Neural Networks 29–30 (2012) 8–19

Contents lists available at SciVerse ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

The eMOSAIC model for humanoid robot control

Norikazu Sugimoto [a,b,*], Jun Morimoto [b], Sang-Ho Hyon [b,c], Mitsuo Kawato [b]

[a] National Institute of Communication Telecommunication, 2-2-2 Hikaridai Seika-cho, Soraku-gun, Kyoto 619-0288, Japan

[b] ATR Computational Neuroscience Laboratories, 2-2-2 Hikaridai Seika-cho, Soraku-gun, Kyoto 619-0288, Japan

[c] Department of Robotics, Ritsumeikan University, 1-1-1 Noji Higashi, Kusatsu, Shiga 525-8577, Japan

ARTICLE INFO

Article history:
Received 16 November 2010
Received in revised form 20 September 2011
Accepted 13 January 2012

Keywords:
Modular architecture
Nonlinear and non-stationary control problem
Humanoid robot
Computational neuroscience

ABSTRACT

In this study, we propose an extension of the MOSAIC architecture to control real humanoid robots. MOSAIC was originally proposed by neuroscientists to understand the human ability of adaptive control. The modular architecture of the MOSAIC model can be useful for solving nonlinear and non-stationary control problems. Both humans and humanoid robots have nonlinear body dynamics and many degrees of freedom. Since they can interact with environments (e.g., carrying objects), control strategies need to deal with non-stationary dynamics. Therefore, MOSAIC has strong potential as a human motor-control model and a control framework for humanoid robots. Yet application of the MOSAIC model has been limited to simple simulated dynamics since it is susceptive to observation noise and also cannot be applied to partially observable systems. Our approach introduces state estimators into MOSAIC architecture to cope with real environments. By using an extended MOSAIC model, we are able to successfully generate squatting and object-carrying behaviors on a real humanoid robot.

1. Introduction

Previous studies have suggested that the human central nervous system acquires and switches internal models of outside environments to adaptively perform motor control of the body in various environments (Imamizu, Kuroda, Yoshioka, & Kawato, 2004; Imamizu et al., 2007). Modular selection and identification for control (MOSAIC) architecture composed of multiple linear state predictors and controllers was originally proposed in order to explain the motor-control strategy of the human brain (Haruno, Wolpert, & Kawato, 2001; Wolpert & Kawato, 1998). Humans and humanoid robots both have nonlinear body dynamics and many degrees of freedom (DOF). Moreover, they both interact with objects in real environments, which naturally requires nonlinear and non-stationary control strategies. MOSAIC architecture has strong potential as a human motor-control model and a control framework for humanoid robots, and its flexible structure enables the robot to control the nonlinear and non-stationary environment. But application of the MOSAIC model has been limited to simple simulated dynamics, which can be partially attributed to two problems: (1) the MOSAIC model does not explicitly consider the existence of noise input to sensory systems and (2) it assumes full observation and thus cannot deal with partially observable systems. In this study, we propose an extension of the MOSAIC architecture to cope with observation noise and partially observable systems. Each module of MOSAIC has a forward model, and we can adopt these to construct a state estimator. Moreover, using state estimators can provide a reasonable model of the sensorimotor function of the central nervous system, as previously suggested in Wolpert, Ghahramani, and Jordan (1995). A state estimation strategy using switching linear models is also considered to be a useful approach for estimating hidden variables of complicated nonlinear dynamics (Ghahramani & Hinton, 2000). Our new method, called extended MOSAIC with state estimators (eMOSAIC), can deal with large observation noise and partially observable systems. Based on these advantages the proposed control framework can be applied to real systems such as humanoid robots.

A popular approach to controlling nonlinear dynamics is using the inverse dynamics model and canceling the nonlinear terms. Then, linear error dynamics are introduced to reduce the tracking error. This approach is called feedback linearization (FL; Slotine & Li, 1991). FL performs well if the given inverse dynamics model is accurate. However, it is well known that estimating the inverse model of the real system is difficult due to the existence of friction. The controller of the proposed method can be derived as a linear optimal controller, for which it is well known that the sensitivity for the modeling error is low. Thus, the proposed method is more robust against the modeling error than FL. Another

* Corresponding author at: National Institute of Communication Telecommunication, 2-2-2 Hikaridai Seiko-cho, Soraku-gun, 619-0288 Kyoto, Japan. Tel.: +81 0774 95 1064; fax: +81 0774 95 1259.

E-mail addresses: xsugi@nict.go.jp (N. Sugimoto), xmorimo@atr.jp (J. Morimoto), gen@fc.ritsumei.ac.jp (S. Hyon), kawato@atr.jp (M. Kawato).

0893-6080/$ – see front matter © 2012 Elsevier Ltd. All rights reserved.
doi:10.1016/j.neunet.2012.01.002

popular nonlinear control method is gain scheduling (GS) control (Rugh, 1991; Rugh & Shamma, 2000); The GS considers nonlinear dynamics as a linear parameter varying (LPV) system (Shamma & Athans, 1992) defined around a pre-determined state. Although an analysis of robustness is well studied (Chesi, 2010; Lim & How, 1998), a design of the scheduling parameter requires a knowledge of structure of task and plant, thus it is difficult to apply the GS to a complex environment such as a humanoid robot. On the other hand, in our proposed method appropriate models can be selected according to prediction accuracy of each linear state predictor.

Adaptive control (Astrom & Wittenmark, 1989) is a standard method for controlling a non-stationary system. In this method, controller parameters are adaptively modified according to the tracking error. Since this method assumes that the environment change is gradual, it cannot cope with a sudden change of dynamics; e.g., carrying or grabbing extra weight. To control such an environment, the switching control method seems to be suitable (Morse, 1996). In the proposed method, again, the module can be switched according to prediction accuracy of each linear state predictor.

In Section 2, we introduce eMOSAIC. In this study, we adopt an optimal control approach in the MOSAIC model (as proposed in Doya, Samejima, Katagiri, and Kawato (2002)).

In Sections 3 and 4, we evaluate the control performance of the eMOSAIC model in environments with large observation noise, partially observable setup, and modeling errors. We investigate (1) nonlinear control performance through a squatting task (Grimes, Chalodhorn, & Rao, 2006; Nakaoka, Nakazawa, & Ikeuchi, 2004) using either a two-link robot model or the humanoid robot, and (2) non-stationary control performance through a object-carrying task using either a single-pendulum robot model or the humanoid robot with a weight. We also show that eMOSAIC significantly outperforms the original MOSAIC in these tasks. In Section 3, we consider the squatting task by using the two-link robot model and the lifting task by using the single-pendulum robot model in a simulated environment. In Section 4, we apply eMOSAIC to control our humanoid robot CB-i (Fig. 1) to demonstrate that the proposed model can be used in a real environment. The results show that a humanoid robot can maintain its balance while performing the squatting and object-carrying tasks using the eMOSAIC model.

2. The eMOSAIC model

In this section, we introduce the eMOSAIC model to control a highly nonlinear system, such as our CB-i humanoid robot (see Fig. 1), in a non-stationary environment.

2.1. Optimal control problem of nonlinear and non-stationary dynamics

We consider an optimal control problem of nonlinear and non-stationary dynamics, where the dynamics are represented as:

$$\mathbf{x}(t+1) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) + \mathbf{n}(t), \tag{1}$$

$$\mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t), t) + \mathbf{v}(t), \tag{2}$$

where $\mathbf{x} \in \Re^N$, $\mathbf{u} \in \Re^D$, and $\mathbf{y} \in \Re^L$ are state, action and observation vectors, respectively, and $\mathbf{n}(t) \sim \mathcal{N}(0, \Sigma_{\mathbf{x}})$ and $\mathbf{v}(t) \sim \mathcal{N}(0, \Sigma_{\mathbf{y}})$ are system and observation noises. $\mathcal{N}(0, \Sigma)$ denotes a Gaussian distribution with zero mean and covariance Σ.

In an optimal control framework, the learning system tries to find the optimal controller to minimize the objective function:

$$J = E\left[\sum_{s=0}^{\infty} r(\mathbf{x}(s), \mathbf{u}(s))\right], \tag{3}$$

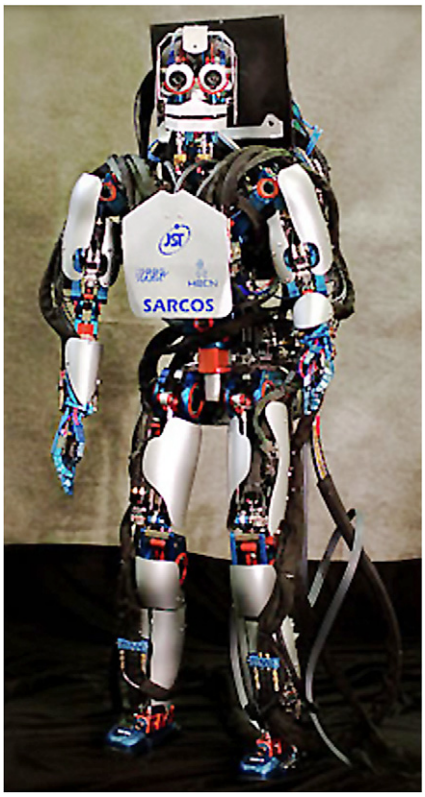

Fig. 1. Humanoid robot CB-i. Height: 1.6 m. Weight: 90 kg. Robot has 51 degrees of freedom.

where $r(\mathbf{x}, \mathbf{u})$ is the cost function. To find the optimal controller for minimizing the objective function, we estimate the value function:

$$V(\mathbf{x}(t)) = E\left[\sum_{s=t}^{\infty} r(\mathbf{x}(s), \mathbf{u}(s))\right]. \tag{4}$$

2.2. The eMOSAIC model

The eMOSAIC model has a modular architecture. Each module is composed of a state estimator, responsibility predictor, value function estimator, and controller. We approximate nonlinear and non-stationary dynamics by switching linear models and a nonlinear cost function by switching quadratic models:

$$\mathbf{x}(t+1) = A_i\mathbf{x}(t) + B_i\mathbf{u}(t) + c_i + \mathbf{n}(t), \tag{5}$$

$$\mathbf{y}(t) = H_i\mathbf{x}(t) + \mathbf{v}(t), \tag{6}$$

$$r_i(\mathbf{x}(t), \mathbf{u}(t)) = \frac{1}{2}\mathbf{x}(t)^{\mathrm{T}}Q_i\mathbf{x}(t) + \frac{1}{2}\mathbf{u}(t)^{\mathrm{T}}R_i\mathbf{u}(t), \tag{7}$$

where $A_i \in \Re^{N\times N}$ and $B_i \in \Re^{N\times D}$ are regression parameter of the ith linear dynamics, $c_i \in \Re^N$ is bias parameter, and $H_i \in \Re^{L\times N}$ is an observation matrix. $Q_i \in \Re^{N\times N}$ and $R_i \in \Re^{D\times D}$ are parameters of the ith quadratic cost function. Therefore, each state estimator and controller can be represented by a linear model, and the value function estimator can be represented by a quadratic model.

Fig. 2 shows a schematic diagram of the eMOSAIC model. The responsibility predictor derives the responsibility of each module based on the state-prediction accuracy of the state estimator. Final output from the learning system is then derived as the weighted sum of each module's output by the responsibility signal. Below, we explain the details of (I) learning forward models, (II) state estimator, (III) responsibility predictor, (IV) value function estimator, and (V) controller.

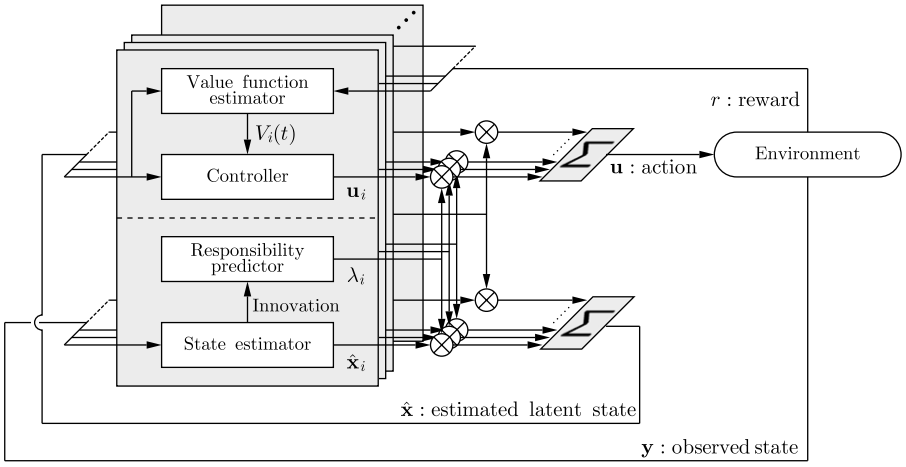

Fig. 2. *e*xtended MOSAIC with state *e*stimators (eMOSAIC).

2.2.1. (I) Learning forward models

The objective of learning in each linear dynamics model is to minimize the weighted prediction error. If each model is represented in a linear form $\mathbf{x}_i(t+1) = W_i\mathbf{z}(t)$,[1] the expected values for the regression parameters W_i can be derived as

$$W_i = \frac{\langle \mathbf{x}\mathbf{z}\rangle_i(T)}{\langle \mathbf{z}\mathbf{z}^{\mathrm{T}}\rangle_i(T)}. \tag{8}$$

The notation $\langle g\rangle_i(T)$ annotates a weighted mean of a function g with respect to the responsibility signal $\lambda_i(t)$:

$$\langle g\rangle_i(T) = \frac{1}{T}\sum_{t=1}^{T} g(t)\lambda_i(t). \tag{9}$$

The responsibility signal is a probability distribution of the module selection (described in Section 2.2.3). By iterating the responsibility signal calculation and parameter update, the likelihood will increase to the suboptimal.

2.2.2. (II) State estimators

The state estimator estimates the latent states of the dynamics from an observation. We use a linear state estimator:

$$\hat{\mathbf{x}}_i(t+1|t) = A_i\hat{\mathbf{x}}_i(t) + B_i\mathbf{u}(t) + c_i, \tag{10}$$

$$\hat{\mathbf{x}}_i(t+1) = \hat{\mathbf{x}}_i(t+1|t) + K_i(\mathbf{y}(t) - H_i\hat{\mathbf{x}}_i(t+1|t)) \tag{11}$$

where $\hat{\mathbf{x}}_i$ is the estimated state and K_i is the parameter of the state estimator. We derive the parameter K_i by solving the linear optimal estimation problem (Kalman & Bucy, 1961; Lewis, 1986) (see Appendix A).

2.2.3. (III) Responsibility predictors

The contribution of each module is represented by probability distribution λ_i, which we call the "responsibility signal". The responsibility signal λ_i is given by the following Bayes' rule:

$$\lambda_i(t) = \frac{P(i)p(\mathbf{x}(t) \mid \mathbf{y}(1:t), i)}{\sum_{i'\in M} P(i')p(\mathbf{x}(t) \mid \mathbf{y}(1:t), i')}, \tag{12}$$

where M is the set of module indices, $p(\mathbf{x}(t) \mid \mathbf{y}(1:t), i)$ is the likelihood of the ith module, and $P(i)$ is the prior. By assuming that the prediction error and estimation error are Gaussian with covariances $\Sigma_{\mathbf{x}}$ and $\Sigma_{\mathbf{y}}$, the likelihood of the ith module $p(\mathbf{x}(t) \mid \mathbf{y}(1:t), i)$ is given by

$$p(\mathbf{x}(t) \mid \mathbf{y}(t), i) \propto p(\mathbf{y}(t) \mid \mathbf{x}(t|t-1), i)p(\mathbf{x}(t|t-1), i), \tag{13}$$

$$p(\mathbf{y}(t) \mid \mathbf{x}(t|t-1), i) = \frac{1}{\sqrt{(2\pi)^L|\Sigma_{\mathbf{y}}|}}\exp\left[-\frac{1}{2}\mathbf{e}_i(t)^{\mathrm{T}}\Sigma_{\mathbf{y}}^{-1}\mathbf{e}_i(t)\right], \tag{14}$$

$$\mathbf{e}_i(t) = \mathbf{y}(t) - H_i\mathbf{x}_i(t|t-1), \tag{15}$$

$$p(\mathbf{x}(t|t-1), i) = p(\mathbf{x}_i(t|t-1) \mid \mathbf{x}_i(t-1), \mathbf{u}(t-1)) = \frac{1}{\sqrt{(2\pi)^N|\Sigma_{\mathbf{x}}|}}\exp\left[-\frac{1}{2}\mathbf{d}_i(t)^{\mathrm{T}}\Sigma_{\mathbf{x}}^{-1}\mathbf{d}_i(t)\right], \tag{16}$$

$$\mathbf{d}_i(t) = \mathbf{x}_i(t) - \{A_i\mathbf{x}_i(t-1) + B_i\mathbf{u}(t-1) + c_i\}, \tag{17}$$

where $\hat{\mathbf{x}}_i(t|t-1)$ is the predicted state of the state estimator of the ith module and $\mathbf{e}_i(t) = \mathbf{y}(t) - H_i\hat{\mathbf{x}}_i(t|t-1)$ is the so-called error of innovation.

Finally, the estimated state is derived by the weighted sum of each module:

$$\hat{\mathbf{x}}(t) = \sum_{i\in M}\lambda_i(t)\hat{\mathbf{x}}_i(t), \tag{18}$$

where $\hat{\mathbf{x}}_i(t)$ is the estimated state of the ith module at time t.

2.2.4. (IV) Value function estimators

We derive the controller by locally solving the linear-quadratic optimal control problem. Since we approximate nonlinear and non-stationary dynamics by multiple linear models in Eq. (5) and the cost function by the quadratic functions in Eq. (7), we can locally estimate the value function by using a quadratic function:

$$V_i(\mathbf{x}(t)) = \frac{1}{2}(\hat{\mathbf{x}}(t) - \mathbf{x}_i^v)^{\mathrm{T}}P_i(\hat{\mathbf{x}}(t) - \mathbf{x}_i^v), \tag{19}$$

where the matrix P_i is given by solving the Riccati equation:

$$0 = P_iA_i + A_i^{\mathrm{T}}P_i - P_iB_iR_i^{-1}B_i^{\mathrm{T}}P_i + Q_i. \tag{20}$$

The center $\mathbf{x}_i^v$ of the ith value function is given by

$$\mathbf{x}_i^v = -(Q_i + P_iA_i)^{-1}P_ic_i. \tag{21}$$

2.2.5. (V) Controllers

By using the estimated value function in Eq. (19), we can derive linear optimal controller for the ith module as (Doya, 2000; Lewis,

[1] W_i and $\mathbf{z}$ represent $[A_i\ B_i\ c_i]$ and $[\mathbf{x}^{\mathrm{T}}\ \mathbf{u}^{\mathrm{T}}\ 1]^{\mathrm{T}}$.

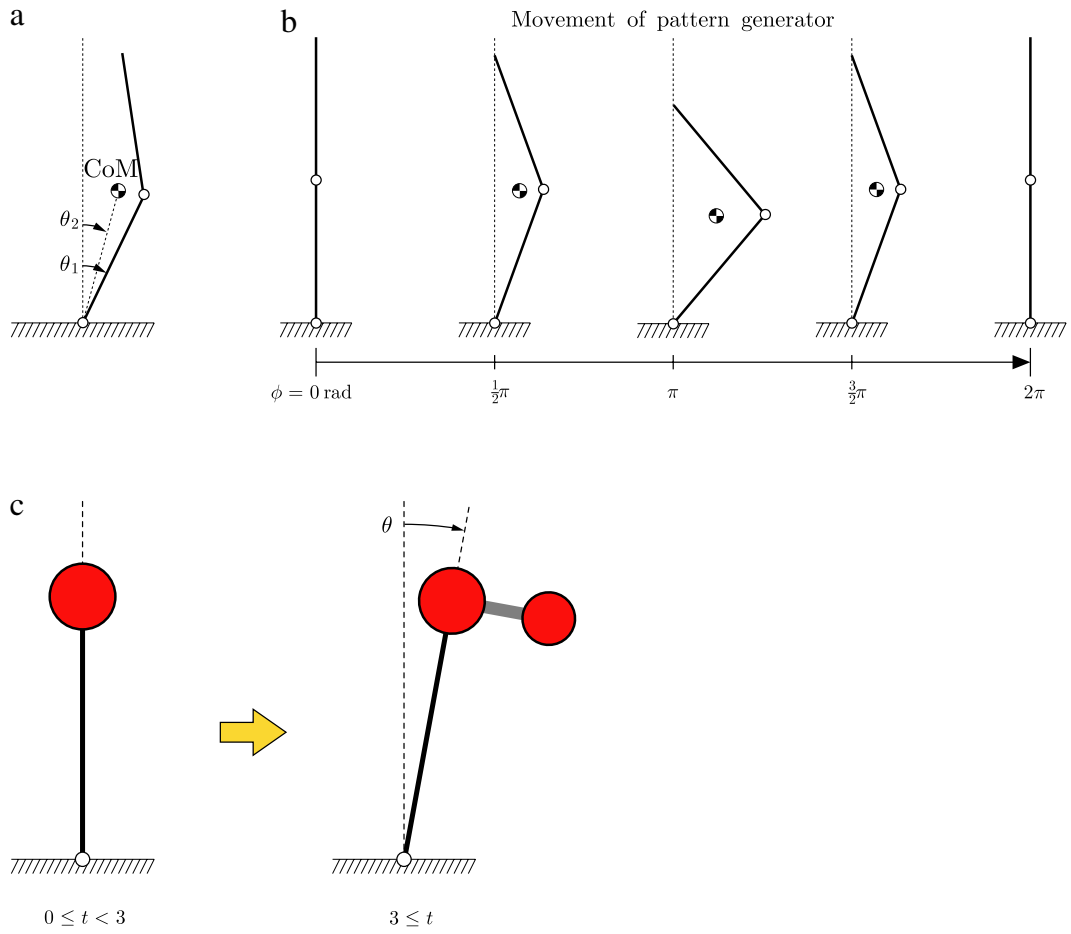

Fig. 3. (a) Two-link robot model. The mass and length of each link are 5 kg and 0.5 m, respectively, and the friction coefficient of each joint is 0.1. (b) Relationship between the phase of the periodic pattern generator ϕ and the posture of the two-link robot model. For details, see Appendix D. (c) Single pendulum model with payload. The masses of weights attaches at the corner and tip are 10 kg and 5 kg, and the lengths of long and short axes are 1.5 m and 0.5 m, respectively.

1986) as:

$$\mathbf{u}_i(t) = -R_i B_i^{\mathrm{T}} P_i(\hat{\mathbf{x}}(t) - \mathbf{x}_i^v). \tag{22}$$

The final output of the controller is derived by the weighted sum of each module output:

$$\mathbf{u}(t) = \sum_{i \in M} \lambda_i(t)\mathbf{u}_i(t). \tag{23}$$

3. Simulations

We evaluated the proposed method by using a simple robot model (see Fig. 3(a) and (c)). The squatting and lifting tasks evaluate the nonlinear and non-stationary control performance, respectively.

3.1. Squatting task

3.1.1. Simulation setup

A basic squatting behavior is provided by a periodic pattern generator. Fig. 3(b) shows the relationship between the phase of the periodic pattern generator (ϕ) and the posture of the two-link robot model (see Appendix D). The pattern generator outputs a simple sinusoidal trajectory to a proportional derivative (PD) controller of the robot. The frequency of the squatting movement is 0.2 Hz. Since a robot flexes and extends its legs periodically, a complex control law is required to prevent it from falling. We apply eMOSAIC to control the two-link robot model so that it can maintain its balance during the squatting. Note that the two-link robot model cannot maintain its balance by only using the output of the periodic pattern generator.

The angle of the lower link and the center of mass (CoM) are represented by θ_1 and θ_2 (see Fig. 3(a)). The input state vector is $\mathbf{x} = [\theta_1\ \theta_2\ \dot{\theta}_1\ \dot{\theta}_2]^{\mathrm{T}}$. The output of eMOSAIC is a relative desired trajectory of the root joint. The sum of the relative desired trajectory and the output of the periodic pattern generator is used to derive torque at each joint based on a PD controller. The torque commands for the actuator are generated by the PD controller which is given by

$$\boldsymbol{\tau} = -K(\mathbf{q} - \mathbf{q}^{\mathrm{d}}), \tag{24}$$

where $\mathbf{q} = [q_{\mathrm{root}}\ q_{\mathrm{middle}}\ \dot{q}_{\mathrm{root}}\ \dot{q}_{\mathrm{middle}}]^{\mathrm{T}}$ represents the position and velocity of the joint angles, $\mathbf{q}^{\mathrm{d}}$ is the desired trajectory, $\boldsymbol{\tau} = [\tau_{\mathrm{root}}\ \tau_{\mathrm{middle}}]^{\mathrm{T}}$ is a torque command for the actuator of the robot, and $K = \begin{bmatrix} 50 & 0 & 10 & 0 \\ 0 & 50 & 0 & 10 \end{bmatrix}$ is a gain matrix of the PD controller.

The cost is given in the quadratic form (Eq. (7)). The parameters of the quadratic cost are $Q_i = \mathrm{diag}\{0, 1, 0, 1\}$ and $R_i = 0.1$ for all modules. One trial lasted 10 s. The simulation and observation time step was 0.002 s.

We evaluated the prediction error, the control cost, and the computation time without the observation and system noise in order to select a number of modules. Each module segmented the squatting movement at regular phase intervals of the pattern generator from $\phi = 0$ (standing position) to $\phi = \pi$ (crouching

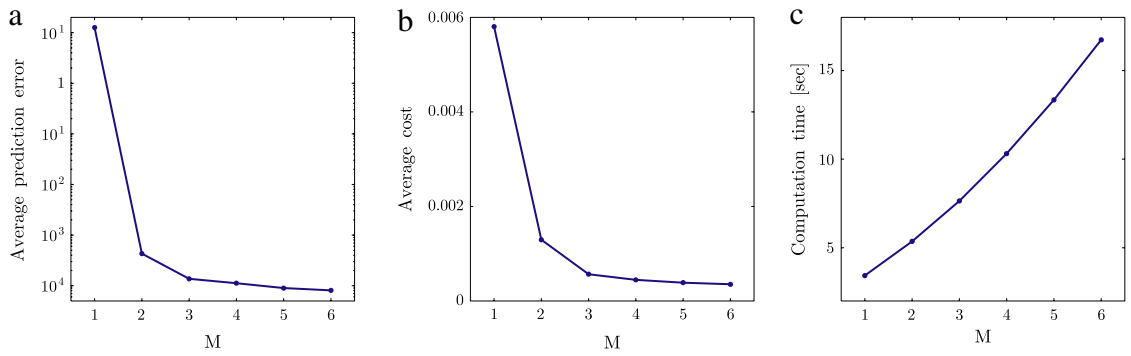

Fig. 4. Evaluation result of (a) average prediction error, (b) average cost, and (c) computation time for different number of modules, respectively.

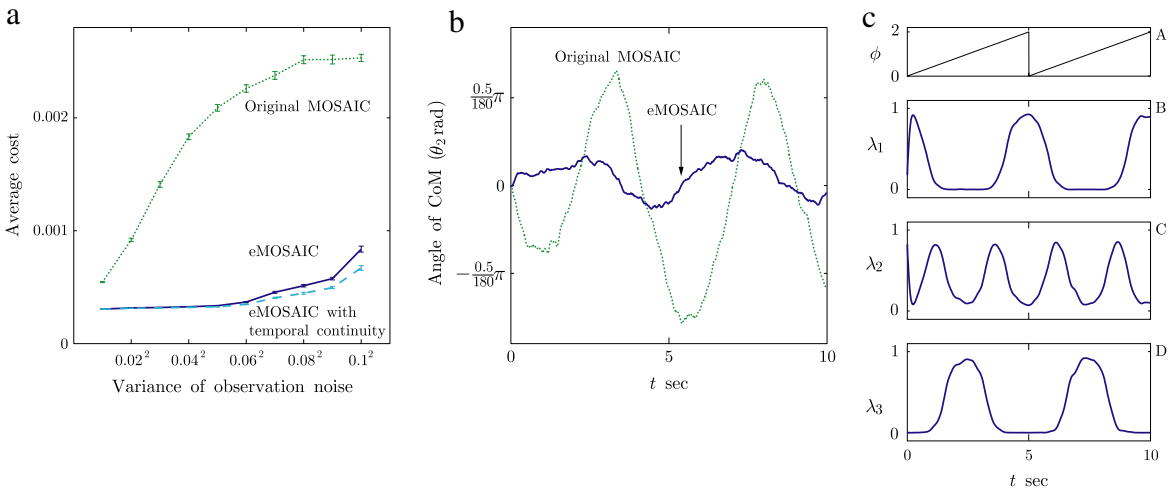

Fig. 5. Results of squatting task. (a) Average costs with different observation noise. The means and standard deviations over 100 simulation runs are plotted. (i) The dotted line indicates the result of original MOSAIC. (ii) The dashed line indicates the results of eMOSAIC without using responsibility predictor. (iii) The solid line indicates the results of the proposed method. (b) The trajectory of CoM (θ_2). The dotted and solid lines indicate the original MOSAIC and eMOSAIC. (c) The responsibility signal of the proposed method (iii) is shown. Panel A indicates the phase of the periodic pattern generator, and panels B–D indicate the responsibility signals of the three modules. The low-passed signals are plotted with a cutoff frequency of 10 Hz.

position). For example, in the case of three modules ($M = 3$), each forward model predicts the dynamics around $\phi = 0, \frac{1}{2}\pi, \pi$. Then, for each module, we sampled trajectory around the corresponding posture by using random Gaussian noise as a control input and estimated the parameters of forward model (see Eq. (8)). Fig. 4 shows the relationships between the number of modules and average prediction error, average control cost, and computation time for one trial. The prediction error and the control cost decreased drastically from $M = 1$ to $M = 3$ while significant differences between these values with $M = 3$ and these with $M = 6$ were not observed. On the other hand, computation time linearly increased when the number of modules increased. Therefore, we decided to use three modules for this task. The acquired parameters of the linear models are presented in Appendix B.

In this simulated environment, we focused on showing three advantages of the eMOSAIC model: (1) it can be applied to an environment with large observation noise (see Section 3.1.2), (2) it can be applied to a partially observable system (see Section 3.1.3), and (3) it can cope with modeling errors (see Section 3.1.4).

3.1.2. Robustness against large observation noises

First we evaluated the robustness of eMOSAIC against large observation noises. For this comparison, we considered two methods: original MOSAIC and eMOSAIC.

We tested the control performance of two methods with observation noises: $\Sigma_{\mathbf{y}} = \sigma^2_{\text{obs}} I$ where $\sigma^2_{\text{obs}} = 0\text{–}0.1^2$. Fig. 5(a) shows the relationship between the size of the observation noise and the average cost in the squatting task. The mean and standard deviation of the cost over 100 simulation runs are plotted. Dotted and solid lines represent results acquired by the two methods, original MOSAIC and eMOSAIC, respectively. The average cost of the original MOSAIC rapidly increased based on according to the size of the observation noise. In contrast, the average cost of eMOSAIC was relatively insensitive to the size of the observation noise. These results indicate that eMOSAIC is robust against observation noise.

Next, we evaluated the effect of the prior information for the module selection. If the linear modules are sparsely allocated, the responsibility signal tends to chatter around a border of the local state spaces specified by the linear models, due to the softmax calculation of Eq. (12). Chattering of the responsibility signals causes damage to the hardware in the real environment. In order to avoid the hardware damage, the temporal continuity, which assumes that the modules switch smoothly, as prior information of module selection can be introduces. We describe the details of the temporal continuity in Appendix C. The natural frequency of the pendulum represented by CoM at the standing position ($\phi = 0$) is 0.705 Hz. We empirically found that the time constant parameter of the temporal continuity ρ around 10–20 times of the natural frequency leads good control performance. In this simulation, we selected the time constant parameter of the temporal continuity as a 15 times of the natural frequency ($\rho = \frac{1}{15\times 0.705}$). Dashed

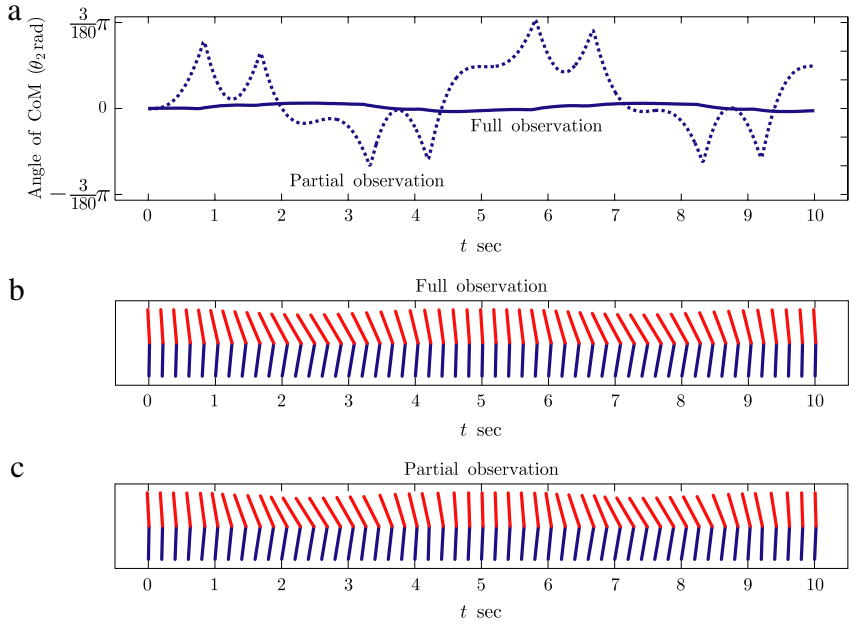

Fig. 6. Results of squatting task in full observation and partial observation environments. (a) Trajectory of CoM (θ_2). The solid and dashed lines represent full and partial observation cases, respectively. (b), (c) Squatting movements of full and partial observation cases (see *Movie* 1).

line represents the performance of eMOSAIC with the temporal continuity as the prior information. The performance was slightly better than the case of without the temporal continuity. This result indicates that introduction of the temporal continuity dose not significantly deteriorate control performance. Note that we use $P(i) = 1/M$ where $M = 3$ as prior information for the case without using the temporal continuity.

Fig. 5(b) plots the trajectory of θ_2 (CoM angle) with the observation noise with variance $\sigma^2_{\text{obs}} = 0.02^2$. The dotted and solid lines represent control performance of the original MOSAIC and the eMOSAIC, respectively. eMOSAIC showed better performance in maintaining balance than the original MOSAIC. Fig. 5(c)-A show the phase of the periodic pattern generator (ϕ), and Fig. 5(c)-B–D show the responsibility signal of each module ($i = 1, 2, 3$) estimated by eMOSAIC. eMOSAIC was able to switch each module periodically for the periodic squat movement.

Through this simulation we demonstrated that the robot model could maintain balance and switch the modules in the noisy environment using eMOSAIC. These results indicate that the adaptation capability of the MOSAIC architecture can be implemented in a real environment.

3.1.3. Application to partially observable environments

In the real environment, there is no guarantee that all states can be measured. Here, we apply eMOSAIC to both a fully observable and partially observable system (only joint angles can be observed, i.e., $H_i = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$ in Eq. (6)).

Fig. 6(a) plots the trajectories of CoM (θ_2) in the fully observable and partially observable system. In the case of full observation, the angle of CoM could be maintained around zero. Table 1 shows the mean squared errors, and Fig. 6(b) and (c) show the sequence of the squatting movements for full observation and partial observation. In the both cases, eMOSAIC was able to maintain the balance of the two-link robot (see *Movie* 1). In the partially observable system, however, the CoM angle deviated from zero, and the robot could avoid failure even if the angular velocities of each joint could not be observed.

Note that the two-link robot model cannot maintain its balance by only using the output of the periodic pattern generator.

Table 1
Mean squared errors of squatting task in fully observable and partially observable system.

	Full observation	Partial observation
MSE	3.66×10^{-6}	4.18×10^{-4}

3.1.4. Robustness against modeling errors

For the real system it is difficult to precisely estimate physical parameters. Here, we evaluate the proposed method's robustness against modeling errors and compare the robustness of the controller with that of a standard nonlinear tracking controller using feedback linearization (FL) (Slotine & Li, 1991) (see Appendix E).

FL cannot balance the CoM of the robot with the desired trajectory used by eMOSAIC since that was a simple sinusoidal trajectory (see Fig. 3(b)). Therefore we modified the desired trajectory of FL ($\mathbf{q}^{\text{des}}$) to maintain the angle of CoM (θ_2) zero precisely. (For details of the modified desired trajectory, see Appendix D.) We optimized the parameter κ that represents the convergence property of the tracking error in order to minimize the tracking error: $\kappa = 5.88$ (see also Appendix D).

To simulate the modeling errors, we multiplied the parameter α to actual mass and length of the robot links. Thus $\alpha = 1$ means a zero modeling error. We evaluated the robustness of both methods against the modeling errors ($\alpha = 0.6$–2). The variance of the system and the observation noises ware fixed to small values, $\sigma^2_{\text{sys}} = 0.001^2$ and $\sigma^2_{\text{obs}} = 0.01^2$. The results are plotted in Fig. 7, in which the horizontal and vertical axes are the parameter α and the mean of the CoM angle (θ_2). The eMOSAIC shows better tracking performance than FL except for the case of $\alpha \neq 1$ in which there is no modeling error. In the implementation of eMOSAIC we derived the controller of each module as an optimal linear feedback controller. It is well known that the optimal linear feedback control law reduces the sensibility for the modeling error (Cruz & Perkins, 1964; Fujii & Narazaki, 1984; Perkins & Cruz, 1971). The tracking error of FL was significantly large when the parameter α was less than one. This is because FL output the smaller than required torque, and then failed to maintain the balance.

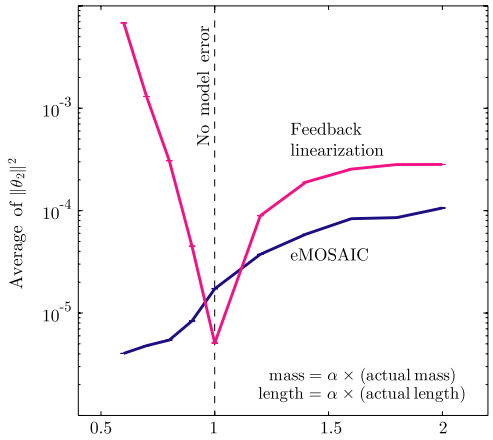

Fig. 7. Results of robustness test of eMOSAIC and feedback linearization. The horizontal axis is the parameter α defining a modeling error. The vertical axis is the angle of CoM (θ_2). The mean and standard deviation of 50 trials are plotted.

3.2. Lifting task

3.2.1. Simulation setup

Here, we consider the lifting task as an example of a non-stationary control problem. Fig. 3(c) shows single-pendulum models with and without payload. For the pendulum model with the payload, the weights attached at the corner and tip represented the mass of the robot's body and the payload. The task was to keep the balance in response to the existence and absence of the payload.

One trial was 6 s and the payload was attached during the last 3 s. The state vector is $\mathbf{x} = [\theta\ \dot{\theta}]^{\mathrm{T}}$, and the output is the desired trajectory of the joint. The torque command for the actuator is generated by the PD controller given by

$$\tau = -K(\mathbf{x} - \mathbf{x}^{\mathrm{des}}), \tag{25}$$

where $\mathbf{x}^{\mathrm{des}}$ is the desired trajectory, τ is a torque command for the actuator, and $K = [50\ 10]$ is the gain parameters of the PD controller.

The cost is given by a quadratic formation:

$$r(t) = \frac{1}{2}\mathbf{x}(t)^{\mathrm{T}}Q\mathbf{x}(t) + \frac{1}{2}\mathbf{u}(t)^{\mathrm{T}}R\mathbf{u}(t), \tag{26}$$

where $Q = \mathrm{diag}\{1, 1\}$ and $R = 0.1$. Thus the minimum cost was given at upright position $\theta = 0$ rad.

Since there are two conditions: with and without payload, we use two modules ($i = 1, 2$). The first and second modules predict a single-pendulum model without and with the payload, respectively. We sampled data around a standing position by using Gaussian random input absence and presence of the payload for the first and second modules respectively and estimated the parameters of forward model (see also Eq. (8)). Obtained parameters are showed in Appendix B. We again test the performance of the two control methods: original MOSAIC and eMOSAIC. We evaluate the robustness of two methods with observation noises ($\Sigma_{\mathbf{y}} = \sigma^2_{\mathrm{obs}} I$ with $\sigma^2_{\mathrm{obs}} = 0$–$0.01^2$).

3.2.2. Results

Fig. 8(a) shows the relationship between the observation noise and the average cost in the lifting task. Dotted and solid lines represent the performance of original MOSAIC and eMOSAIC. The proposed method shows the higher performance than the original MOSAIC. The performances of the original MOSAIC decreased drastically with large observation noise while eMOSAIC did not.

We again evaluated the effects of the temporal continuity. Dashed line in Fig. 8(a) represents the performance of eMOSAIC with the temporal continuity. The natural frequency of single-pendulum without payload is 0.407 Hz, we also selected the time constant parameter ρ as 15 times of the natural frequency ($\rho = \frac{1}{15\times 0.407}$). The performance was better than the case of without the temporal continuity. This result indicates that our proposed method can switch module fast enough even with the temporal continuity with the selected time constant parameter.

Fig. 8(b) and (c) represent the responsibility signals of original MOSAIC and eMOSAIC. Original MOSAIC was not able to switch the module according to the payload placement due to the large observation noise. These results indicate that the state estimator for the responsibility predictor is essential for coping with noisy non-stationary environments.

4. Real robot experiments

Both for the squatting task and lifting task, we implemented eMOSAIC for the CB-i real robot (see Fig. 1).

4.1. Squatting task

4.1.1. Experimental setup

We sought to maintain the balance of the CB-i in the squatting task. Fig. 9(a) shows the pitch-joint coordination of the CB-i. A periodic pattern generator output the desired trajectories to the hip, knee, and ankle joints (see Appendix F).

We approximated the dynamics of CB-i using a two-link robot with a foot model (Atkeson & Stephens, 2007; Hyon, Osu, & Otaka, 2009; Stephens, 2007) (Fig. 9(b)). The input state is $\mathbf{x} = \left[\theta_{\mathrm{torso}}\ \theta_{\mathrm{ankle}}\ \theta_{\mathrm{gyro}}\ \dot{\theta}_{\mathrm{torso}}\ \dot{\theta}_{\mathrm{ankle}}\ \dot{\theta}_{\mathrm{gyro}}\right]^{\mathrm{T}}$. The outputs of eMOSAIC are the relative desired torso and hip joint angles: $\mathbf{u} = \left[\theta^{\mathrm{add}}_{\mathrm{torso}}\ \theta^{\mathrm{add}}_{\mathrm{ankle}}\right]^{\mathrm{T}}$. These angles are added to the desired joint angles generated by the periodic pattern generator. To follow the desired joint angles, the torque output at each joint is derived by a PD controller:

$$\boldsymbol{\tau} = -K(\mathbf{q} - \mathbf{q}^{\mathrm{d}}), \tag{27}$$

where $\mathbf{q} = [q_{\mathrm{torso}}\ q_{\mathrm{hip}}\ q_{\mathrm{knee}}\ q_{\mathrm{ankle}}\ \dot{q}_{\mathrm{torso}}\ \dot{q}_{\mathrm{hip}}\ \dot{q}_{\mathrm{knee}}\ \dot{q}_{\mathrm{ankle}}]^{\mathrm{T}}$ represents the joint angles' position and velocity, $\mathbf{q}^{\mathrm{d}}$ is the desired trajectory, $\boldsymbol{\tau} = [\tau_{\mathrm{torso}}\ \tau_{\mathrm{hip}}\ \tau_{\mathrm{knee}}\ \tau_{\mathrm{ankle}}]$ is torque commands for the robot's actuators, and K is a gain matrix of the PD controller.

Here, the purpose of each linear optimal controller is to keep the pitch angle of the gyro sensor close to zero $\theta_{\mathrm{gyro}} = 0$, where the sensor is attached to the hip of CB-i (see Fig. 9(b)). The cost function is defined by quadratic form (see Eq. (7)). The parameter for state deviation is given as $Q_i = \mathrm{diag}\{0, 0, 1, 0, 0, 1\}$ to minimize the angle of the gyro (θ_{gyro}) and the angular velocity of the gyro ($\dot{\theta}_{\mathrm{gyro}}$), and the parameter for output deviation is given as $R_i = \mathrm{diag}\{0.01, 0.01\}$ ($i = 1, 2, 3$) to minimize the control cost.

We prepared three modules: $i = 1, 2, 3$, same as a simulation of squatting task. The forward models of the first, second, and third modules estimated states around three different postures. We sampled data around the posture corresponding to the phase of $\phi = 0, \frac{\pi}{2}, \pi$ by using smoothed Gaussian random input and estimated the parameters of forward models (see also Eq. (8)).

Since frequent module switching cause damage to the hardware, we introduced temporal continuity of the responsibility signal (see also Appendix C). The height of CoM of CB-i at a standing position is about 0.98 m, the natural frequency of the pendulum model represented by CoM and the center of pressure (CoP) is 0.504 Hz. By following the simulation results, we selected the time constant parameter of the temporal continuity as a 15 times of the natural frequency ($\rho = \frac{1}{15\times 0.504}$).

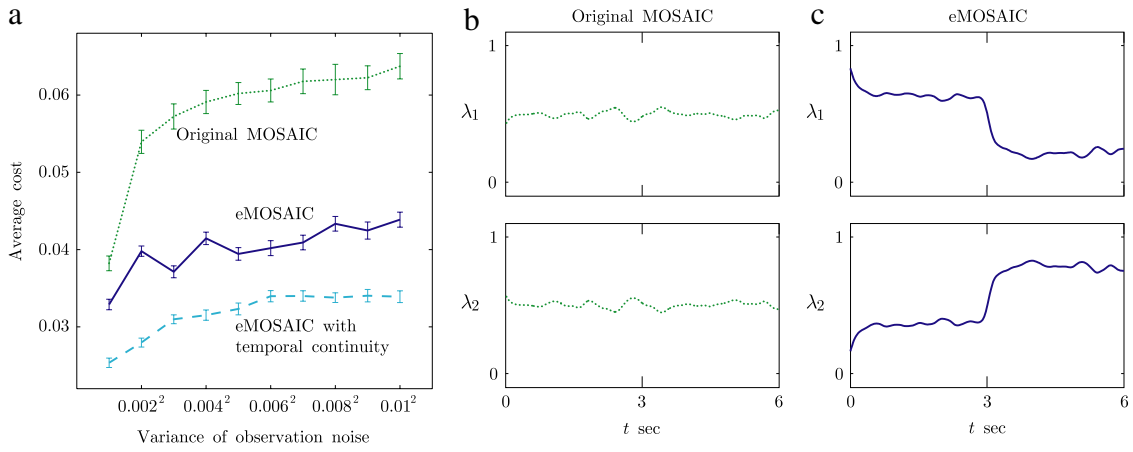

Fig. 8. Results of lifting task. (a) Average costs with different observation noise are displayed in the same format as in Fig. 5. Figs. (b) and (c) show the results obtained by original MOSAIC and eMOSAIC. Each figure shows the responsibility signals of the module. The low-passed signals are plotted with a cutoff frequency of 10 Hz.

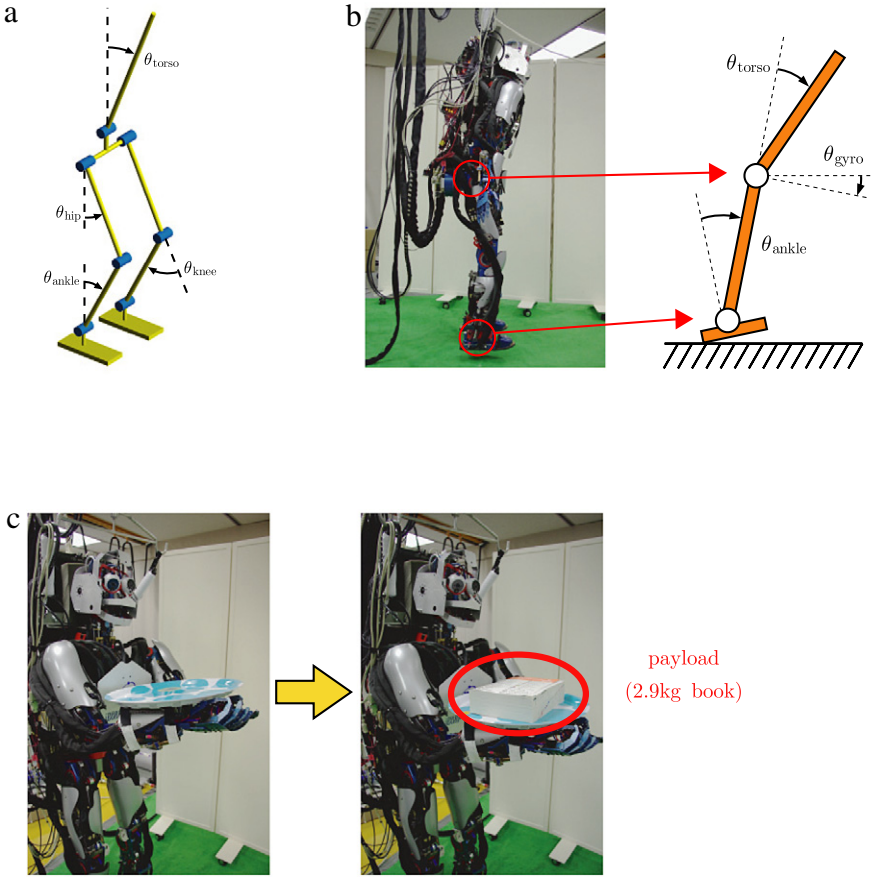

Fig. 9. (a) Joint coordinates of humanoid robot CB-i. (b) Two-link robot with foot model of the CB-i. The two joints of the two-link model represent the torso and ankle joint of the CB-i. The gyro sensor is attached to its hip. (c) Lifting task. Payload mass is 2.9 kg. Left: without payload. Right: with payload.

4.1.2. Results

Fig. 10 shows the results of the 0.5 Hz squatting task. Fig. 10(a)-A shows the phase of the periodic pattern generator (ϕ). Fig. 10(a)-B shows the trajectory of the pitch angle of the gyro sensor (θ_{gyro}). The solid line represents the eMOSAIC result (see also *Movie* 2). The pitch angle of the gyro sensor (θ_{gyro}) is maintained approximately in the range of $\left[-\frac{2}{180}\pi, \frac{2}{180}\pi\right]$, within which the robot does not fall over. The dashed line represents the result obtained by using only the first module (see also *Movie* 3). At approximately $t = 3$ s, the robot in which only the single module was used fell backward. By using the multiple-module, we successfully balanced the robot because the second and third modules were subsequently selected. Fig. 10(a)-C–E show the responsibility signals of three modules. Fig. 10(b) shows the squatting movements by using multiple modules. The upper body of the CB-i was stable even if the lower body was flexed and extended. In contrast, in the case of a single module (see Fig. 10(c)), the upper body moved backward and forward. The robot could not stand without support. The robot was able to maintain the balance with squatting frequency up to 1.5 Hz by using the eMOSAIC model (see also *Movie* 4).

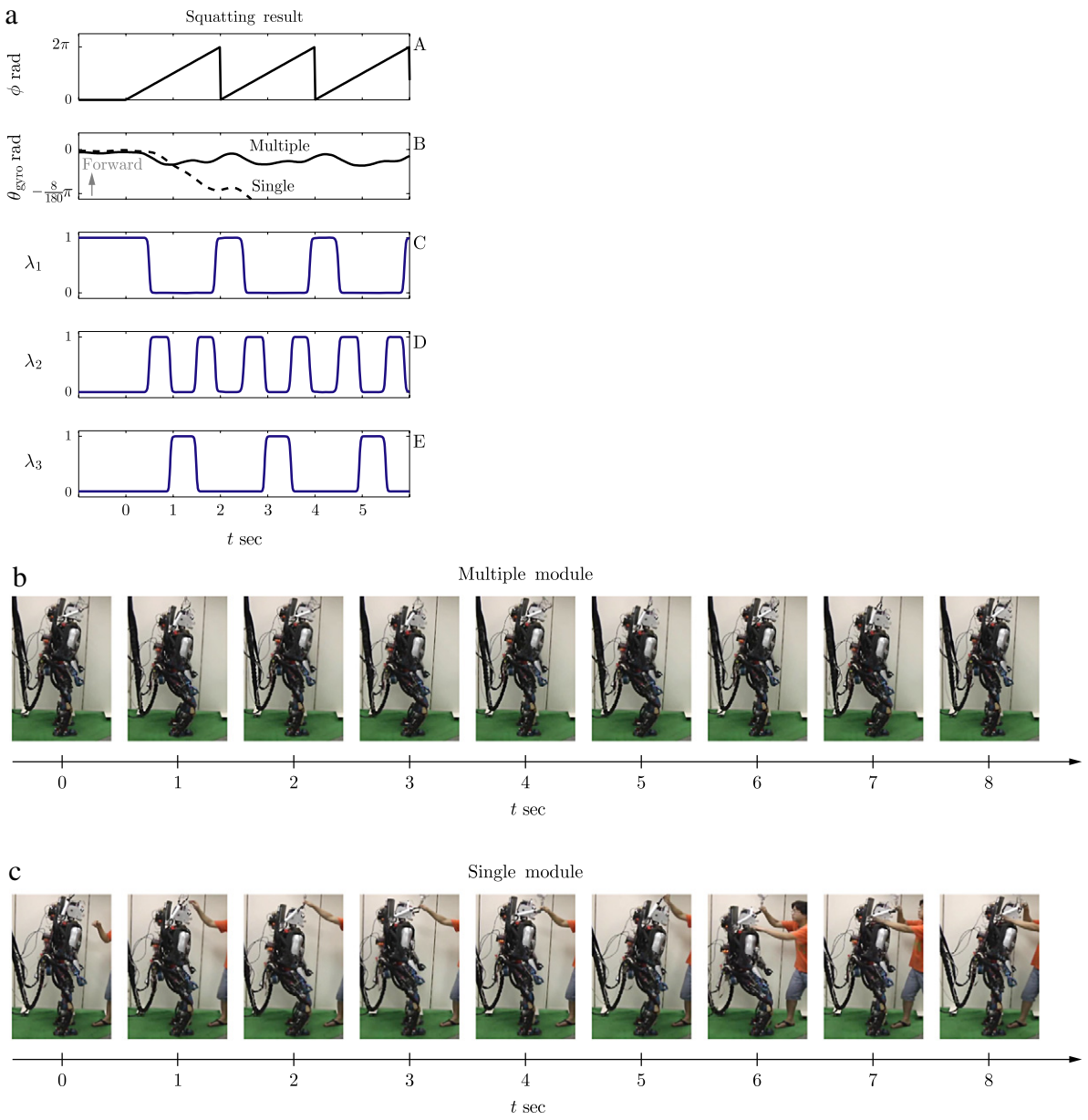

Fig. 10. Results of 0.5 Hz squatting task in real-robot experiments. (a) Panel A shows the phase of the periodic pattern generator (ϕ). Panel B shows the pitch angle of the gyro sensor. The solid line represents the results of the proposed method (*Movie* 2), and the dashed line represents the results obtained by using only the first module (*Movie* 3). The other panels C, E show the responsibility signals of the proposed method. (b), (c) Squatting movement of 0.5 Hz by using multiple modules and single module, respectively. The robot was able to maintain balance during the squatting task with frequency up to 1.5 Hz (see also *Movie* 4).

4.2. Lifting task

4.2.1. Experimental setup

We tried to maintain the balance of the CB-i in the lifting task (Fig. 9(c)). Again, we approximated the dynamics of our humanoid robot by using the two-link robot with a foot model (Fig. 9(b)). We used two modules ($i = 1, 2$) in eMOSAIC. The first and the second models corresponded to the state of the CB-i absence and presence of the payload, respectively. We again sampled data by using smoothed Gaussian random input and estimated the parameters of forward model (see also Eq. (8)). For the module selection, we assumed the temporal continuity of the responsibility signal. We selected the same time constant parameter to the squatting task ($\rho = \frac{1}{15\times 0.504}$).

In this task, each linear optimal controller tried to balance the pitch angle of the gyro. Therefore, we used the quadratic cost function the same as for the squatting task ($Q_i = \text{diag}\{0, 0, 1, 0, 0, 1\}$ and $R_i = \text{diag}\{0.01, 0.01\}$, $i = 1, 2$).

4.2.2. Results

Fig. 11(a) shows the results of the lifting task. Fig. 11(a)-A shows the pitch angle of the gyro sensor. We applied a payload to the tray at a time of $t = 0$ s. The solid and dashed lines represent the proposed method (see also *Movie* 5) and the single-module method (see also *Movie* 6), respectively. The robot with a single module fell forward at approximately $t = 3$ s. However, by the proposed method, the pitch angle of the gyro sensor θ_{gyro} was successfully maintained at less than $\frac{2}{180}\pi$ rad. Fig. 11(a)-B and C show the responsibility signal of the modules for no-payload and payload, respectively. First, the proposed method selected the first module, which controls the environment without a payload. After applying the payload, the second module, which controls the environment

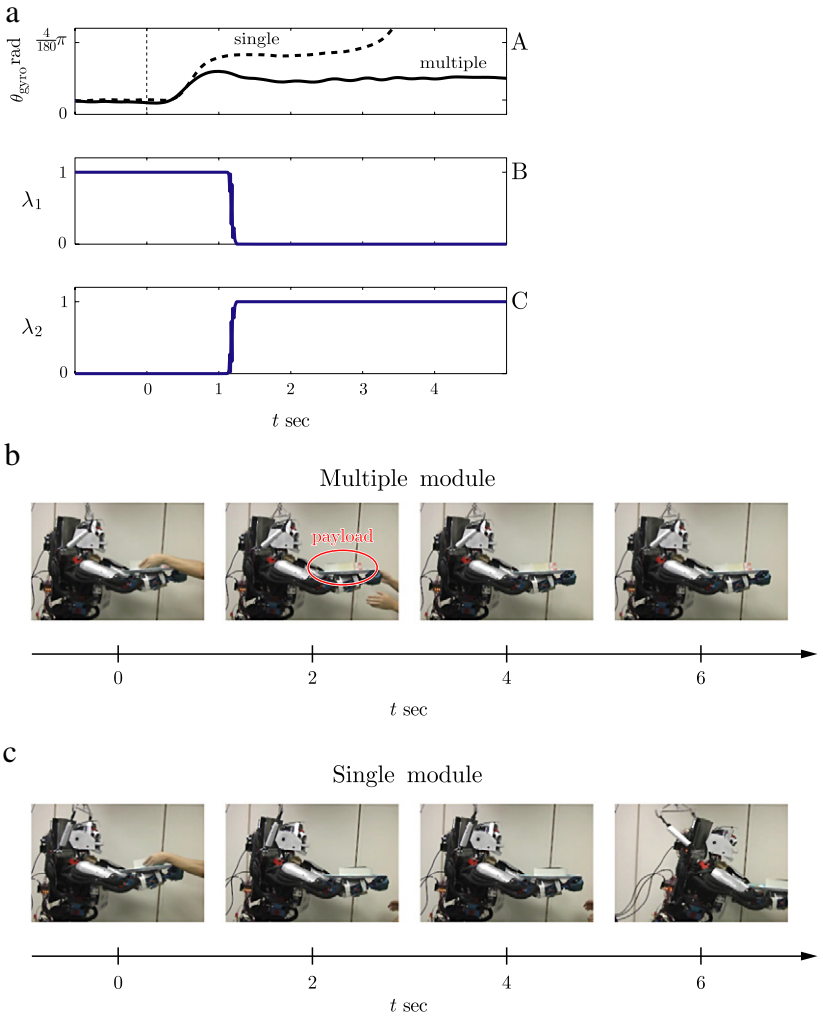

Fig. 11. Results of lifting task in real-robot experiments. (a)-A The pitch angle of the gyro sensor in the multiple-module and single-module cases (see also *Movie* 5 and *Movie* 6). B and C show the responsibility signals of the proposed method. (b), (c) The movement of the lifting payload by using multiple modules and single module, respectively. The robot was able to maintain the balance with module switching even if the payload was applied, but it could not maintain balance without module switching.

when a payload is used, was automatically selected. Fig. 11(b) and (c) show the sequence of the movements during the lifting task. The robot was able to maintain balance even when the payload was applied, but could not maintain balance without module switching.

5. Conclusion

We have extended MOSAIC architecture by using state estimators. We compared the proposed method with the previous methods with large observation noise, partially observable environment, and modeling errors. The simulation results indicate that our extended MOSAIC architecture improves robustness against large observation noise and partial observation. We farther investigated robustness of our proposed method against modeling errors by comparing with the feedback linearization method. We also showed that we can roughly estimate proper number of modules by checking the average prediction errors. The parameters of each module can be estimated from generated data by using Gaussian random input. We then implemented our proposed method on the CB-i humanoid robot. We tried to stabilize the CB-i in a squatting task and lifting task. In these tasks, the CB-i was not able to maintain balance by only using the single module. Our proposed method however successfully stabilized the robot.

The squatting task is suitable to evaluate the control performance of nonlinear control methods. Grimes et al. (2006) proposed a imitation leaning algorithm and evaluated the algorithm on the squatting task. They transferred the captured human motion to the humanoid robot via a latent state space composed of low-dimensional posture information, gyro sensor information, and foot pressure information. However, this imitation learning method can only be used to generate feed-forward squatting trajectories. Therefore, the control performance can be susceptible to external disturbances. On the other hand, our proposed method consists of multiple linear feedback controllers. Thus, our proposed controller can be robust and suitable to generate fast movements. Nakaoka et al. (2004) proposed a motion planning algorithm and the evaluation includes squatting task. In this motion planning method, a whole body motion sequence can be composed by combining motion primitives. Since the motion primitives need to be designed for a particular motion sequence, generalization performance of this planning method can be limited. On the other hand, our proposed method uses linear dynamical systems as primitive representations. Therefor, the primitives do not directly depend on a particular movement and the proposed method may have better generalization performance.

Using modular architecture is a practical approach toward dealing with a large state space. Many robotics and machine learning studies proposed modular control methods (Doya et al., 2002; Haruno et al., 2001; Morimoto & Doya, 2001; Samejima, Doya, & Kawato, 2003; Wiering & Schmidhuber, 1997). "Macro actions" (Hauskrecht, Meuleau, Kaelbling, Dean, & Boutilier, 1998;

Sutton, Precup, & Singh, 1999) or "sub-goals" (Dietterich, 2000; Morimoto & Doya, 2001) have been used in hierarchical learning systems. However, in the most of the previous works, each module is represented by complex nonlinear functions. In this study, each module is represented by a linear model. Therefore, parameter for the controller can be solved. Thus, proposed method can be considered as more practical approach than the previous methods.

The cost function cannot be always modeled by a quadratic form for all tasks. In such cases it is necessary to approximate the complex cost function. As a future study, we will consider a hierarchical learning algorithm. In the hierarchical system, the higher layer can be used to properly allocate local quadratic cost functions to approximate the complex cost function.

Acknowledgments

This research was supported by the Strategic Research Program for Brain Sciences (SRPBS). This work was partially supported by MEXT KAKENHI 23120004. The authors gratefully acknowledge N. Nakano for assistance with the experimental setup.

Appendix A. Linear optimal state estimator

The linear optimal state estimator (i.e., the Kalman filter) assumes that dynamics and observation are linear (see Eqs. (5) and (6)), and system noise and observation noise ($\mathbf{n}(t)$ and $\mathbf{v}(t)$) are zero-mean Gaussian. The optimal state estimator estimates latent state by alternating two phases: prediction and update (Thrun, Burgard, & Fox, 2005). The prediction phase is as follows:

$$\mathbf{x}(t+1|t) = A\mathbf{x}(t|t) + B\mathbf{u}(t) + c \tag{A.1}$$

$$\Sigma(t+1|t) = A\Sigma(t|t)A^{\mathrm{T}} + \Sigma_{\mathbf{x}}, \tag{A.2}$$

where $\Sigma_{\mathbf{x}}$ is the covariance of system noise, $\mathbf{x}(t+1|t)$ and $\Sigma(t+1|t)$ are the mean and covariance of the predicted latent state at time $t+1$ based on the information of previous step t. A, B and c are the parameter of linear dynamics in the same format as Eq. (5). The update phase is as follows:

$$K = \Sigma(t+1|t)H^{\mathrm{T}}(H\Sigma(t+1|t)H^{\mathrm{T}} + \Sigma_{\mathbf{y}})^{-1}, \tag{A.3}$$

$$\mathbf{x}(t+1) = \mathbf{x}(t+1|t) + K(\mathbf{y} - H\mathbf{x}(t+1|t)), \tag{A.4}$$

$$\Sigma(t+1) = (I - KH)\Sigma(t+1|t), \tag{A.5}$$

where $\Sigma_{\mathbf{y}}$ is the covariance of observation noise.

Appendix B. Parameters of linear models

Table 2 shows parameters of linear models (A_i, B_i, and c_i), which we obtained for squatting task in the real robot environment. Table 3 also shows obtained parameters of linear models for lifting task in the real robot environment.

Appendix C. Prior distribution of responsibility predictor

We introduce the prior probability $P(i)$ based on "temporal continuity", for the module selection. The prior can be given by

$$P(i) \propto \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left[-\frac{1}{2\sigma^2}E_i(t-1)\right]. \tag{C.1}$$

E_i represents a smoothed square error (weighted sum from $t=0$ to current time) of innovation $\mathbf{e}_i(t)$ at time t:

$$E_i(t) = \sum_{s=0}^{t} \exp\left[-\frac{(t-s)\Delta t}{\tau}\right] \|\mathbf{e}_i(s)\|^2 \Delta t, \tag{C.2}$$

where $0 < \rho$ is a parameter that controls the strength of the temporal continuity and Δt is a time step of the observation. Eq. (C.2) can be expanded to the following recursive form:

$$E_i(t) = \|\mathbf{e}_i(t)\|^2 \Delta t + \exp\left[-\frac{\Delta t}{\rho}\right] E_i(t-1). \tag{C.3}$$

Appendix D. Periodic pattern generator for two-link robot model

The periodic pattern generator for the two-link robot model (see Fig. 3) outputs a simple sinusoidal trajectory at each joint:

$$q_1^{\mathrm{d}} = D\frac{1-\cos\phi}{2} + q_1^{\mathrm{rest}}, \tag{D.1}$$

$$q_2^{\mathrm{d}} = -2D\frac{1-\cos\phi}{2} + q_2^{\mathrm{rest}}, \tag{D.2}$$

where q_1^{d} and q_2^{d} are the desired trajectory of the first and second joints, q_1^{rest} and q_2^{rest} are rest positions, and D is amplitude of a periodic movement.

The modified desired trajectory of the first joint which is used by the feedback linearization is given by

$$q_1^{\mathrm{d}} = -\arctan\frac{M_2 L g_2 \sin q_2}{M_1 L g_1 + M_2 L_1 + M_2 L g_2 \cos q_2}. \tag{D.3}$$

The two-link robot can maintain a balance of CoM (θ_2) by tracking this desired trajectory.

Appendix E. Feedback linearization

Feedback linearization (FL) uses an inverse dynamics model to cancel nonlinear terms of original dynamics. Linear error dynamics are then introduced. FL considers following nonlinear dynamics:

$$H(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + g(\mathbf{q}) = \boldsymbol{\tau}, \tag{E.1}$$

where $\mathbf{q} = [q_1\ q_2\ \cdots]^{\mathrm{T}}$ is the position in a general coordinate, and $\boldsymbol{\tau} = [\tau_1\ \tau_2\ \cdots]^{\mathrm{T}}$ is the control output. The control law to track the desired trajectory $\mathbf{q}^{\mathrm{d}}$ is given by

$$\boldsymbol{\tau} = H(\mathbf{q})\mathbf{v} + C(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + g(\mathbf{q}), \tag{E.2}$$

where $\mathbf{v}$ is the equivalent input:

$$\mathbf{v} = \ddot{\mathbf{q}}^{\mathrm{d}} - 2\kappa\dot{\tilde{\mathbf{q}}} + \kappa^2\tilde{\mathbf{q}}, \tag{E.3}$$

$\tilde{\mathbf{q}} = \mathbf{q} - \mathbf{q}^{\mathrm{d}}$ is the position tracking error and κ is positive constant (Slotine & Li, 1991).

Appendix F. Squatting controller

The squatting controller outputs desired trajectories at the hip, knee and ankle joints. Each desired trajectory ($\theta_{\mathrm{hip}}^{\mathrm{d}}$, $\theta_{\mathrm{knee}}^{\mathrm{d}}$, and $\theta_{\mathrm{ankle}}^{\mathrm{d}}$) is given by following the central pattern generator (CPG),

$$\theta_{\mathrm{hip}}^{\mathrm{d}} = D\frac{1-\cos\phi(t)}{2} + \theta_{\mathrm{hip}}^{\mathrm{rest}}, \tag{F.1}$$

$$\theta_{\mathrm{knee}}^{\mathrm{d}} = 2D\frac{1-\cos\phi(t)}{2} + \theta_{\mathrm{knee}}^{\mathrm{rest}}, \tag{F.2}$$

$$\theta_{\mathrm{ankle}}^{\mathrm{d}} = D\frac{1-\cos\phi(t)}{2} + \theta_{\mathrm{ankle}}^{\mathrm{rest}}, \tag{F.3}$$

where D is the amplitude of a squatting movement. $\theta_{\mathrm{hip}}^{\mathrm{rest}}$, $\theta_{\mathrm{knee}}^{\mathrm{rest}}$ and $\theta_{\mathrm{ankle}}^{\mathrm{rest}}$ define the rest posture. ϕ is the phase of the periodic pattern generator.

Appendix G. Supplementary data

Supplementary material related to this article can be found online at doi:10.1016/j.neunet.2012.01.002.

Table 2
Linear model parameters (A_i, B_i, and c_i) for squatting task in the real robot environment.

i	A_i	B_i	c_i
1	$\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 \\ -0.0801 & -0.0043 & -0.0054 & 0.9947 & 0.0012 & 0.0001 \\ 0.0045 & -0.2328 & -0.0511 & -0.0011 & 0.9814 & -0.0124 \\ -0.0103 & 0.0668 & -0.0093 & -0.0003 & -0.0018 & 0.9996 \end{bmatrix}$	$\begin{bmatrix} 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0023 & 0.0003 \\ 0.0002 & 0.0059 \\ 0.0001 & -0.0001 \end{bmatrix}$	$\begin{bmatrix} 0.0000 \\ 0.0000 \\ 0.0000 \\ -0.0175 \\ -0.0202 \\ 0.0046 \end{bmatrix}$
2	$\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 \\ -0.0979 & -0.0009 & 0.0003 & 0.9950 & 0.0011 & -0.0006 \\ 0.0030 & -0.3422 & -0.0731 & -0.0027 & 0.9770 & -0.0146 \\ -0.0140 & 0.0695 & -0.0089 & -0.0005 & -0.0022 & 0.9959 \end{bmatrix}$	$\begin{bmatrix} 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0028 & 0.0002 \\ 0.0004 & 0.0093 \\ 0.0002 & 0.0001 \end{bmatrix}$	$\begin{bmatrix} 0.0000 \\ 0.0000 \\ 0.0000 \\ -0.0216 \\ -0.0902 \\ 0.0165 \end{bmatrix}$
3	$\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 \\ -0.0986 & -0.0044 & 0.0010 & 0.9952 & 0.0022 & 0.0001 \\ 0.0003 & -0.2937 & -0.0477 & -0.0021 & 0.9822 & -0.0069 \\ -0.0014 & 0.0984 & -0.0057 & 0.0003 & 0.0004 & 0.9945 \end{bmatrix}$	$\begin{bmatrix} 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0028 & 0.0004 \\ 0.0006 & 0.0066 \\ -0.0003 & 0.0005 \end{bmatrix}$	$\begin{bmatrix} 0.0000 \\ 0.0000 \\ 0.0000 \\ -0.0237 \\ -0.1308 \\ 0.0449 \end{bmatrix}$

Table 3
Linear model parameters (A_i, B_i, and c_i) for lifting task in the real robot environment.

i	A_i	B_i	c_i
1	$\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 \\ -0.0886 & -0.0005 & 0.0004 & 1.0001 & 0.0014 & 0.0004 \\ 0.0239 & -0.3999 & -0.0039 & -0.0050 & 0.9873 & 0.0147 \\ -0.0038 & 0.0491 & -0.0191 & 0.0002 & -0.0111 & 0.9960 \end{bmatrix}$	$\begin{bmatrix} 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0003 & -0.0003 \\ 0.0001 & 0.0069 \\ 0.0001 & 0.0007 \end{bmatrix}$	$\begin{bmatrix} 0.0000 \\ 0.0000 \\ 0.0000 \\ -0.0382 \\ 0.1012 \\ -0.0125 \end{bmatrix}$
2	$\begin{bmatrix} 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 & 0.0000 \\ 0.0000 & 0.0000 & 1.0000 & 0.0000 & 0.0000 & 0.0020 \\ -0.0867 & -0.0029 & -0.0001 & 0.9994 & 0.0003 & -0.0010 \\ -0.0264 & -0.3879 & 0.0131 & 0.0002 & 0.9872 & 0.0098 \\ -0.0170 & 0.0421 & -0.0075 & 0.0037 & -0.0094 & 0.9979 \end{bmatrix}$	$\begin{bmatrix} 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0000 & 0.0000 \\ 0.0007 & -0.0002 \\ -0.0004 & 0.0065 \\ 0.0001 & 0.0008 \end{bmatrix}$	$\begin{bmatrix} 0.0000 \\ 0.0000 \\ 0.0000 \\ -0.0365 \\ 0.0765 \\ -0.0169 \end{bmatrix}$

References

Astrom, K. J., & Wittenmark, B. (1989). *Adaptive control*. Addison Wesley.

Atkeson, C., & Stephens, B. (2007). Multiple balance strategies from one optimization criterion. In *The IEEE-RAS 2007 international conference on humanoid robots* (pp. 57–64).

Chesi, G. (2010). LMI techniques for optimization over polynomials in control: a survey. *IEEE Transactions on Automatic Control*, *55*, 2500–2510.

Cruz, J. B., & Perkins, W. R. (1964). A new approach to the sensitivity problem in multivariable feedback system design. *IEEE Transactions on Automatic Control*, *9*, 216–223.

Dietterich, T. G. (2000). Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, *13*, 227–303.

Doya, K. (2000). Reinforcement learning in continuous time and space. *Neural Computation*, *12*, 219–245.

Doya, K., Samejima, K., Katagiri, K., & Kawato, M. (2002). Multiple model-based reinforcement learning. *Neural Computation*, *14*, 1347–1369.

Fujii, T., & Narazaki, M. (1984). A complete optimality condition in the inverse problem of optimal control. *SIAM Journal on Control and Optimization*, *22*(2), 327–341.

Ghahramani, Z., & Hinton, G. E. (2000). Variational learning for switching state-space models. *Neural Computation*, *12*(4), 831–864.

Grimes, D. B., Chalodhorn, R., & Rao, R. P. N. (2006). Dynamic imitation in a humanoid robot through nonparametric probabilistic inference. In *Proceedings of robotics: science and systems*, RSS'06. MIT Press.

Haruno, M., Wolpert, D. M., & Kawato, M. (2001). Mosaic model for sensorimotor learning and control. *Neural Computation*, *13*.

Hauskrecht, M., Meuleau, N., Kaelbling, L. P., Dean, T., & Boutilier, C. (1998). Hierarchical solution of Markov decision processes using macro-actions. In *Uncertainty in artificial intelligence* (pp. 220–229).

Hyon, S., Osu, R., & Otaka, Y. (2009). Integration of multi-level postural balancing on humanoid robots. In *IEEE international conference on robotics and automation* (pp. 1549–1556).

Imamizu, H., Kuroda, T., Yoshioka, T., & Kawato, M. (2004). Functional magnetic resonance imaging examination of two modular architectures for switching multiple internal models. *Journal of Neuroscience*, *24*(5), 1173–1181.

Imamizu, H., Sugimoto, N., Osu, R., Tsutsui, K., Sugiyama, K., Wada, Y., et al. (2007). Explicit contextual information selectively contributes to predictive switching of internal models. *Experimental Brain Research*, *181*(3), 395–408.

Kalman, R. E., & Bucy, R. S. (1961). New results in linear filtering and prediction theory. *Transactions of the ASME Series D-Journal of Basic Engineering*, *83*(1), 95–108.

Lewis, F. L. (1986). *Optimal estimation*. John Wiley & Sons, Inc.

Lim, S., & How, J. P. (1998). Control of LPV systems using a quasi-piecewise affine parameter-dependent Lyapunov function. In *American control conference*.

Morimoto, J., & Doya, K. (2001). Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning. *Robotics and Autonomous Systems*, *36*(1), 37–51.

Morse, A. S. (1996). *Lecture notes in control and information sciences*, *Control using logic-based switching*. Springer-Verlag Telos.

Nakaoka, S., Nakazawa, A., & Ikeuchi, K. (2004). An efficient method for composing while body motions of a humanoid robot. In *Proceedings of international conference on virtual systems and multimedia. VSMM2004*.

Perkins, W. R., & Cruz, J. B. (1971). Feedback properties of linear regulators. *IEEE Transactions on Automatic Control*, *16*(6), 659–664.

Rugh, W. J. (1991). Analytical framework for gain scheduling. *IEEE Control Systems Magazine*, *11*, 79–84.

Rugh, W. J., & Shamma, J. S. (2000). Research on gain scheduling. *Automatica*, *36*, 1401–1425.

Samejima, K., Doya, K., & Kawato, M. (2003). Inter-module credit assignment in modular reinforcement learning. *Neural Networks*, *16*, 985–994.

Shamma, J. S., & Athans, M. (1992). Gain scheduling: potential hazards and possible remedies. *IEEE Control Systems Magazine*, *6*, 101–107.

Slotine, J.-J. E., & Li, W. (1991). *Applied nonlinear control*. Prentice Hall.

Stephens, B. (2007). Humanoid push recovery. In *The IEEE-RAS 2007 international conference on humanoid robots* (pp. 589–595).

Sutton, R. S., Precup, D., & Singh, S. P. (1999). Between MDPs and semi-MDPs: a framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, *112*(1–2), 181–211.

Thrun, S., Burgard, W., & Fox, D. (2005). *Probabilistic robotics*. The MIT Press.

Wiering, M., & Schmidhuber, J. (1997). HQ-learning. *Adaptive Behavior*, *6*(2), 219–246.

Wolpert, D. M., Ghahramani, Z., & Jordan, M. I. (1995). An internal model for sensorimotor integration. *Science*, *269*(5232), 1880–1882.

Wolpert, D. M., & Kawato, M. (1998). Multiple paired forward and inverse models for motor control. *Neural Networks*, *11*, 1317–1329.