NEURAL NETWORK BASED ADAPTIVE CONTROL WITH HIERARCHY-STRUCTURED DYNAMIC INVERSION

○ Tetsujiro Ninomiya[1] and Yoshikazu Miyazawa[2]

[1]Japan Aerospace Exploration Agency, Tokyo 182-0015, Japan
[2]Department of Aeronautics and Astronautics, Kyushu University, Fukuoka 819-0395, Japan

ABSTRACT

Japan Aerospace Exploration Agency plans a new flight demonstration project for a super sonic transport. A typical supersonic transport has a larger flight envelope than conventional aircraft, so that the flight controller is required to have flexible control capability. Therefore, adaptive control is promising to be applied to the flight control system. In this paper, we propose an adaptive control method which is comprised of neural networks and Hierarchical-Structured Dynamic Inversion. Its structure is clearly comprehensible to the users and numerical simulation shows its effectiveness.

1 INTRODUCTION

Japan Aerospace Exploration Agency (JAXA) studies key technologies to develop a super sonic transport, and now JAXA plans a new flight demonstration project, the Silent Supersonic Technology Demonstration "S3TD" project. JAXA has had a steady progress in flight control of UAV in preceding flight experiments [1], [2]. To improve our art of flight contrl, advanced contrl technologies are investigated to apply to this project.

A typical supersonic transport has a larger flight envelope than conventional aircraft, so that the flight controller is required to be more flexible to the flight conditions. Therefore, the adaptive control method is promissing to be applied to the flight control systems.

Neural networks (NNs) are widely researched in the field of flight control [3] [4], for a NN has an adaptive capability. Many of these studies are using NNs with dynamic inversion(DI) controller to compensate the modling errors, and the output of NNs are added to the outputs of DI controllers. Consequently, the total input to the plant heavily depends on the behavior of NNs, and it is not easy to predict what they realy do. This is why there are only few applications of NN based adaptive control to the actual systems, even though there are so many research in this field.

To solve this drawback, we propose a novel control scheme, which consists of a dynamic inversion (DI) controller and NNs based adaptive estimator. This framework realizes that the controller is complihensible and it has adaptive ability.

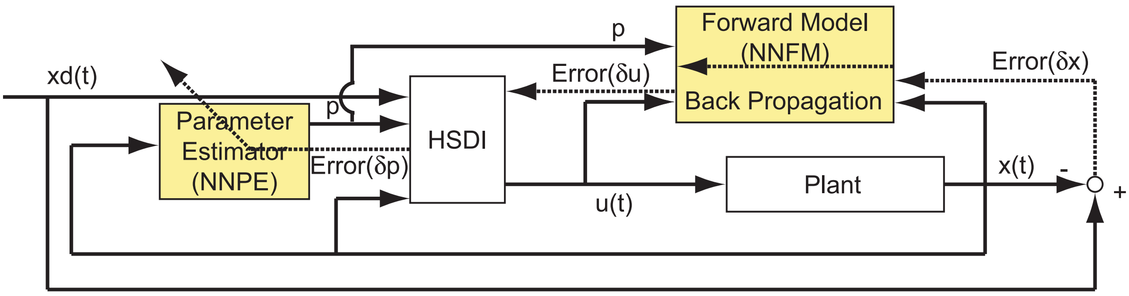

Figure 1: Block diagram of proposed control scheme EFIM. Yellow blocks show NNs.

2 PROPOSED CONTROLLER SCHEME

To solve the forementioned drawback, we propose a control scheme consisting of a hierarchy-structured dynamic inversion (HSDI) controller[5] and a neural network parameter estimator (NNPE). Figure 1 depicts the block diagram of this control scheme. This method is an extention of the forward and inverse modeling (FIM) method [6].

2.1 HIERARCHY-STRUCTURED DYNAMIC INVERSION

HSDI exploits characteristics in flight dynamics. Figure 2 shows the concept of HSDI. A rigid-body motion of aircraft has different timescale motion of angular acceleration, angular rate, angle and acceleration, velocity, and position. Therefore, the entire controller structure can be devided to form a hierarchy structure and DI can be appleid to each layer.

DI can cancel and modify the characteristic stability of the plant by using state feedback. Concerning the typical flight dynamics, the relationship between control surfaces and state variable follows the subsequent procedure. Control surfaces generate momentum which excites angular acceleration, and angular acceleration changes angular ratio and attitude angle. Then attitude angle affect the aerodynamic force which changes the acceleration, velocity, and position. DI reverses this causal link to calculate control commands from the position commands.

The concrete procedure of HSDI is as follows. In ith layer, let a controlled variable be $\boldsymbol{x}_i$, a control variable be $\boldsymbol{x}_{i+1}$, other variables and control commands be $\tilde{\boldsymbol{x}}_i, \delta$ respectively. Then the equation of motion is

$$\dot{\boldsymbol{x}}_i = \boldsymbol{f}_i(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \delta). \tag{1}$$

$\boldsymbol{x}_{i+1}$ can be obtained by solving

$$\boldsymbol{f}_i(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_i, \boldsymbol{x}_{i+1_c}, \delta) = -K_i(\boldsymbol{x}_i - \boldsymbol{x}_{i_c}). \tag{2}$$

The left-hand side is linearized and the $\boldsymbol{x}_{i+1}$ can be written as

$$\boldsymbol{x}_{i+1_c} = \boldsymbol{x}_{i+1} + \Delta \boldsymbol{x}_{i+1} \tag{3}$$

$$\Delta \boldsymbol{x}_{i+1} = -\left(\frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{x}_{i+1}}\right)^{-1} \{\boldsymbol{f}_i(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \delta) + K_i(\boldsymbol{x}_i - \boldsymbol{x}_{i_c})\}\,. \tag{4}$$

To reduce workload of design process, gains are simplified to be a two tuning parameters; one is gain of the most inner loop, and the other is constant gain ratio of adjacent layers.

This method can cope with nonlinear dynamics and therefor it is able to deal with wide flight envelope without switching gain values. Another merit of HSDI is that it suppresses increasing resources necessary for reliable flight control law development.

2.2 FORWARD AND INVERSE MODELING

Fig. 3 shows the block diagram of FIM, and it depicts that two NNs are utilized in FIM. One is a controller, and the other is a forward model of the plant. The neural network forward model (NNFM) is required to be trained before the NN controller learns. The NN controller generates a control input u_{NN} to the plant, and it learns to generate suitable commands such that the plant follows the reference trajectory x_c. The NNFM provides the differential of plant dynamics which is necessary to revise the weights of NN controller.

FIM can be applied to a nonlinear system, for NN has the ability to approximate a nonlinear function. Moreover, FIM has outstanding merit comparing to the direct inverse modeling (DIM), which uses only one NN as an inverse model of a target plant to generate u_{NN} from x_c. FIM is goal directed but DIM is not, and FIM can solve redundant problem but DIM can not[7]. In this way, FIM is a good method for NN controller, however it directly uses NN output signals as a plant input, and moreover it is impossible to control the plant properly until the NN controller obtains the appropriate weights.

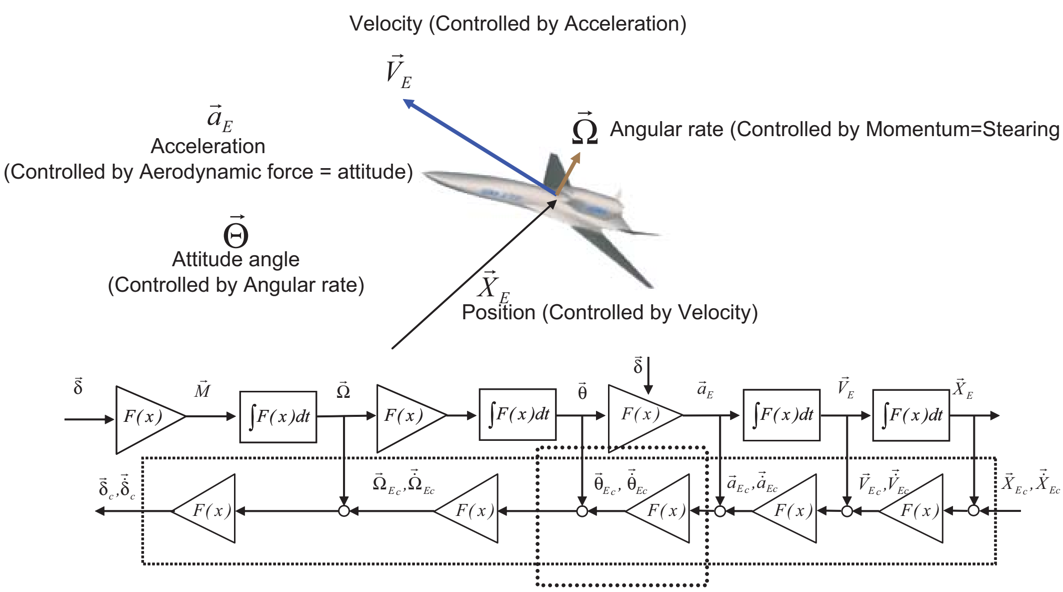

Figure 2: Consept of Hierarchy-Structured Dynamic Inversion

2.3 EXTENDED FORWARD AND INVERSE MODELING

Hence, we extended FIM to deal with these two defects. The NN controller is divided to two components; one is a NNPE, and the other is a HSDI controller.

As is shown in Figure 1, the NNPE gives the estimated value of parameters of the plant to the HSDI controller, and the HSDI controller generates control commands to the plant. The error signal of the plant states are fed back through a NNFM and HSDI, NNPE is trained by parameter estimation errors to estimate proper parameter value which results in appropriate control commands to let the plant follow the reference trajectory.

Extending FIM in this way, Extended Forward and Inverse Modeling (EFIM) inherit the advantages of FIM. In addition, adaptive capability of a NN can be added to a clearly defined controller of HSDI, and NN output can be easily suppressed by a limiter. This will reduce user's reluctance to apply a NN whose behavior is not clearly described. Besides, NNFM can be trained the actual plant online that might be different from the learned design model. In other words, an online learning of NNFM can compensate the modeling error.

If FMNN is not favorable to users, it is possible to replace FMNN with nonlinear mathematical model. In this case, however, the ability to reduce modeling error is lost.

2.4 LEARNING PROCESS OF EFIM

The equations of plant dynamics and control commands generated by DI can be written as

$$\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{u}, \boldsymbol{p}) \tag{5}$$

$$\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{x_m}, \boldsymbol{x}, \tilde{\boldsymbol{p}}), \tag{6}$$

where the state variable is $\boldsymbol{x}$, the state variable of the reference trajectory is $\boldsymbol{x_m}$, control command is $\boldsymbol{u}$, parameter to be estimated is $\boldsymbol{p}$, and its estimated value is $\tilde{\boldsymbol{p}}$. Since the weight of PENN w is updated to reduce the evaluation function

$$r = \frac{1}{2}(\boldsymbol{x_m} - \boldsymbol{x})^T \cdot (\boldsymbol{x_m} - \boldsymbol{x}), \tag{7}$$

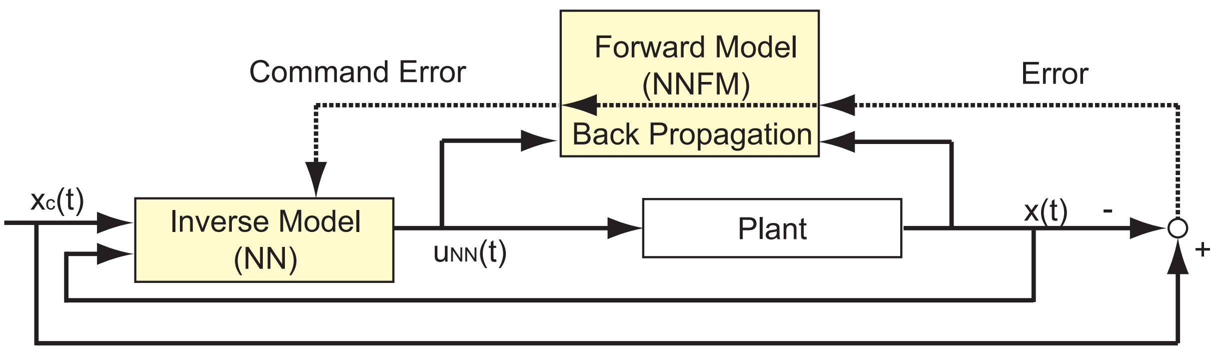

Figure 3: Forward and Inverse Modeling. Yellow blocks show NNs.

the direction of the update of w, Δw, is given by the differentiate of r with w. Therefore, Δw is given as

$$\begin{aligned}
\Delta w &= -\varepsilon \frac{\partial r}{\partial w} \\
&= \varepsilon (\boldsymbol{x_m} - \boldsymbol{x})^T \frac{\partial \boldsymbol{g}}{\partial w} \\
&= \varepsilon (\boldsymbol{x_m} - \boldsymbol{x})^T \left(\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{p}} + \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} \cdot \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{p}} \right) \frac{\partial \boldsymbol{p}}{\partial w},
\end{aligned} \tag{8}$$

where a learning ratio is ε. In this equation, $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{p}}$ denotes the variation of plant characteristics by parameter change, and $\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}} \cdot \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{p}}$ shows the plant behavior according to the command change caused by parameter change. Because the real plant model is different from the design plant model, the differentiate of $\boldsymbol{g}$ is not trivial.

Meanwhile, FMNN learns an approximate plant model, and $\boldsymbol{g}$ can be approximated as

$$\boldsymbol{x}^{\#} = \boldsymbol{g}^{\#}(\boldsymbol{u}, \tilde{\boldsymbol{p}}), \tag{9}$$

where FMNN output is $\boldsymbol{x}^{\#}$. Consequently, Δw can be obtained by using the approximation of the differentiate of $\boldsymbol{g}$,

$$\frac{dw}{dt} \approx \varepsilon (\boldsymbol{x_m} - \boldsymbol{x})^T \left(\frac{\partial \boldsymbol{g}^{\#}}{\partial \boldsymbol{p}} + \frac{\partial \boldsymbol{g}^{\#}}{\partial \boldsymbol{u}} \cdot \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{p}} \right) \frac{\partial \boldsymbol{p}}{\partial w}. \tag{10}$$

This equation shows the update rule of PENN weight.

3 NUMERICAL SIMULATION

3.1 FORMULATION

The EFIM method is applied to a longitudinal motion of ALFLEX[1] linear model. The equation of motion is described as follows

$$\boldsymbol{x} = [q\ \theta\ u\ \dot{z}\ z]^T \tag{11}$$

$$\boldsymbol{u} = [\delta_e\ \delta_{SB}]^T \tag{12}$$

$$\dot{\boldsymbol{x}} = A\boldsymbol{x} + B\boldsymbol{u}, \tag{13}$$

where the state variable is $\boldsymbol{x}$, and the control command is $\boldsymbol{u}$. Each element of system matrices A, B is increased by half, and their response are compared. Then $(4,4)$ element of A turned out to be the most sensitive parameter. For this reason, this parameter is selected to be estimated.

Table 1: Simulation conditions

item	value
Control gain (K_4)	5.236
Control gain ratio (K_i/K_{i+1})	0.4
T_1, T_2	0.4
Control Cycle	0.05 [sec]
Estimation Cycle	0.05 [sec]

In addition to this model, a delay model of actuators are added to see the effect of modeling error between the actual plant model and the design model for HSDI.

$$\dot{\boldsymbol{u}}_d = \begin{pmatrix} -\frac{1}{T_1} & 0 \\ 0 & -\frac{1}{T_2} \end{pmatrix} \boldsymbol{u}_d + \begin{pmatrix} \frac{1}{T_1} & 0 \\ 0 & \frac{1}{T_2} \end{pmatrix} \boldsymbol{u} \tag{14}$$

Altitude and velocity commands are given and the HSDI controller is designed to follow these commands. Note that the actuator delay model is not given to the HSDI controller.

3.2 SIMULATION RESULTS

Table 1 shows simulation conditions, and Figure 4 shows the state variables of the plant. Commands of velocity u and altitude z are given (purple), and HSDI generates reference model behavior (green) and control commands, which result in good tracking performance (blue). Figure 5 shows the actuator commands and parameter estimation results. As time evolves, estimated value of parameter $\boldsymbol{p}$ (green) approaches to the true value (blue), even though the initial estimation (red) has 50% error. Figure 6 shows the tracking errors. From this, we can see that the state errors get smaller, as the estimated value gets closer to the true value.

4 CONCLUSIONS

We proposed a novel control scheme as a candidate of advanced control method to apply for S3TD project. This method consists of NN and DI. it is adaptive and well described, and it requires less resources to tune the gains. Its numerical simulation results showed the effectiveness of this method.

References

[1] Y. Miyazawa, T. Motoda, T. Izumi, and T. Hata, "Longitudinal Landing Control Law for Autonomous Reentry Vehicle," Journal of Guidance, Control and Dynamics, Vol. 22, No. 6, pp.791–800, November–December 1999.

[2] T. Ninomiya, H. Suzuki, and T. Tsukamoto, Evaluation of Guidance and Control Systems of a Balloon-Launched Drop-Test Vehicle, Vol. 43, No. 6, pp1423/1425, 2006

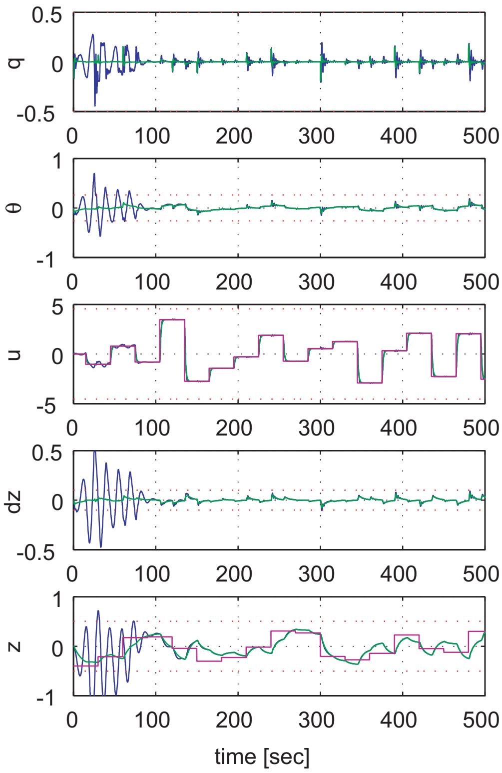

Figure 4: Simulation results: state variables of the plant. Green is reference model, blue is true value, and purple is command.

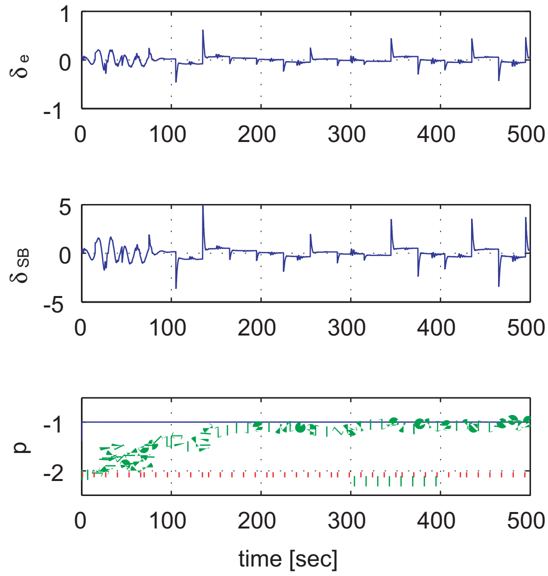

Figure 5: Simulation results: parameter estimation. Blue is true value, green is estimation, and red is initial estimation.

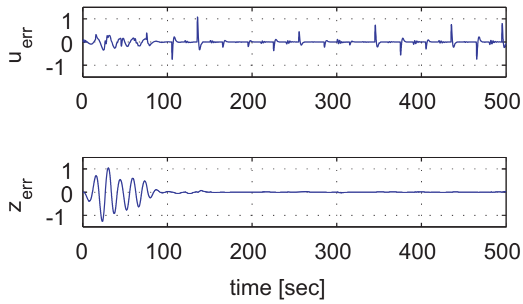

Figure 6: Simulation results: tracking error

[3] M. Sharma, and A. J. Calise, Neural-Network Augmentation of Existing Linear Controllers, Journal of Guidance, Control, and Dynamics, Vol. 28, No. 1, pp.12/19, 2005.

[4] Y. Shin, A. J. Calise, and M. A. Motter, Adaptive Autopilot Designs for an Unmanned Aerial Vehicle, AIAA 2005-6166, 2005.

[5] Kawaguchi J., Miyazawa Y., and Ninomiya, T., "Flight Control Law Design with Hierarchy-Structured Dynamic Inversion Approach," AIAA 2008–6959

[6] M. I. Jordan, and D. E. Rumelhart, "Forward models: Supervised learning with a distal teacher, " Cognitive Science, 1992.

[7] W. T. Miller, R. S. Sutton, and P. J. Werbos, "Neural Networks for Control," MIT press, 1995