A probabilistic modeling of MOSAIC learning

Satoshi Osaga, Jun-ichiro Hirayama, Takashi Takenouchi and Shin Ishii
Graduate School of Information Science
Nara Institute of Science and Technology
Takayama 8916-5, Ikoma, Nara, 630-0192, Japan
{satosh-o, junich-h, ttakashi, ishii}@is.naist.jp

Abstract

Humans can generate accurate and appropriate motor commands in various and even uncertain environments. MOSAIC (MOdular Sellection And Identification for Control) was originally proposed to describe such human ability, but it includes some complex and heuristic procedures. In this article, we present an alternative and probabilistic model of MOSAIC (p-MOSAIC) as a mixture of normal distributions, and an online EM-based learning method for its predictors and controllers. A theoretical consideration shows that the learning rule of p-MOSAIC corresponds to that of MOSAIC except for some points mostly related to the learning of controllers. The results of experiments using synthetic datasets demonstrate some practical advantages of p-MOSAIC. One is that the learning rule of p-MOSAIC stabilizes the estimation of "responsibility." Another is that p-MOSAIC realizes accurate control and robust parameter learning in comparison to the original MOSAIC, especially in noisy environments, due to the direct incorporation of the noise into the model.

1 Introduction

Humans have the remarkable ability to generate accurate and appropriate motor commands in various and even uncertain environments. Studies of human motor controls have shown that dis-adaptation and re-adaptation to a learned environment are more rapid than adaptation to a novel environment [5], implying that the human motor control could be performed by a modular structure consisting of multiple controllers each adapting to a specific environment.

MOSAIC [1] was originally proposed to model the motor control system with such a modular structure. In MOSAIC, each controller is coupled with a corresponding predictor, and a motor command is determined by a weighted mean of outputs of multiple controllers, where the weight for each controller (responsibility) is estimated based on the prediction error of the corresponding predictor. However, MOSAIC includes some complex and heuristic procedures that make it difficult to understand the model.

In this study, we re-formulate MOSAIC as a probabilistic model in order to construct an easily understandable framework. Parameters of predictors and controllers are estimated by the online EM algorithm [4], which maximizes the log-likelihood of the model, given the history of control results. We also show results of computer simulations in which behaviors of responsibility and controller learning of p-MOSAIC are compared with those of MOSAIC.

2 MOSAIC

We consider a situation where the dynamics of the motor system is given by a discrete-time system:

$$\tilde{x}_{t+1} = \Phi(\tilde{x}_t, u_t),$$

where $\tilde{x}_t$ and u_t are the system state and the applied motor command, respectively, at time t. The task of the motor control is to make the system state $\tilde{x}_t$ to keep on a given trajectory x^*_t.

To perform the control task, we assume M pairs of a controller and a predictor. The aim of the controller is to generate an appropriate motor command u_t which produces the desired state x^*_{t+1}. We assume that an output of the i-th controller is represented as

$$\psi_{i,t} = \psi(\tilde{x}_t, x^*_{t+1}; v_i),$$

where v_i is the parameter of the i-th controller. The objective of the predictor is to accurately predict the system state at the next time step, and an output of the i-th predictor is given by

$$\phi_{i,t} = \phi(\tilde{x}_{t-1}, u_{t-1}; w_i),$$

where w_i is the variable parameter of the i-th predictor. Because there are M pairs of a controller and a predictor, the responsibility for each controller (and predictor) should be defined. The responsibility signal

$\lambda_{i,t}$ for the i-th pair is defined by

$$\lambda_{i,t} = \frac{\exp(-|\tilde{x}_t - \phi_{i,t}|^2/\sigma^2)\hat{\lambda}_{i,t}}{\sum_{j=1}^{M}\exp(-|\tilde{x}_t - \phi_{j,t}|^2/\sigma^2)\hat{\lambda}_{j,t}}, \quad (1)$$

where σ is a constant and $\hat{\lambda}_{i,t}$ is a rough prediction of the responsibility signal $\lambda_{i,t}$ which is typically given as a constant (then ignored). The responsibility represents how well each predictor reproduces the target dynamics, then an overall motor command $\tilde{u}_t$ at time t is given by a linear combination of outputs $\psi_{i,t}$ of the M controllers as

$$\tilde{u}_t = \sum_{i=1}^{M}\lambda_{i,t}\psi_{i,t} + u_t^{\mathrm{fb}}. \quad (2)$$

Here, u_t^{fb} is a feedback motor command, which is assumed to be produced by a PID or PAD controller, based on the difference between x_t^* and $\tilde{x}_t$.

MOSAIC is trained by updating the parameters of controllers and predictors. A learning rule is given by

$$\Delta v_i = \kappa\lambda_{i,t}\frac{\partial\psi_{i,t}}{\partial v_i}(u_t^* - \psi_{i,t}) \quad (3)$$

$$\Delta w_i = \kappa\lambda_{i,t}\frac{\partial\phi_{i,t}}{\partial w_i}(\tilde{x}_t - \phi_{i,t}), \quad (4)$$

where Δv_i and Δw_i are the updates of parameters v_i, w_i in a single learning step, κ is the learning rate, and u_t^* is the desired motor command. Although it is assumed that the desired motor command u_t^* is available in Eq. (3), this assumption is not practical. Thus, the controller learning (3) is approximately performed using the feedback-error learning [3] as

$$\Delta v_i \approx \kappa\lambda_{i,t}\frac{\partial\psi_{i,t}}{\partial v_i}u_t^{\mathrm{fb}}. \quad (5)$$

3 p-MOSAIC

With a set of M predictors, $\tilde{x}_t = \phi(\tilde{x}_{t-1}, \tilde{u}_{t-1}; w_i) + \varepsilon_i$, where ε_i is the noise of the i-th predictor, the state prediction by integrating those predictions is given probabilistically as a mixture of normal distributions:

$$p(x_t|\tilde{x}_{t-1}, \tilde{u}_{t-1}; \boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{v}) = \sum_{i=1}^{M}\lambda_i \mathrm{N}(x_t|\phi(\tilde{x}_{t-1}, \tilde{u}_{t-1}; w_i), \alpha_i^{-1}),$$

where x_t is a random variable for the predicted state at time t, $\boldsymbol{\lambda} = (\lambda_1, \cdots, \lambda_M)$ is the mixing rate vector such that $\lambda_i \geq 0$ and $\sum_{i=1}^{M}\lambda_i = 1$, $\boldsymbol{w} = (w_1, \cdots, w_M)$ is the set of predictors' parameters, and $\boldsymbol{v} = (v_1, \cdots, v_M)$ is the set of controllers' parameters. The motor command $\tilde{u}_{T-1}$ is deterministically given by Eq. (2). In our particular experiments in Section 4, we use a linear predictor:

$$\phi(x_{t-1}, u_{t-1}; w_i) = w_{i,x}x_{t-1} + w_{i,u}u_{t-1}. \quad (6)$$

For a desired trajectory $x_{1:T}^* = (x_1^*, \cdots, x_T^*)$ and an actual trajectory $\tilde{x}_{0:T} = (\tilde{x}_0, \cdots, \tilde{x}_T)$, the probability of a state sequence $x_{1:T} = (x_1, \cdots, x_T)$ of random variables is represented as

$$p(x_{1:T}|\tilde{x}_{0:T}, x_{1:T}^*; \boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{v}) = \prod_{t=1}^{T} p(x_t|\tilde{x}_{t-1}, x_t^*; \boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{v}),$$

where the random variables are assumed to be independent of each other. Given $x_{1:T}^*$ and $\tilde{x}_{1:T}$, the parameters of the predictors and the controllers are determined by the maximum likelihood estimation. In the following two subsections, we describe learning rules of the predictors and the controllers.

3.1 Learning rule of predictors

Parameters $\boldsymbol{\lambda}$ and $\boldsymbol{w}$ of the predictors are primarily estimated so as to maximize the log-likelihood:

$$\sum_{t=1}^{T}\log p(x_t = \tilde{x}_t|\tilde{x}_{t-1}, x_t^*; \boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{v}),$$

by means of the online EM algorithm, in which the controller parameters $\boldsymbol{v}$ are fixed. By introducing a hidden variable c_t that indexes predictor-controller pairs, the online free energy for any distribution of the hidden variable, $q_{\mathrm{p}}(c_t)$, is defined as

$$F_T[\{q_{\mathrm{p}}(c_t)\}, \boldsymbol{\lambda}, \boldsymbol{w}] = \sum_{t=1}^{T}\Gamma_T(t)\left\langle \log\frac{q_{\mathrm{p}}(c_t)}{p(\tilde{x}_t, c_t|\tilde{x}_{t-1}, x_t^*, \boldsymbol{\lambda}, \boldsymbol{w})}\right\rangle_{q_{\mathrm{p}}(c_t)},$$

where $p(\tilde{x}_t, c_t|\tilde{x}_{t-1}, x_t^*, \boldsymbol{\lambda}, \boldsymbol{w}) = \mathrm{N}(\tilde{x}_t|\phi_{c_t,t}, \alpha_{c_t}^{-1})\lambda_{c_t}$. $< \cdot >_{q_{\mathrm{p}}(c_t)}$ is the expectation with respect to the distribution $q_{\mathrm{p}}(c_t)$, and $\Gamma_T(t)$ is given by

$$\Gamma_T(t) = \begin{cases} 1 & (t = T) \\ \prod_{s=t+1}^{T}\gamma_s & (0 \leq t < T), \end{cases}$$

where γ_s $(0 \leq \gamma_s < 1)$ is called the forgetting factor. The online free energy is minimized by the online EM algorithm, in which the following two steps are implemented once after seeing x_T^* and $\tilde{x}_{T-1}$ at a time step T:

E-step

$$q_{\mathrm{p}}(c_T) \propto p(\tilde{x}_T, c_T|\tilde{x}_{T-1}, x_T^*, \boldsymbol{\lambda}, \boldsymbol{w}) \propto \mathrm{N}(\tilde{x}_t|\phi_{c_T,T}^{(T-1)}, 1/\alpha_{c_T})\lambda_{c_T}^{(T-1)},$$

where the superscript $T-1$ means the time step, $T-1$.

M-step

$$\lambda_i^{(T)} = (1-\eta_T)\lambda_i^{(T-1)} + \eta_T q_{\mathrm{p}}(c_T = i) \tag{7}$$

$$\Delta w_i^{(T)} = (1-\eta_T)\Delta w_i^{(T-1)} + \eta_T \kappa \alpha_i q_{\mathrm{p}}(c_T = i)(\tilde{x}_T - \phi_{i,T})\frac{\partial \phi_{i,T}}{\partial w_i}, \tag{8}$$

where η_T is given by

$$\eta_T = 1/N_T, \quad N_T = \gamma_T N_{T-1} + 1 \quad (N_0 = 0).$$

The above learning rules of p-MOSAIC involve a smoothing effect on the sufficient statistics in the M-step, because of the online free energy. On the other hand, they become similar to the learning rules of MOSAIC in a special setting of $\gamma_t = 0 (t = 1, \cdots, T)$, which corresponds to discarding the smoothing effect. Even in this special setting, however, the learning rule of p-MOSAIC contains an additional term associated with the inverse variance α_i of each predictor (Eq. (8)), which represents the noise level of the predictor.

3.2 Learning method of controllers

The controller parameters $\boldsymbol{v}$ are primarily estimated so as to maximize the log-likelihood:

$$\sum_{t=1}^{T} \log p(x_t = x_t^* | \tilde{x}_{t-1}, x_t^*; \boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{v}),$$

while the predictor parameters, $\boldsymbol{\lambda}$ and $\boldsymbol{w}$, are fixed. According to the online EM algorithm, instead of the log-likelihood, the online free energy:

$$F_T[\{q_{\mathrm{c}}(c_t)\}, \boldsymbol{v}] = \sum_{t=1}^{T} \Gamma_T(t) \left\langle \log \frac{q_{\mathrm{c}}(c_t)}{p(x_t = x_t^*, c_t | \tilde{x}_{t-1}, x_t^*, \boldsymbol{v})} \right\rangle_{q_{\mathrm{c}}(c_t)}$$

for any distribution of the hidden variable, $q_{\mathrm{c}}(c_t)$, is minimized, where $p(x_t = x_t^*, c_t | \tilde{x}_{t-1}, x_t^*, \boldsymbol{v}) = \mathrm{N}(x_t^* | \phi_{c_t,t}, \alpha_{c_t}^{-1})\lambda_{c_t}$. $< \cdot >_{q_{\mathrm{c}}(c_t)}$ is the expectation with respect to the distribution $q_{\mathrm{c}}(c_t)$. As an incremental minimization of the online free energy, the following two steps are implemented once, given the desired state x_T^* and the previous state $\tilde{x}_{T-1}$:

E-step

$$q_{\mathrm{c}}(c_T) \propto p(x_T = x_T^*, c_T | \tilde{x}_{T-1}, x_T^*, \boldsymbol{v}) \propto \mathrm{N}(x_T^* | \phi_{c_T,T}^{(T-1)}, 1/\alpha_{c_T})\lambda_{c_T}^{(T-1)}.$$

M-step

$$\Delta v_i^{(T)} = (1-\eta_T)\Delta v_i^{(T-1)} + \eta_T \kappa \lambda_i \frac{\partial \psi_{i,T-1}}{\partial v_i} \sum_{j=1}^{M} \alpha_j q(c_T = j) w_{j,u}(x_T^* - \phi_{j,T}). \tag{9}$$

Here, $w_{j,u}$ is the predictor parameter defined in Eq. (6). Even if the forgetting factor γ_t is constant at zero, the M-step equation reduces to

$$\Delta v_i^{(T)} = \kappa \lambda_i \frac{\partial \psi_{i,T-1}}{\partial v_i} \sum_{j=1}^{M} \alpha_j q(c_T = j) w_{j,u}(x_T^* - \phi_{j,T}),$$

which is obviously different from Eq. (3), the learning rule of controllers in MOSAIC. The controller learning in MOSAIC is defined as a gradient-based feedback-error learning, which tries to minimize the time-lag difference between the previous actual state $\tilde{x}_{t-1}$ and the previous desired state x_{t-1}^*. In p-MOSAIC, the controller learning tries to minimize the difference between the current predicted state $\hat{x}_t$ and the current desired state x_t^*. Moreover, the learning rule of p-MOSAIC includes the inverse variance α_j (Eq. (9)). These two points arise from the difference in the learning criteria between MOSAIC and p-MOSAIC.

4 Simulation studies

To compare p-MOSAIC with MOSAIC, we simulated the control of a spring-mass-damper system. The desired trajectory of the object (mass position) followed a mixture of sine waves for 12 seconds. To show the adaptability of the motor control system, environment (mass of the object M, damping B and spring constant K) switchs every 4 sec between the following three settings:

$$(M, B, K) = \begin{cases} 1.0,\ 2.0,\ 8.0 & (0-4\text{sec}) \\ 5.0,\ 7.0,\ 4.0 & (4-8\text{sec}) \\ 8.0,\ 3.0,\ 1.0 & (8-12\text{sec}). \end{cases}$$

In both MOSAIC and p-MOSAIC, we prepared three predictor-controller pairs. Observation and control were performed at 1,000 Hz, and a single trial was continued for 12 seconds. The predictors (6) were input by the motor command, the state (position and velocity) of the object at the present time, and output the predicted acceleration of the object at the next time. The controllers were input by the state at the present time and the desired acceleration at the next time, and output a motor command at the present time. In this simulation, we used a PAD controller to produce the feedback motor command. Note that

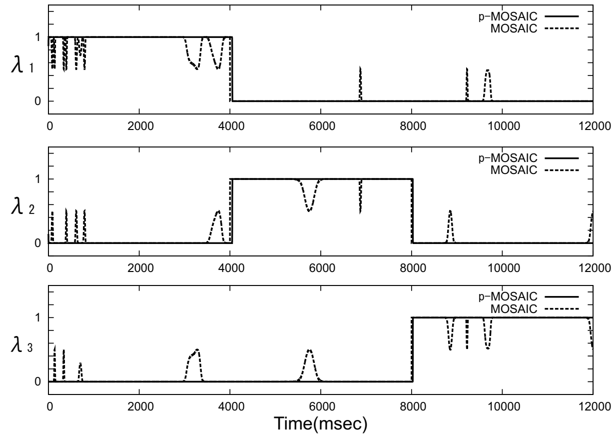

Figure 1: The responsibility along time

our task for the spring-mass-damper system is almost the same as in the previous work [2]. A regularization term is introduced to the estimation of responsibility in MOSAIC and p-MOSAIC in order to suppress any overfitting to the noisy environment.

4.1 Responsibility

We first examined how the responsibility behaves. Prior to the experiment, three predictor-controller pairs were completely adapted to their own environments. Since there is no learning factor, we can compare solely the estimation of the responsibility between Eq. (7) with the forgetting factor being zero (for comparison), and Eq. (1). Figure 1 shows the result. Although p-MOSAIC achieved a complete switching of controllers in response to changes of environments, MOSAIC sometimes failed.

4.2 Controller learning

We compared the controller learning, Eq. (9) of p-MOSAIC, and Eq. (5) of MOSAIC, assuming the predictors were completely trained to adapt to their own environments. To compare controller learning only, we used Eq. (7) in both MOSAIC and p-MOSAIC to estimate responsibility, and the forgetting factor was fixed at zero. We examined the controller learning in particular when the actual state $\tilde{x}_t$ is disturbed by a noise.

Figure 2 shows the results for a small noise and a relatively large one. When the noise level was low (upper panel), p-MOSAIC achieved more accurate control than MOSAIC. When the noise level was relatively high (lower panel), on the other hand, the learning by MOSAIC proceeded faster, but it was substantially unstable; hence, the performance improved due to p-MOSAIC after about 1,000 trials. In the early learning phase, the controller learning of p-MOSAIC proceeded

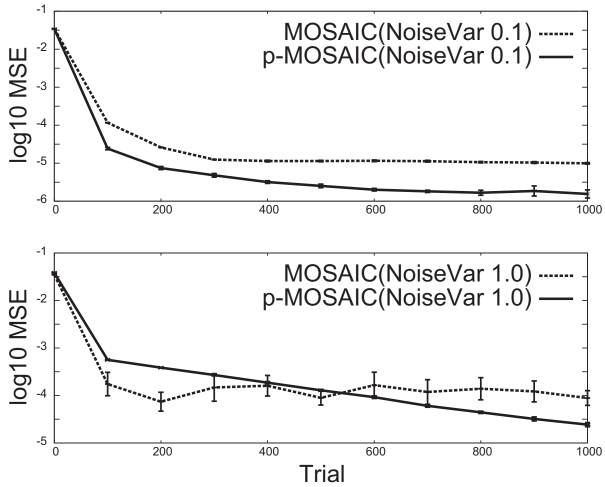

Figure 2: Logarithm of the mean square error between the actual and the desired trajectory vs. number of trials

slowly due to the control of the inverse variance α_i. Because the environmental noise was large, the adaptive control of the inverse variance made the learning slow but stable, suggesting adaptive adjustment of learning speed in p-MOSAIC.

5 Summary

In this study, we proposed p-MOSAIC, a probabilistic model of MOSAIC, and derived learning rules according to the online EM algorithm. P-MOSAIC achieved an appropriate estimation of responsibility in the predictor, and accurate control and robust learning when the controllers learned.

References

[1] D.M. Wolpert and M. Kawato. (1998). *Neural Networks*, **11**, 1317-1329.

[2] M. Haruno, D.M. Wolpert, M. Kawato. (2001). *Neural Computation*, **13**, 2201-2220.

[3] M. Kawato. (1990). Feedback-error-learning neural network for supervised learning. In R. Eckmiller (Ed.), *Advanced neural computers* (pp. 365-372). Amsterdam: North-Holland.

[4] M. Sato and S. Ishii. (2000). *Neural Computation*, **12**(2), 407-432.

[5] R.B. Welch, B. Bridgeman, S. Anand and K.E. Browman. (1993). *Perception and Psychophysics*, **54**(2), 195-204.