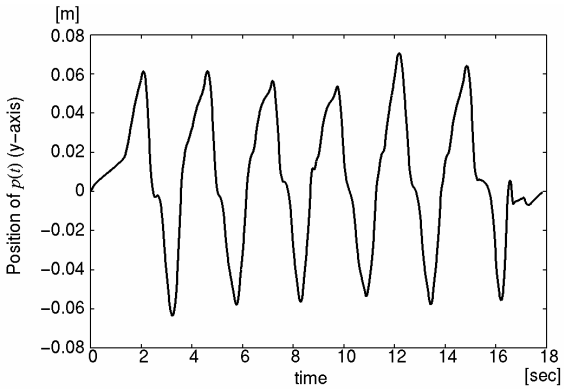


Fig. 15. Position of $p(t)$ in horizontal direction in 986th trial.

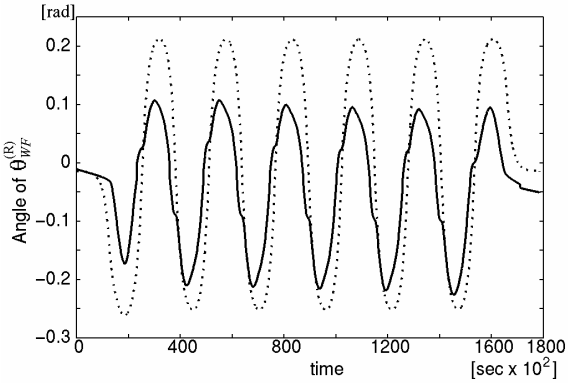


Fig. 16. Ideal angle and output angle of $\theta_{WF}^{(R)}$ with AE-GSBFN in 986th trial. The dotted line indicates an ideal motion and the solid line indicates the acquired motion with AE-GSBFN. It is clear that the acquired motion consists of small periodic motions compared with the deal motion.

5. Conclusion

In this chapter, we proposed a dynamic allocation method of basis functions, AE-GSBFN, in reinforcement learning. Through allocation and elimination processes, AE-GSBFN overcomes the curse of dimensionality and avoids a fall into local minima. To confirm the effectiveness of AE-GSBFN, we applied it to the motion control of a humanoid robot. We demonstrated that AE-GSBFN is capable of providing better performance than A-GSBFN, and we succeeded in enabling the learning of motion control of the robot.

The future objective of this study is to do some general comparisons of our method with other dynamic neural networks, for example, Fritzke's "Growing Neural Gas" (Fritzke, 1996) and Marsland's "Grow When Required Nets" (Marsland et al., 2002). An analysis of the necessity of hierarchical reinforcement learning methods proposed by Morimoto and Doya (Morimoto & Doya, 2000) in relation to the standing up simulation is also an important issue for the future study.

6. References

Albus, J. S. (1981). *Brains, Behavior, and Robotics*, Byte Books

Boyan, J. A. & Moore, A. W. (1995). Generalization in reinforcement learning: Safely approximating the value function, *Advances in Neural Information Processing Systems*, Vol. 7, 369–376

Fritzke, B. (1996). Growing self-organizing networks -- why?, *European Symposium on Artificial Neural Networks*, 61–72

Gullapalli, V. (1990). A stochastic reinforcement learning algorithm for learning real valued functions, *Neural Networks*, Vol. 3, 671–692

Marsland, S. ; Shapiro, J. & Nehmzow, U. (2002). A self-organizing network that grows when required, *Neural Networks*, Vol. 15, 1041–1058

Michie, D. & Chambers, R. A. (1968). BOXES: An Experiment in Adaptive Control, In: *Machine Intelligence 2*, E. Dale and D. Michie (Ed.), pp. 137-152, Edinburgh

Moore, A. W. & Atkeson, C. G. (1995). The parti-game algorithm for variable resolution reinforcement learning in multidimensional state space, *Machine Learning*, Vol. 21, 199–234

Mori, T.; Nakamura, Y., Sato, M., & Ishii, S. (2004). Reinforcement Learning for CPG-driven Biped Robot, *Nineteenth National Conference on Artificial Intelligence (AAAI2004)*, pp. 623-630

Morimoto, J. & Doya, K. (1998). Reinforcement learning of dynamic motor sequence: Learning to stand up, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1721–1726

Morimoto, J. & Doya, K. (1999). Learning dynamic motor sequence in high-dimensional state space by reinforcement learning -- learning to stand up -- , *IEICE Transactions on Information and Systems*, Vol. J82-D2, No. 11, 2118–2131

Morimoto, J. & Doya, K. (2000). Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning, *International Conference on Machine Learning*, pp. 623–630

Nakamura, K. & Hanafusa, H. (1984). Singularity Low-Sensitive Motion Resolution of Articulated Robot Arms, *Transactions of the Society of Instrument and Control Engineers*, Vol. 20, No. 5, pp. 453–459 (in Japanese)

Samejima, K. & Omori, T. (1998). Adaptive state space formation method for reinforcement learning, *International Conference on Neural Information Processing*, pp. 251–255

Schaal, S. & Atkeson, C. C. (1996). From isolation to cooperation: An alternative view of a system of experts, *Advances in Neural Information Processing System*, Vol. 8, 605–611

Smith, R. Open Dynamics Engine, http://opende.sourceforge.net/ode.html

Sutton, R. S. & Barto, A. G. (1998). *Reinforcement Learning: An Introduction*, MIT Press

Takahashi, Y. ; Asada, M. & Hosoda, K. (1996). Reasonable performance in less learning time by real robot based on incremental state space segmentation, *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1518–1524

Humanoid Robots: New Developments

Edited by Armando Carlos de Pina Filho

ISBN 978-3-902613-00-4

Hard cover, 582 pages

Publisher I-Tech Education and Publishing

Published online 01, June, 2007

Published in print edition June, 2007

For many years, the human being has been trying, in all ways, to recreate the complex mechanisms that form the human body. Such task is extremely complicated and the results are not totally satisfactory. However, with increasing technological advances based on theoretical and experimental researches, man gets, in a way, to copy or to imitate some systems of the human body. These researches not only intended to create humanoid robots, great part of them constituting autonomous systems, but also, in some way, to offer a higher knowledge of the systems that form the human body, objectifying possible applications in the technology of rehabilitation of human beings, gathering in a whole studies related not only to Robotics, but also to Biomechanics, Biomimmetics, Cybernetics, among other areas. This book presents a series of researches inspired by this ideal, carried through by various researchers worldwide, looking for to analyze and to discuss diverse subjects related to humanoid robots. The presented contributions explore aspects about robotic hands, learning, language, vision and locomotion.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Masayoshi Kanoh and Hidenori Itoh (2007). Obtaining Humanoid Robot Controller Using Reinforcement Learning, Humanoid Robots: New Developments, Armando Carlos de Pina Filho (Ed.), ISBN: 978-3-902613-00-4, InTech, Available from: http://www.intechopen.com/books/humanoid_robots_new_developments/obtaining_humanoid_robot_controller_using_reinforcement_learning

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821