Evolutionary Creation of an Adaptive Controller for a Legged-Robot: A Dynamically-Rearranging Neural Network Approach

Akinobu Fujii[1] Akio Ishiguro[1] Kei Otsu[1] Yoshiki Uchikawa[1]
Takesi Aoki[2] Peter Eggenberger[3]

[1]Dept. of Computational Science and Engineering, Nagoya University
Nagoya 464-8603, Japan, {akinobu, ishiguro, kei, uchikawa}@cmplx.cse.nagoya-u.ac.jp

[2]Nagoya Municipal Industrial Research Institute, Nagoya 456-0058, Japan, aoki@nmiri.city.nagoya.jp

[3]ATR Human Information Processing Research Laboratories, Kyoto 619-0288, Japan, eggen@hip.atr.co.jp

Abstract

As there exists highly complicated interaction dynamics, it is in general extremely difficult to design controllers for legged robots. Therefore, the Evolutionary Robotics is one of the most promising approaches since it can automatically construct controllers by taking embodiment and the interaction dynamics with the environment into account. Although this approach has such advantages, there still exists several problems that have to be solved. One of the critical problems is known as the gap problem; the controller evolved in the simulator show not the same fitness as those in the real world due to unforeseen perturbation. Therefore, it is highly necessary to establish a method enables to efficiently construct adaptive controllers that can cope with different situation. For this purpose we introduce the concept of neuromodulators, allowing to evolve neural networks which can adjust not only the synaptic weights, but also the structure of the neural network by blocking and/or activating synapses or neurons. We apply this concept to create an adaptive legged–robot controller which realizes not only follow the desired walking velocity but also regulate the amount of the torque output applied to each joint for energy efficiency according to the current situation.

1. Introduction

Legged robots show siginificant advantages over wheeled robots since they can traverse in uneven and unstructured environments. This high mobility stems from the fact that in contrast to wheeled or tracked robots legged robots *discretely* contact with their environments via their legs. This, however, inevitably causes highly complicated dynamics between the robots and their environments. Thus, it is in general extremely difficult to design controllers for legged robots.

So far various methods have been proposed to construct legged–robot controllers, whilst very few studies have investigated the design of controllers considering the interaction dynamics with the environments [1, 4, 17, 18]. However, since the above approaches are based on a hand–crafted manner, it is questionable whether or not these approaches will be still feasible (i.e. easily implemented) as the complexity of the desired task and the interaction dynamics increases.

On the other hand, recently the *Evolutionary Robotics* (ER) approach has been attracting a lot of concern in the field of robotics and artificial life [6, 16]. In contrast to the conventional approaches where designers have to construct controllers in a top–down manner, the methods in the ER approach have significant advantages since they can autonomously and efficiently construct controllers by taking *embodiment* (e.g. physical size and shape of robots, sensor/motor properties and disposition etc.) and the *interaction dynamics* between the robot and its environment into account.

In this paper, an evolutionary creation of an *adaptive* neuro–contorlller for a legged–robot is investigated. In order to construct adaptive controllers, we focus on creation of feedback loops between the robot and its environment based on its embodiment, and their regulation mechanisms as the targets to be evolved instead of evolving synaptic weights as in the conventional ER apprroaches. To this end, we introduce the concept of *neuromodulators*, allowing to evolve neural networks which can adjust not only the synaptic weights but also the structure of the neural network by blocking and/or activating synapses or neurons according to the current situation.

Here as the initial step of the investigation, we attempt to create a single–leg controller which realizes not only follow the desired walking velocity but also regulate the amount of the torque output applied to each joint for energy efficiency under various body weights. As there exists no theory about how such dynamic neural network can be constructed, the evolutionary approach is the method of choice to explore the interactions between the neuromodulators, receptors, synapses and neurons. Simulations are carried out to verify the feasability of the proposed method.

2. Issues in the Evolutionary Robotics Approach

In the ER approaches, artificial neural networks are widely used to construct controllers for autonomous mobile agents, because they can generalize, are non–linear and noise–tolerant. Another advantage of neural network–driven robots is that a neural network is *low level description* of a controller. More precisely, it directly maps sensor readings onto motor outputs. Due to this rich emergent properties can be expected.

Although the ER approach has the above advantages, the following drawbacks still exist:

First, as the complexity of the desired task increases, it becomes significantly difficult to evolve the whole controller in one go. This problem is sometimes reffered as the *bootstrap problem* [16]. Thus it is demanded to develop a methodology which can automatically synthesize more complex behavior than those designed by hand. In order to alleviate this problem, several authors introduced the concept of *shaping* [3], *canalization* [16], *incremental evolution* [2] and so forth.

Second, as the evolution in the real world is time–consuming, simulations are used instead to evolve the controller in simulated environments and the best individuals are tested in the real world. The flaw of this combined approach is that evolved agents in simulated environments often show a siginificantly different behavior in the real world due to unforeseen perturbations, since they tend to overadapt to the given environments through the evolutionary process. In other words, a *gap* between the simulated and real environments exists. Therefore, it is indispensable to establish a method which enables the evolved controllers adapt not only to sepecific environments, but also to environmetnal perturbations.

In the following, we particularly deal with the second problem. Now, the the following question arises. How can robots recognize their current situation and regulate their behavior appropriately?

A part of the answer may be that in the studies so fare made in ER no attempt was made to select directly for adaptation by changing the settings of the experiments. As even a simple thermostat needs sensory feedback to be able to control the temperature, an essential ingredient to any adaptive controller are controlling sensors to give the neural controller data how to change its current state towards the "good" one.

To construct robust controllers against environmental changes, in this study we focus on creation of feedback loops and their regulation mechanisms as the target to be evolved instead of evolving the synaptic weights(see Figure 1). If we can successfully evolve the appropriate regulation mechanism, we can expect high adaptability against environmental perturbations.

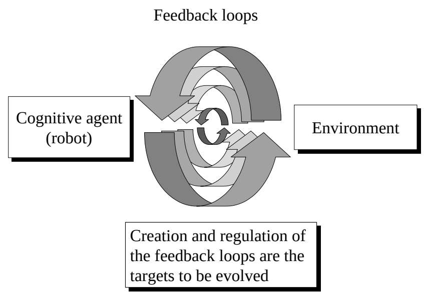

Figure 1: Feedback loops between the robot and its environment.

In principle the information carried by the feedback loops can have the following two effects: Either one changes the weights of the synapses and the neurons' thresholds or one alters dynamically the structure of the neural network itself. The question is how can this be done and can such methods be used to solve the above problem.

Interestingly, neuroscientific results suggest that biological networks not only adjust the synaptic weights, but also the neural structure by blocking or activating synapses or neurons by the use of signaling molecules, so called *neuromodulators* [13]. These findings stem from investigations made with the lobster's stomatogastric nervous system in which certain active neurons diffuse neuromodulators which then rearrange the networks. Note that the effect of a neuromodulator depends not only on theses substances, but also on the specific receptors, which are differently expressed in different cells.

The release of the neuromodulators depends on the activity of the neurons and therefore different sensor inputs may cause different patterns of released neuromodulators. As such dynamic mechanisms yield remarkable adaptation in living organisms, the proposed approach not only carries promise for a better understanding of adaptive networks, but they can be also applied to real–world problems as we already showed in the previous work [5, 11].

3. Lessons from the Biological Findings

3.1. Dynamic rearrangement in the biological nervous system

Investigations carried out on the lobsters' stomatogastric nervous system suggest that biological nervous systems are able to dynamically change their structure as well as their synaptic weights [13].

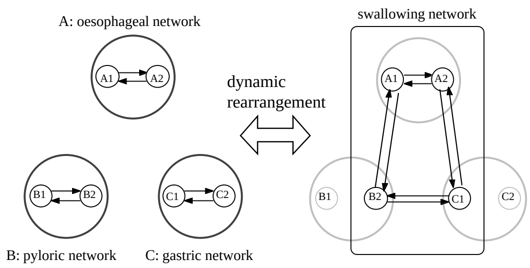

Figure 2: Dynamically–rearrangement of a lobster's stomatogastric nervous system.

This stomatogastric nervous system mainly consists of an *oesophageal*, a *pyloric*, and a *gastric* network. Normally, these three individual networks show their own independent oscillatory behaviors, but in the moment a lobster is eating the networks are integrated and reconstructed to a new one, the swallowing network, in which certain neurons and connections are excluded and formerly inactive connections are activated (see Figure 2).

Recent studies in neurophysiology showed neuromodulators (hereafter: NMs) play a crucial role to regulate this remarkable phenomenon (e.g. changing properties of synapses as well as neurons).

3.2. Neuromodulators

NMs are substances that can dynamically influence several properties of synapses as well as neurons and therefore the function of a neural network. In contrast to *neurotransmitters* (NTs) the effect of NMs spreads slower and lasts longer. NMs change the processing characteristics of neural networks by affecting the membrane potential, the rate of changing the synapses (i.e. influence on learning mechanisms) and other parameters. Typical NMs are *acetylcholine*, *norepinephrine*, *serotonin*, *dopamin* (all are also used as NTs), *somatostatine* and *cholecystokinine* (both also used as hormones in the human body) and many small proteins. Although these substances are released in a less local manner than NTs, the effects can be quite specific. This specificity comes from specific receptors on the neurons and their synapses.

These NMs stem either locally from the neural network itself or from specific sub–cortical nuclei. The local release of NMs depends on the activity of the local neural network itself. On the other hand, sub–cortical nuclei as the locus coeruleus (*noradrenergic innervation*), the ventral tegmental area (*dopaminergic innervation*) or the basal fore–brain nuclei (*cholinergic innervation*) send neuromodulatory axons to cortical structures to release NMs from axonal varicosities which is called volume transmission. Many publications in neuroscience show the importance of NMs for dynamic rearrangement of neuronal modules [13, 9] or for learning and memory (switching between learning and recall mode) [8].

In this study we implemented the following properties:

- dynamic change of a neuron's threshold
- dynamic blocking of synapses (possibly neurons)
- dynamic change of the inhibitory or excitatory properties of a synapse
- dynamic modulation of synaptic weights (i.e. learning).

4. Proposed Method

4.1. Basic concept

The basic concept of our proposed dynamically–rearranging neural networks (hereafter: DRNN) is schematically depicted in Figure 3. As in the figure, unlike the conventional neural networks, we assume that each neuron can potentially diffuse its specific (i.e. genetically–determined) NMs according to its activity, and each synapse has receptors for the diffused NMs. We also assume that each synapse independently interprets the received NMs, and changes its properties (e.g. synaptic weight). The way of these changes exerted on the synapses is also genetically–determined.

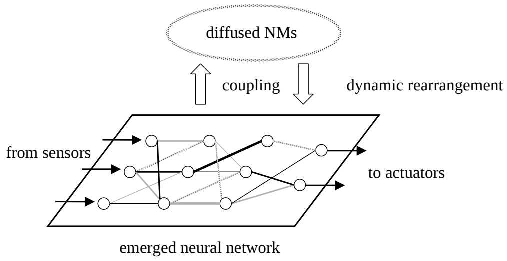

Figure 3: Basic concept of the DRNN.

By selecting for regulatory feedback loops (cyclical interaction between the diffusion and reaction of NMs), we expect to be able to evolve adaptive neural networks, which show not only a seamless transfer from simulations to the real world but also robustness against environmental perturbations (in the figure, the thick and thin lines denote the connections being strengthened and weakened by NMs, respectively).

In summary, in contrast to the conventional ER approach that evolves synaptic weights and neuron's bias of neuro–controllers, in this approach we evolve the following mechanisms:

- Diffusion of NMs (when, which type of NMs are diffused from each neuron?)
- Reaction to NMs (how do the receptors on each synapse interpret the received NMs, and modify the synaptic property?)
- Network architecture (the number of interneurons, and how to connect among the sensory, inter– and motor neurons)

To determine the above parameters, we use a Genetic Algorithm (GA). Detailed explanation on how a GA is implemented is given later.

4.2. Application problem

4.2.1. *Task*

Our aim is to create an adaptive controller for a multi-legged robot that can appropriately cope with different situation. However, in general it is extremely difficult to evolve the whole controller in one go.

Thus, in order to investigate the feasibility of the DRNN approach, in this study we attempt to construct an adaptive controller for a single–legged robot as the initial step of the investigation. Here the task of the robot is to not only follow the desired walking velocity but also regulate the amount of the torque output applied to each joint for energy efficiency under various body weights.

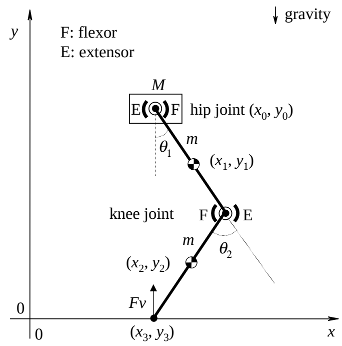

Figure 4: Model of the single-legged robot.

4.2.2. *Single–legged robot model*

The model of the single-legged robot is schematically illustrated in Figure 4. The robot consists of a body and two physical links (i.e. thigh and shank) with two joints (i.e. hip joint and knee joint). These joints are all independently driven by pairs of antagonistic actuators (i.e. *flexor* and *extensor*) in order to take not only static torque but also the stiffness of the joints (for energy efficiency) into account.

The hip angle (θ_1) is measured according to the deviation from the vertical line. We assume that the hip joint can rotate between an angle of $-60°$ (full extension) and $60°$ (full flexion). On the other hand, the knee angle (θ_2) is measured with reference to the thigh position, where the full knee extension is an angle of $0°$ and full knee flexion $-120°$.

We assume that each joint is equipped with a pair of AEP (anterior extreme position) and PEP (posterior extreme position) sensors which inform the current hip and knee angles, and also equipped with a torque sensor to measure the applied static torque. In addition, there exist a load sensor at the tip of the leg to detect the amount of the vertical force from the ground (F_v in the figure).

4.2.3. DRNN controller

Figure 5 schematically represents the structure of the DRNN controller for the single–legged robot. In the figure, the neurons with S and M denote the sensory and motor neurons, respectively. The neuron S_8 is a neuron which is always activated (i.e. output value of 1.0) to encourage any motor activity even under no sensory inputs. The rest of the neurons are interneurons. Detailed explanation on how to create the controller will be given in the next section.

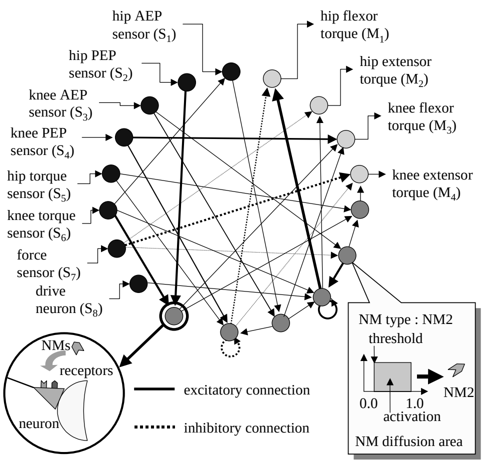

Figure 5: Controller for the single-legged robot.

As for the neural dynamics, we use a *leaky integrator* model which is expressed as:

$$\tau_i \frac{du_i}{dt} = -u_i + \sum_j w_{ij} \cdot a_j - \theta_i \qquad (1)$$

$$a_i = \frac{1}{1 + \exp(-0.5u_i)} \qquad (2)$$

where a_i is the activity of neuron i, and w_{ij} represents the synaptic weight of a connection from neuron j to neuron i. u_i is the membrane potential of neuron i, and τ_i denotes the time constant of the membrane potential. θ_i is the threshold of the neuron's activity. We use a standard sigmoidal function to limit the neuron's activity.

4.3. Encoding scheme for the DRNN

Both the network architecture and the way of diffusion and reaction to NMs in the DRNN are closely related to not only the given task but also the embodiment and the interaction dynamics with the environment. Thus it is preferable to automatically determine these parameters through the evolutionary process. In order to exclude possible presupposition on these parameters we introduce the concept of the *developmental process*.

4.3.1. Structure of the Genotype

The genotype for the DRNN (hereafter: network genotype) is expressed as a binary bit string, and is composed of a set of blocks, each block corresponding to the genotype for a single neuron (neuron genotype). For the ease of understanding, we first explain the structure of the neuron genotype in detail.

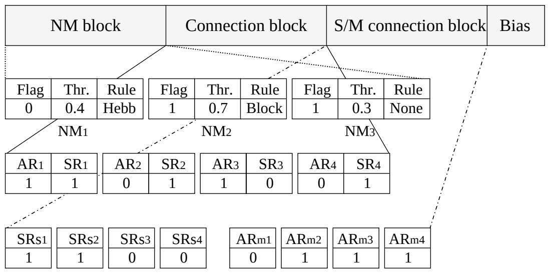

Figure 6: Structure of the neuron genotype.

Figure 6 shows an example of the neuron genotype. As in the figure, each neuron genotype consists of the bias gene and three blocks: *NM block*, *connection block*, and *S/M connection block*. The bias gene represents the threshold of the neuron. NM, connection and S/M connection blocks contain genetic information which specify the following properties of the corresponding neurons, respectively:

(a) *NM block*

The genes contained in this block determine the way of diffusion and reaction to the NMs. As in the figure, this block is composed of a series of the parameter set (*flag*, *threshold*, *rule*), and each set is responsible for one specific type of the NMs. Thus in this example at most three types of NMs can be diffused from the neuron concerned.

The parameters *flag* and *threshold* determine under which condition the corresponding type of NM is diffused from the neuron concerned. If the neuron's activity exceeds the genetically–determined threshold value and also the *flag* is active, the corresponding NM will be immediately diffused. Here, we assume that the concentration of the diffused NM (denoted as $c(NM_k)$) is proportional to the avtivation value of the neuron concerned (a_i) within the diffusible area.

On the other hand, the parameter *rule* determines how the receptor on the synapses which outgrow from the neuron concerned interprets the corresponding type of NM and modifies the synaptic properties. In order to reduce the genetic information here we assume that all the synapses outgrowing from a given neuron have the same set of the receptors.

Each parameter *rule* can take one of the following four types of modulation: *Hebbian learning*, *anti–Hebbian learning*, *non–learning*, and *blocking* (i.e. excluding the synapse), respectively.

Suppose that there exists k types of receptors on a given synapse, the following equation is used for the dynamic modulation of the synaptic weights as:

$$c_{total}(NM_k) = \sum_N c(NM_k) \quad (3)$$

$$s = \sum_k R_{ij}(NM_k) \cdot c_{total}(NM_k) \quad (4)$$

$$w_{ij}^{t+1} = \begin{cases} w_{ij}^t + \eta|s|(-1 - w_{ij}^t)a_i a_j & \text{for } s < 0 \\ w_{ij}^t & \text{for } s = 0 \\ w_{ij}^t + \eta|s|(1 - w_{ij}^t)a_i a_j & \text{for } s > 0 \end{cases} \quad (5)$$

where, $c_{total}(NM_k)$ represents the total concentration of the diffused NM of type k in the network at a given time, and N is the number of the neurons. η is the leaning rate, and $R_{ij}(NM_k)$ denotes the parameter which determines how the synapse concerned modifies its property when the NM of type k is combined with the receptor on it. For this, we use +1, −1, and 0 to express *Hebbian learning*, *anti–Hebbian learning*, and *non–learning*, respectively.

We assume that the *blocking* modulation has the highest priority among the four types of modulation. Thus, if a receptor which expresses blocking is activated, the corresponding synaptic weights will be forcibly set to zero notwithstanding other receptors' states.

(b) *Connection block*

This block is responsible for the connection establishment among the neurons. As in fig.6, this block possesses a series of the parameter set (AR_i, SR_i). Here AR_i and SR_i stand for an axonal and synaptic receptors, respectively. If the parameter AR_i is activated, the neuron concerned outgrows axons. Each AR (SR) has its own ID–number, and the AR can make connections only with the SR with the same ID–number.

(c) *S/M connection block*

This block determines the connection establishment between the corresponding neuron and the sensor and/or motors, equipped with the robot. We assume that each sensor (motor) has its specially dedicated axonal (synaptic) receptor ARs_i (SRm_i), and this receptor is always active. If the neuron concerned activates the gene SRs_i (ARm_i), then this neuron can connect with the sensor with ARs_i (the motor with SRm_i).

4.3.2. *Genetic operators*

Each gene in the neuron genotype is randomly changed with the prespecified probalility. In addition to this, as for the NM block and connection block in each neuron genotype we use specially dedicated mutation operators; random *insertion* and/or *deletion* of the parameter set (i.e. (*flag*, *threshold*, *rule*) for the NM block, (AR_i, SR_i) for the connection block) is applied in order to explore the appropriate number of NM types and connections with other neurons.

Due to the above insertion and deletion operators, each neuron genotype can have different length. Consequently, the length of a network genotype can differ from one to another. Thus, in this study we use a *mixing pot method* [15] as a crossover operator. Despite of its simplicity, this operator can efficiently deal with the number of neurons as the target to be evolved (detailed description of this method see [15]).

5. Results

As mentioned before, it becomes significantly difficult to evolve the whole controller in one go as the complexity of the desired task increases. To alleviate this, we introduce the concept of an incremental evolution. In this study, we first evolve the controller under the following evaluation criterion:

$$fitness_1 = \frac{1.0}{D^* - D} \times \sum_{i=1}^{2} \int_0^{T_{max}} |\dot{\theta}_i| dt \quad (6)$$

where D^* and D denote the desired and resultant walking distance, respectively. $\dot{\theta}_i$ is the angular velocity of joint i, and T_{max} represents the duration of the evaluation process. The second term of the right hand side of the equation is for encouraging any oscillatory behavior.

After evolving during 200 generations under this fittness function, the population obtained in the last generation is used for the second stage of the evolution as the initial population. The fitness function used in the second stage is expressed as:

$$fitness_2 = \frac{fitness_1}{E_{con}} \tag{7}$$

$$E_{con} = \sum_{i=1}^{2} \int_{0}^{T_{max}} \{\delta(T_i\dot{\theta}_i) + {T_{fi}}^2 + {T_{ei}}^2\}dt \tag{8}$$

$$T_i = T_{fi} - T_{ei} \tag{9}$$

$$\delta(x) = \begin{cases} x & \text{for } x \geq 0 \\ 0 & \text{for otherwise} \end{cases} \tag{10}$$

where E_{con} is the amount of the energy consumed during the evolutionary process. T_{fi} and T_{ei} denote the flexor and extensor torque applied to joint i, respectively. The body parameters used in the following simulations are listed on Table 1[1].

To verify the adaptability of the DRNN, in the simulations each individual is tested under various body mass (M=1.8kg, 2.8kg) and the averaged fitness value is used for the evaluation.

Table 1: Body parameters of the single–legged robot.

part	length	mass
body	-	1.80/2.80[kg]
thigh	0.1[m]	0.15[kg]
shank	0.1[m]	0.15[kg]

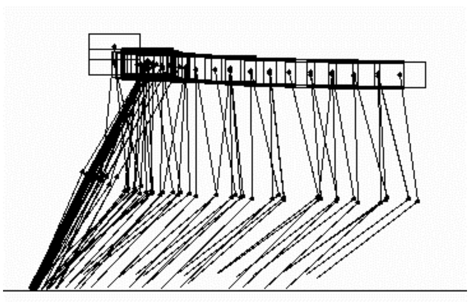

Figure 7: Resultant trajectory of the best evolved agent in the case of M=1.8kg.

Figure 7 and 8 represent the resultant trajectory of the best evolved agent under the bady mass of M=1.8kg and 2.8kg, respectively. Figure 9 and 10 are the transition of the torque outputs at the waist joint under the same condition. From the

[1]For simplicity, the body is represented as a material particle (i.e. no physical entity) with mass.

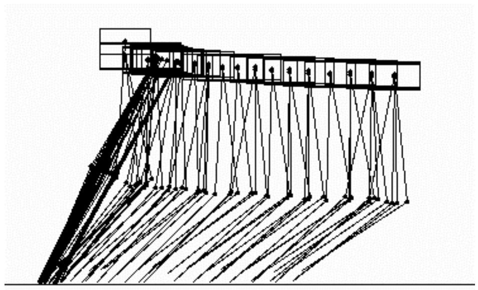

Figure 8: Resultant trajectory of the best evolved agent in the case of M=2.8kg.

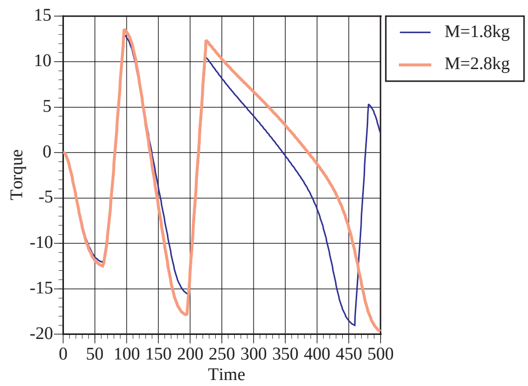

Figure 9: Transition of the torque output at the beginning of walking.

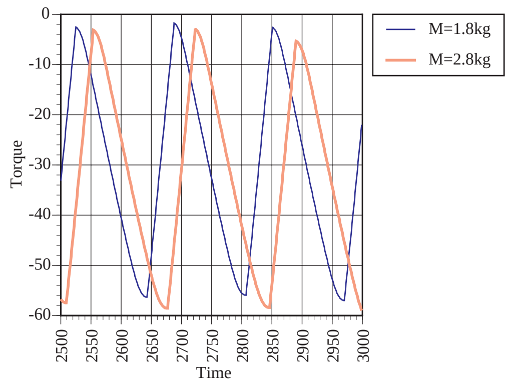

Figure 10: Transition of the torque output under the steady–state walking.

figures, it is understood that irrespective of the different body mass, the robot can successfully cover almost the same distance by adjusting the amount of the torque output at the joints. We observed that different types of the NMs were diffused according to the sensory inputs in order to regulate the torque outputs applied to each joint.

6. Conclusions

In this study, evolutionary creation of an adaptive controller for a single–legged robot was investigated. To this end we introduced the concept of the dynamically–rearranging function in the biological nervous systems. The preliminary simulation results were encouraging. We expect this concept will provide a methodology for not only seamless transfer from the simulated to the real environments but also evolutionary creation of controllers with high adaptability. Detailed analysis of the evolved DRNN is currently under investigation. We will also quantitatively investigate the adaptability of the evolved controllers compared with the ones of the conventional approach where the syanaptic weights are the targets to be evolved.

In the future, we will apply this concept to biped and quadruped robots. For this purpose we are currently developing a 3D simulator with the use of a general–purpose fully dynamics simulator DADS (Dynamic Analysis and Design System).

References

[1] R. Beer, J. Chiel, and L. Sterling. Evolution of plastic neurocontrollers for situated agents. *An artificial insect, American Scientist,* **79***, 444–452*, 1989.

[2] D. Cliff, I. Harvey, and P. Husbands. Incremental evolution of neural network architectures for adaptive behaviour. *Proc. of the First European Symposium on Artificial Neural Networks (ESANN93), 39–44*, 1993.

[3] M. Colombetti and M. Dorigo. Training agents to perform sequential behavior. *Adaptive Behavior,* **2(3)***, 247–276*, 1994.

[4] H. Cruse, C. Bartling, J. Dean, M. Dreifert, T. Kindermann, and J. Schmitz. Walking: a complex behavior controlled by simple networks. *Adaptive Behavior,* **3***, 385–418*, 1995.

[5] P. Eggenberger, A. Ishiguro, S. Tokura, T. Kondo, T. Kawashima, and T. Aoki. Toward seamless transfer from simulated to real worlds: A dynamically–rearranging neural network approach. *Proc. of the Eighth European Workshop on Learning Robot (EWLR-8), 4–13*, 1999.

[6] D. Floreano and F. Mondada. Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. *Proc. of the 3rd International Conference on Simulation of Adaptive Behavior, MIT Press, 421–430*, 1994.

[7] Floreano, D. and Mondada, F. (1996). Evolution of plastic neurocontrollers for situated agents. *From animals to animats 4: Proc. of the 4rd International Conference on Simulation of Adaptive Behavior, MIT Press, 401–411.*

[8] M. Hasselmo. Neuromodulation and cortical function: modeling the physiological basis of behavior. *Behavioral Brain Research,* **67***, Elsevier Science B.V., 1–27*, 1995.

[9] S.L. Hooper and M. Moulins. Switching of a neuron from one network to another by sensory–induced changes in membrane properties. *SCIENCE,* **244***, 1587–1589*, 1989.

[10] Husbands, P., Smith, T., O'Shea, M., Jakobi, N., Anderson, J., and Philippides, A. (1998). Brains, gases and robots. *In Proc. ICANN98, Springer–Verlag, 51–63.*

[11] A. Ishiguro, T. Kondo, Y. Uchikawa, and P. Eggenberger. Autonomous robot control by a neural network with dynamically–rearranging function. *Proc. of the 11th SICE Symposium on Decentralized Autonomous Systems, 213–218 (in Japanese)*, 1999.

[12] Jacobi, N., Husbands, P., and Hervey, I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. *Third European Conf. on Artificial Life (ECAL95), Advances in Artificial Life, Springer, 704–720.*

[13] P. Meyrand, J. Simmers, and M. Moulins. Construction of a pattern–generating circuit with neurons of different networks. *NATURE,* **351***-2MAY, 60–63*, 1991.

[14] Miglino, O., Lund, H., and Nolfi, S. (1995). Evolving mobile robots in simulated and real environments. *Artificial Life 2, 417–434.*

[15] K. Nakano, H. Hiraki, and S. Ikeda. A learning machine that evolves. *Proc. of International Conference on Evolutionary Computation (ICEC95), 808–813*, 1995.

[16] S. Nolfi. Evolving non–trivial behaviors on real robots: A garbage collecting robot. *Robotics and Autonomous System,* **22***, 187–198*, 1997.

[17] G. Taga, Y. Yamaguchi, and H. Shimizu. Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment. *Biological Cybernetics,* **65***, 147–159*, 1991.

[18] T. Wadden and Ö. Ekeberg. A neuro-mechanical model of legged locomotion: single leg control. *Biological Cybernetics,* **79***, 161–173*, 1998.