Proceedings of the International MultiConference of Engineers and Computer Scientists 2015 Vol I,
IMECS 2015, March 18 - 20, 2015, Hong Kong

Remarks on Solving Algebraic Riccati Matrix Equations using a Hopfield Neural Network and Application to Optimal Control Problems

Kazuhiko Takahashi, Sakie Sakae, and Masafumi Hashimoto

Abstract—**This paper discusses a method of solving algebraic Riccati matrix equations using a Hopfield neural network and presents its application to the optimal control of dynamic systems with a quadratic cost function. To solve an algebraic Riccati matrix equation using the optimization ability of a Hopfield neural network, an energy function is defined using the elements of the algebraic Riccati matrix equation and a penalty function of incorporating the positive definite constraint of the solution. The energy function is minimized through the dynamics of the Hopfield neural network, and the converged neuron states provide the solution of the algebraic Riccati matrix equation. Computational experiments using a linear second-order system confirm that the Hopfield neural network can solve the algebraic Riccati matrix equation with sufficient accuracy. The optimal control of an automotive vehicle is demonstrated as a practical application of controlling dynamic systems using the solution obtained by the Hopfield neural network; the simulation results indicate the feasibility and effectiveness of the proposed neural network–based optimal control.**

Index Terms—**Hopfield neural network, Algebraic Riccati matrix equation, Energy function, Optimal control, Dynamic systems.**

I. INTRODUCTION

DURING the past quarter century, artificial neural networks have been applied worldwide in many scientific fields because of their flexibility and learning ability. In control engineering, several types of neural network–based control systems that take advantage of the neural network characteristics, such as non–linear function approximation ability, adaptive/learning ability, generalizability and optimization ability, have been proposed, and many successful control applications have been demonstrated [1], [2], [3], [4]. It is well known that Riccati matrix equations should be solved to obtain state feedback gain parameters in linear–quadratic optimal control problems and robust control problems with H_2 and H_∞ control. Many numerical methods of solving algebraic Riccati matrix equations have been investigated because the solutions of these equations in continuous/discrete–time play an important role in control problems. Accordingly, several studies have investigated the use of neural networks to solve algebraic Riccati matrix equations [5], [6], [7], [8].

Manuscript received December 16, 2014; revised January 12, 2015. This research was partially funded by MEXT-Supported Program for the Strategic Research Foundation at Private Universities, 2014-2018.

K. Takahashi is with Information Systems Design, Doshisha University, Kyoto, Japan e-mail: katakaha@mail.doshisha.ac.jp.

S. Sakae is with Faculty of Science and Engineering, Doshisha University, Kyoto, Japan e-mail: bul1070@mail4.doshisha.ac.jp.

M. Hashimoto is with Intelligent Information Engineering and Science, Doshisha University, Kyoto, Japan e-mail: mhashimo@mail.doshisha.ac.jp.

In this paper, a neural network–based optimal control whose feedback gain parameters are calculated using the solution of a Riccati matrix equation with parallel computing by a Hopfield neural network is investigated. In the designed optimal controller, the Hopfield neural network provides the solution of the Riccati matrix equation through an energy minimization process in its off-line training. Computational experiments for controlling dynamic systems for a quadratic cost function are conducted to evaluate the feasibility of the proposed neural network–based optimal control.

II. SOLUTION OF ALGEBRAIC RICCATI MATRIX EQUATION USING HOPFIELD NEURAL NETWORK

Let the following continuous-time linear system be considered as a target plant:

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t) \\ \boldsymbol{x}(0) = \boldsymbol{x}_0 \end{cases}, \tag{1}$$

where $\boldsymbol{x}(t) \in \mathbb{R}^n$ is the state vector, $\boldsymbol{u}(t) \in \mathbb{R}^m$ is the input vector, $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{B} \in \mathbb{R}^{n\times m}$ are coefficient matrices and $\boldsymbol{x}_0$ is an initial state vector. When the system is controllable and all state variables are measurable, an optimal state feedback input can be obtained so as to minimize the quadratic cost function I as follows:

$$\boldsymbol{u}(t) = -\frac{1}{2}\boldsymbol{H}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{G}\boldsymbol{x}(t), \tag{2}$$

$$I = \int_0^\infty \left\{\boldsymbol{x}^{\mathrm{T}}(t)\boldsymbol{Q}\boldsymbol{x}(t) + \boldsymbol{u}^{\mathrm{T}}(t)\boldsymbol{H}\boldsymbol{u}(t)\right\} dt \tag{3}$$

where $\boldsymbol{Q} \in \mathbb{R}^{n\times n}$ and $\boldsymbol{H} \in \mathbb{R}^{m\times m}$ are symmetric and positive definite matrices, and $\boldsymbol{G} \in \mathbb{R}^{n\times n}$ is the solution of the following algebraic Riccati matrix equation:

$$\boldsymbol{A}^{\mathrm{T}}\boldsymbol{G} + \boldsymbol{G}\boldsymbol{A} - \frac{1}{2}\boldsymbol{G}\boldsymbol{B}\boldsymbol{H}^{-1}\boldsymbol{B}^{\mathrm{T}}\boldsymbol{G} + 2\boldsymbol{Q} = \boldsymbol{0}. \tag{4}$$

Describing the i–th row and j–th column element of the left hand side of Eq. (4), ψ_{ij}, the algebraic Riccati matrix equation can be expressed using its element as follows:

$$\psi_{ij}(\boldsymbol{\xi}) = 0 \qquad (i, j = 1, 2, \cdots, n) \tag{5}$$

where $\boldsymbol{\xi} = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{22} & g_{23} & \cdots & g_{nn} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{\frac{n(n+1)}{2}}$ is defined by the upper triangle elements of matrix $\boldsymbol{G}$.

The Hopfield neural network composes a pseudo–gradient system that optimizes an energy function as follows:

$$\dot{\boldsymbol{z}} = -\boldsymbol{\mu}\nabla_{\boldsymbol{z}} E(\boldsymbol{z}) \tag{6}$$

ISBN: 978-988-19253-2-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

where $\boldsymbol{z}$ is the state vector of the neuron, $\boldsymbol{\mu} = \mathrm{diag}(\tau_i^{-1})$ is the learning factor matrix, E is the energy function given by

$$E(\boldsymbol{z}) = -\frac{1}{2}\sum_i\sum_j w_{ij}z_iz_j - \sum_i z_i\theta_i + \sum_i \frac{1}{\tau_i}\int_0^{z_i} F_i^{-1}(z)dz,$$

w_{ij} is the weight, θ_i is the threshold, τ_i is the constant and F_i is the activation function of the neuron. To solve the algebraic Riccati matrix equation using the Hopfield neural network, we define the energy function as follows [9]:

$$E(\boldsymbol{\xi}, \alpha) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=i}^{n}\psi_{ij}^2(\boldsymbol{\xi}) + \frac{\alpha}{2}\sum_{i=1}^{n}P_i[\phi_i(\boldsymbol{\xi})] \qquad (7)$$

where ϕ_i is the i–th order leading principal minor of matrix $\boldsymbol{G}$, P_i is the penalty function and $\alpha \gg 0$ is the penalty parameter. Because of the positive definite matrix $\boldsymbol{G}$, the following inequality constraints should be considered:

$$\phi_i(\boldsymbol{\xi}) > 0 \qquad (i = 1, 2, \cdots, n). \qquad (8)$$

The penalty function is used to transform an optimization problem with inequality constraint conditions into an optimization problem without constraint conditions. Here the second–order function is used as the penalty function as follows:

$$P_i(s) = \begin{cases} 0 & (s \geq 0) \\ [\min(0, s)]^2 & (s < 0) \end{cases} . \qquad (9)$$

As a result, the dynamics of the Hopfield neural network is defined as follows:

$$\dot{\boldsymbol{\xi}} = -\boldsymbol{\mu}\nabla_{\boldsymbol{\xi}}E(\boldsymbol{\xi}, \alpha). \qquad (10)$$

III. Computational Experiments

Computational experiments were conducted to evaluate the feasibility of using the Hopfield neural network to solve the algebraic Riccati matrix equation. In the experiments, we used Mathematica™ ver. 7 (Wolfram Research, Inc.) to solve the simultaneous differential equations that represent the dynamics of the Hopfield neural network.

First, the plant was assumed to be a linear second-order system in which

$$\boldsymbol{A} = \begin{bmatrix} 0 & 1 \\ -2 & -3 \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \boldsymbol{Q} = \begin{bmatrix} 5 & 0 \\ 0 & 5 \end{bmatrix}, H = 1.$$

The energy function of the Hopfield neural network was defined as follows:

$$\begin{aligned} E(\boldsymbol{\xi}, \alpha) = \frac{1}{2}\Big\{&\left(10 - 4\xi_2 - 0.5\xi_2^2\right)^2 \\ &+ (\xi_1 - 3\xi_2 - 2\xi_3 - 0.5\xi_2\xi_3)^2 \\ &+ \left(10 + 2\xi_2 - 6\xi_3 - 0.5\xi_3^2\right)^2\Big\} \\ &+ \frac{\alpha}{2}\left\{P_1(\xi_1) + P_2(\xi_1\xi_3 - \xi_2^2)\right\}, \end{aligned}$$

where $\xi_1 = g_{11}$, $\xi_2 = g_{12}$ and $\xi_3 = g_{22}$. Figure 1 shows the minimization process for the energy function using the Hopfield neural network. Here the initial condition was $\boldsymbol{\xi}(0) = \begin{bmatrix} 0.1 & 0.01 & 0.1 \end{bmatrix}^{\mathrm{T}}$, the learning factor was $\mu_i = 10^4$ $(i = 1, 2, 3)$ and the penalty factor was $\alpha = 10^7$. After minimizing the energy function, the solution of the algebraic Riccati matrix equation is defined by the neuron state of the Hopfield neural network as follows:

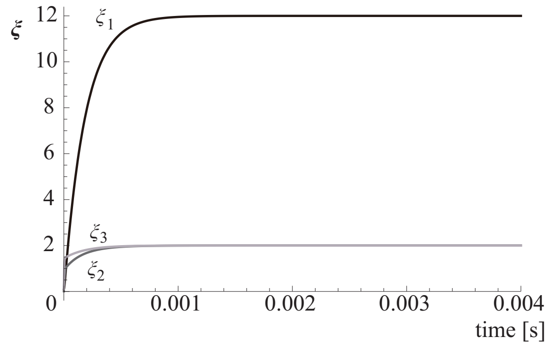

Fig. 1. Solving process of the algebraic Riccati matrix equation using the Hopfield neural network.

$$\boldsymbol{G} = \begin{bmatrix} 12 & 2 \\ 2 & 2 \end{bmatrix},$$

where the energy function $E = 1.38813\times10^{-17}$ at $t = 0.004$ s. Furthermore, we can obtain the analytical solutions of the algebraic Riccati matrix equation as follows:

$$\boldsymbol{G} = \begin{bmatrix} 0 & -10 \\ -10 & -10 \end{bmatrix}, \begin{bmatrix} -24 & -10 \\ -10 & -2 \end{bmatrix}, \begin{bmatrix} -36 & 2 \\ 2 & -14 \end{bmatrix}, \begin{bmatrix} 12 & 2 \\ 2 & 2 \end{bmatrix}.$$

Thus, the positive definite matrix is

$$\boldsymbol{G} = \begin{bmatrix} 12 & 2 \\ 2 & 2 \end{bmatrix}.$$

This result indicates that the Hopfield neural network can solve the algebraic Riccati matrix equation with sufficient accuracy.

Next, the optimal control of an automotive vehicle system was considered as an example of a practical application. Figure 2 shows the model of the automotive vehicle that drives with a constant velocity v [10]. Here, $\eta(t)$ is the angle between the velocity vector of the vehicle and the X-axis, $\zeta(t)$ is the yaw angle of the vehicle, $\sigma(t)$ is the steering angle of the front wheel, L is the wheel base and $[x(t), y(t)]$ is the position of the vehicle's centre of gravity. If the vehicle drives on a straight road and moves without sudden operation

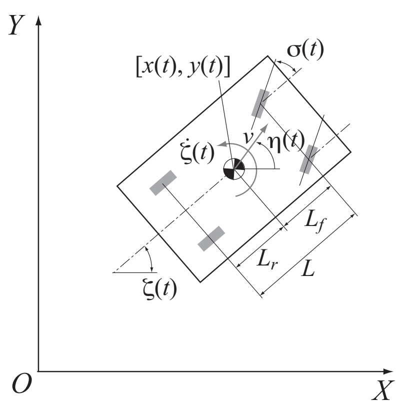

Fig. 2. Model of an automotive vehicle driving with a constant velocity v.

ISBN: 978-988-19253-2-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

of the steering angle, the equations of motion of the vehicle can be derived with the conditions of $|\eta(t)| \ll 1$, $|\zeta(t)| \ll 1$ and $|\sigma(t)| \ll 1$ as follows [11]:

$$\begin{cases} M\ddot{y}(t) + \dfrac{2(c_f + c_r)}{v}\dot{y}(t) + \dfrac{2(L_f c_f - L_r c_r)}{v}\dot{\zeta}(t) \\ \qquad\qquad -2(c_f + c_r)\zeta(t) = 2c_f\sigma(t) \\ \dfrac{2(L_f c_f - L_r c_r)}{v}\dot{y}(t) + J\ddot{\zeta}(t) + \dfrac{2(L_f^2 c_f + L_r^2 c_r)}{v}\dot{\zeta}(t) \\ \qquad\qquad -2(L_f c_f - L_r c_r)\zeta(t) = 2L_f c_f\sigma(t) \end{cases},$$

where M is the mass of the vehicle, J is the moment of inertia of the yaw angle, c_f is the cornering power of the front wheel and c_r is the cornering power of the rear wheel. To describe the state space form of the above equations of motion, the state vector is defined by $\boldsymbol{x}(t) = \begin{bmatrix} y(t) & \zeta(t) & \dot{y}(t) & \dot{\zeta}(t) \end{bmatrix}^{\mathrm{T}}$, and the steering angle $\sigma(t)$ is selected as the control input $u(t)$. The coefficient matrices are

$$\boldsymbol{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & a_{32} & a_{33} & a_{34} \\ 0 & a_{42} & a_{43} & a_{44} \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0 \\ 0 \\ b_3 \\ b_4 \end{bmatrix},$$

where

$$\begin{array}{ll} a_{32} = \frac{2(c_f+c_r)}{M}, & a_{33} = \frac{-2(c_f+c_r)}{Mv}, \\ a_{34} = \frac{-2(L_f c_f - L_r c_r)}{Mv}, & a_{42} = \frac{2(L_f c_f + L_r c_r)}{J}, \\ a_{43} = \frac{-2(L_f c_f - L_r c_r)}{Jv}, & a_{44} = \frac{-2(L_f^2 c_f + L_r^2 c_r)}{Jv}, \\ b_3 = \frac{2c_f}{M} \quad \text{and} & b_4 = \frac{2L_f c_f}{J}. \end{array}$$

In the computational experiment, the parameters were $L_f = 1.733$, $L_r = 0.867$, $c_f = c_r = 17658$, $M = 1373.4$, $v = 11.1$ and $J = 1962$. The energy function of the Hopfield neural network was defined as follows:

$$\begin{aligned} E(\boldsymbol{\xi}, \alpha) = \frac{1}{2}\Big[&\left\{-0.5h^{-1}(b_3\xi_3 + b_4\xi_4)^2 + 2q_{11}\right\}^2 \\ &+ \left\{a_{32}\xi_3 + a_{42}\xi_4 - 0.5h^{-1}(b_3\xi_6 + b_4\xi_7) \times (b_3\xi_3 + b_4\xi_4)\right\}^2 \\ &+ \left\{a_{33}\xi_3 + a_{43}\xi_4 - 0.5h^{-1}(b_3\xi_8 + b_4\xi_9) \times (b_3\xi_3 + b_4\xi_4) + \xi_1\right\}^2 \\ &+ \left\{a_{34}\xi_3 + a_{44}\xi_4 - 0.5h^{-1}(b_3\xi_9 + b_4\xi_{10}) \times (b_3\xi_3 + b_4\xi_4) + \xi_2\right\}^2 \\ &+ \left\{2a_{32}\xi_6 + 2a_{42}\xi_7 - 0.5h^{-1}(b_3\xi_6 + b_4\xi_7)^2 + 2q_{22}\right\}^2 \\ &+ \{a_{32}\xi_8 + a_{33}\xi_6 + a_{42}\xi_9 + a_{43}\xi_7 + \xi_2 - 0.5h^{-1}(b_3\xi_8 + b_4\xi_9)(b_3\xi_6 + b_4\xi_7)\}^2 \\ &+ \{a_{32}\xi_9 + a_{34}\xi_6 + a_{42}\xi_{10} + a_{44}\xi_7 + \xi_5 - 0.5h^{-1}(b_3\xi_9 + b_4\xi_{10})(b_3\xi_6 + b_4\xi_7)\}^2 \\ &+ \{2a_{33}\xi_8 + 2a_{43}\xi_9 + 2\xi_3 - 0.5h^{-1}(b_3\xi_8 + b_4\xi_9)^2 + 2q_{33}\}^2 \\ &+ \{a_{33}\xi_9 + a_{34}\xi_8 + a_{43}\xi_{10} + a_{44}\xi_9 + \xi_4 + \xi_6 - 0.5h^{-1}(b_3\xi_9 + b_4\xi_{10})(b_3\xi_8 + b_4\xi_9)\}^2 \\ &+ \{2a_{34}\xi_9 + 2a_{44}\xi_{10} + 2\xi_7 - 0.5h^{-1}(b_3\xi_9 + b_4\xi_{10})^2 + 2q_{44}\}^2\Big] \\ + \frac{\alpha}{2}\Big\{&P_1(\xi_1) + P_2(\xi_1\xi_5 - \xi_2^2) \\ &+ P_3(\xi_1\xi_5\xi_8 - \xi_1\xi_6^2 - \xi_2^2\xi_8 + 2\xi_2\xi_3\xi_6 - \xi_3^2\xi_5) \\ &+ P_4(\xi_1\xi_5\xi_8\xi_{10} + 2\xi_1\xi_6\xi_7\xi_9 - \xi_1\xi_5\xi_9^2 - \xi_1\xi_8\xi_7^2 \\ &\quad -\xi_1\xi_{10}\xi_6^2 + 2\xi_2\xi_3\xi_6\xi_{10} - 2\xi_2\xi_3\xi_7\xi_9 - 2\xi_2\xi_4\xi_6\xi_9 \\ &\quad +2\xi_2\xi_4\xi_7\xi_8 - \xi_2^2\xi_8\xi_{10} + \xi_2^2\xi_9^2 + 2\xi_3\xi_4\xi_5\xi_9 \\ &\quad -2\xi_3\xi_4\xi_6\xi_7 - \xi_3^2\xi_5\xi_{10} + \xi_3^2\xi_7^2 - \xi_4^2\xi_5\xi_8 + \xi_4^2\xi_6^2)\Big\}, \end{aligned}$$

where $\xi_1 = g_{11}$, $\xi_2 = g_{12}$, $\xi_3 = g_{13}$, $\xi_4 = g_{14}$, $\xi_5 = g_{22}$, $\xi_6 = g_{23}$, $\xi_7 = g_{24}$, $\xi_8 = g_{33}$, $\xi_9 = g_{34}$ and $\xi_{10} = g_{44}$. Assuming that the weights in the cost function are $\boldsymbol{Q} = \mathrm{diag}(1)$ and $H = 0.1$, the minimization process of the energy function using the Hopfield neural network is shown in Fig. 3 for the following initial conditions: $\boldsymbol{\xi}(0) = \begin{bmatrix} 1 & 0.1 & 0.1 & 0.1 & 1 & 0.1 & 0.1 & 1 & 0.1 & 1 \end{bmatrix}^{\mathrm{T}}$, learning factor $\mu_i = 10^4$ $(i = 1, 2, \cdots, 10)$ and penalty factor $\alpha = 10^7$. The solution of the algebraic Riccati matrix equation is as follows:

$$\boldsymbol{G} = \begin{bmatrix} 2.459528 & 0.577600 & 0.270507 & -0.202674 \\ 0.577600 & 3.977375 & 0.139925 & -0.070456 \\ 0.270507 & 0.139924 & 0.199556 & -0.149347 \\ -0.202674 & -0.070456 & -0.149347 & 0.135941 \end{bmatrix},$$

where the energy function $E = 1.63132 \times 10^{-20}$ at $t = 0.004$ s. As a reference, the algebraic Riccati matrix equation was solved in Scilab [12]. Using the command 'riccati()' yields the following solution:

$$\boldsymbol{G} = \begin{bmatrix} 2.459528 & 0.577600 & 0.270507 & -0.202674 \\ 0.577600 & 3.977375 & 0.139925 & -0.070456 \\ 0.270507 & 0.139924 & 0.199556 & -0.149347 \\ -0.202674 & -0.070456 & -0.149347 & 0.135941 \end{bmatrix}.$$

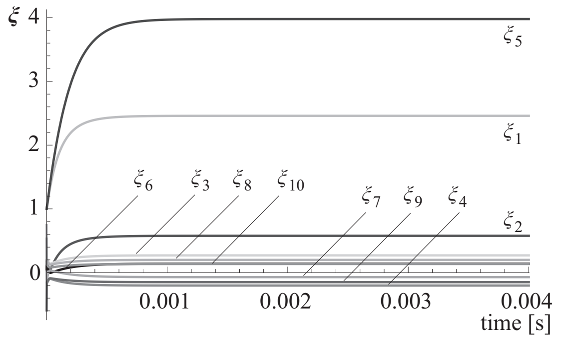

Fig. 3. Solving process of the algebraic Riccati matrix equation using the Hopfield neural network where the weight parameters of the cost function are $\boldsymbol{Q} = \mathrm{diag}(1)$ and $H = 0.1$.

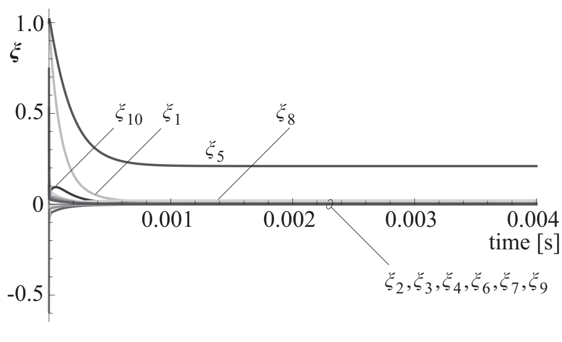

Fig. 4. Solving process of the algebraic Riccati matrix equation using the Hopfield neural network where the weight parameters of the cost function are $\boldsymbol{Q} = \mathrm{diag}(10^{-3})$ and $H = 0.1$.

Figure 4 shows the minimization process of the energy function using the Hopfield neural network when the weight parameters in the cost function are $\boldsymbol{Q} = \mathrm{diag}(10^{-3})$ and $H = 0.1$. The solution of the algebraic Riccati matrix equation is as follows:

$$\boldsymbol{G} = \begin{bmatrix} 0.008254 & 0.003098 & 0.002160 & -0.001139 \\ 0.003098 & 0.209896 & -0.002588 & 0.020648 \\ 0.002160 & -0.002588 & 0.000927 & -0.000810 \\ -0.001139 & 0.020648 & -0.000810 & 0.002493 \end{bmatrix},$$

where the energy function $E = 6.32672 \times 10^{-14}$ at $t = 0.004$ s. The solution obtained by Scilab is as follows:

$$\boldsymbol{G} = \begin{bmatrix} 0.008254 & 0.003098 & 0.002160 & -0.001139 \\ 0.003098 & 0.209896 & -0.002588 & 0.020648 \\ 0.002160 & -0.002588 & 0.000927 & -0.000810 \\ -0.001139 & 0.020648 & -0.000810 & 0.002493 \end{bmatrix}.$$

The solution obtained by the Hopfield neural network corresponds to that calculated using the Scilab function. These results confirm the effectiveness of the proposed method to obtain the solution of the algebraic Riccati matrix equation.

Figure 5 shows an example of system response controlled by the optimal controller in which the feedback gain parameters are calculated with the solution of the algebraic Riccati matrix equation shown in Fig. 3. Here the optimal feedback input was $u(t) = \begin{bmatrix} -3.162278 & -6.999119 & -2.359092 & -2.005059 \end{bmatrix} \boldsymbol{x}(t)$ and the initial condition was $\boldsymbol{x}(0) = \begin{bmatrix} 1 & -\frac{\pi}{9} & 0 & 0 \end{bmatrix}^{\mathrm{T}}$. As shown in Fig. 5, each state variable can be regulated to the equilibrium point.

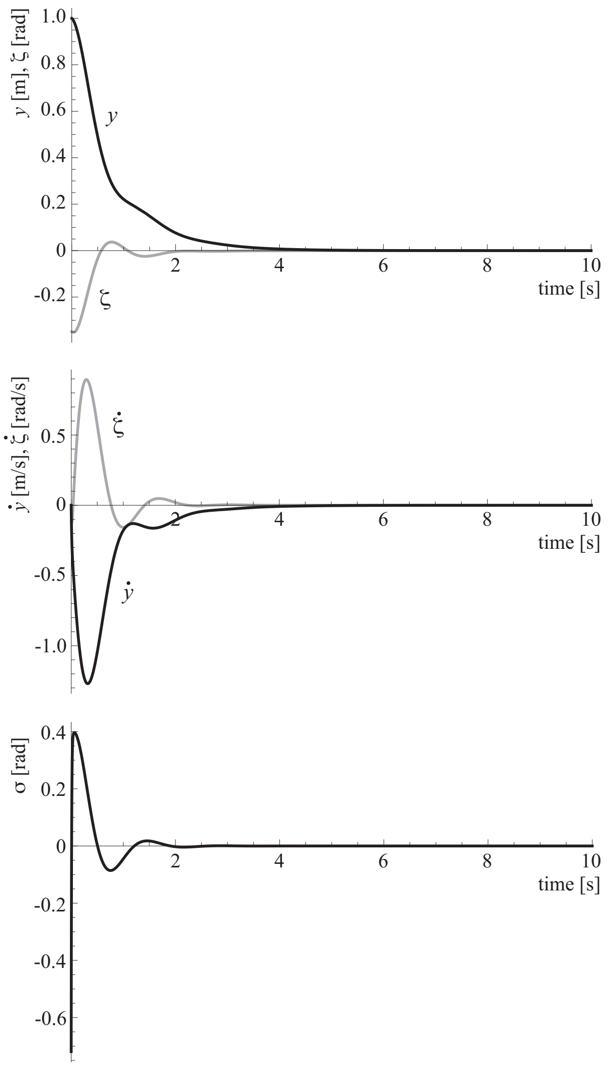

Fig. 5. Example of optimal control for a moving automotive vehicle where $\boldsymbol{Q} = \mathrm{diag}(1)$ and $H = 0.1$ in the cost function.

IV. Conclusions

In this study, we investigated the capability of a Hopfield neural network for solving algebraic Riccati matrix equations and explored its application to the optimal control of dynamic systems. An energy function composed of the elements of the algebraic Riccati matrix equation was considered, and a penalty function was introduced into the energy function because of the positive definite constraint of the solution. The dynamics of the Hopfield neural network optimized the energy function in an off–line process, and the converged neuron states yielded the solution of the algebraic Riccati matrix equation. Computational experiments were conducted to evaluate the effectiveness of the proposed method. Comparing the solution obtained using the Hopfield neural network with either analytical or numerical solutions, it was confirmed that the Hopfield neural network could solve the algebraic Riccati matrix equation with sufficient accuracy. As a practical application example of controlling dynamic systems, the optimal control problem of an automotive vehicle was considered; the simulation results confirmed the feasibility and effectiveness of the proposed neural network–based optimal control.

References

[1] M. T. Hagan, H. B. Demuth and O. D. Jesus, “An Introduction to the Use of Neural Networks in Control Systems”, *International Journal of Robust and Nonlinear Control*, Vol. 12, No. 11, pp. 959–985, 2002.

[2] M. R. G. Meireles, P. E. M. Almeida and M. G. Simoes, “A Comprehensive Review for Industrial Applicability of Artificial Neural Networks”, *IEEE Transactions on Industrial Electronics*, Vol. 50, No. 3, pp. 585–601, 2003.

[3] J. Balderud and L. Giovanini, “Adaptive Control and Signal Processing Literature Survey”, *International Journal of Adaptive Control and Signal Processing*, Vol. 22, No. 3, pp. 318–321, 2008.

[4] D. N. Tuong and J. Peters, “Model Learning for Robot Control: A Survey”, *Cognitive Processing*, Vol. 12, pp. 319–340, 2011.

[5] S. W. Ellacott, “Aspects of the Numerical Analysis of Neural Networks”, *Acta Numerica*, Vol. 3, pp. 145–202, 1994.

[6] F. M. Ham and E. G. Collins, “A Neurocomputing Approach for Solving the Algebraic Matrix Riccati Equation”, in Proceedings of IEEE International Conference on Neural Networks, Vol. 1, pp. 617–622, 1996.

[7] J. Wang and W. Guang, “A Multilayer Recurrent Neural Network for Solving Continuous–Time Algebraic Riccati Equations”, *Neural Networks*, Vol. 11, No. 5, pp. 939–950, 1998.

[8] M. Karam, M. A. Zohdy and S. S. Farinwata, “Robust Optimal Control Using Recurrent Dynamic Neural Network”, in Proceedings of IEEE International Symposium on Intelligent Control, pp. 331–336, 2001.

[9] D. Gong, M. Gen, G. Yamazaki and W. Xu, “Neural Network Approach for General Assignment Problem”, in Proceedings of IEEE International Conference on Neural Networks, pp. 1861–1866, 1995.

[10] H. Hashimoto and C. Ishii, Basics of Simulation using Scilab/Scicos, Ohmusha, 2008 (in Japanese).

[11] M. Abe, Automotive Vehicle Dynamics - Theory and Applications - Second Edition, Tokyo Denki University Press, 2012 (in Japanese).

[12] Scilab Enterprises, Scilab: Free and Open Source software for numerical computation, Scilab Enterprises, Orsay, France, http://www.scilab.org, 2012.

ISBN: 978-988-19253-2-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)