How does a robot find redundancy by itself? — A control architecture for adaptive multi-DOF robots —

Koh Hosoda* **and Minoru Asada**

* *Dept. of Adaptive Machine Systems, Osaka University, hosoda@ams.eng.osaka-u.ac.jp*

Abstract. A hybrid architecture consisting of several adaptive controllers is effective to control a multi-degree-of-freedom robot acting in unknown/dynamic environment. For such controllers, it is important to find out whether they are redundant with respect to the sub-tasks given for the sub-controllers. In this paper, we propose an adaptive sub-controller which have an ability to finding out its redundancy automatically. We show a preliminary simulation results to demonstrate the effectiveness of the proposed method.

1 Introduction

The number of degrees of freedom of a robot is concerning to its adaptability. Recently, a robot is required to accomplish a variety of tasks. To accomplish such tasks, the more degrees of freedom the robot has, the more adaptive it becomes. In this sense, to have many degrees of freedom is essential for an adaptive robot.

The number of degrees of freedom is not only concerning to the adaptability, but also to the robustness against malfunction. If the robot has redundant degrees of freedom with respect to the given task, the robot can deal with malfunction of several actuators. The more degrees the robot has, the more robust it becomes.

If the robot has many degrees of freedom, however, the effort to program all degrees to achieve a behavior may be large. It may be incredibly complicated if one tries to program several behaviors and to switch them according to situations. In this sense, programming a behavior with a multi-DOF robot is not a good idea. For example, if one uses the subsumption architecture approach(Brooks, 1986), which is famous to make an adaptive robot, to program a muti-degree-of-freedom robot, it will be hard to program each behavior because of too many actuators.

One method to deal with many degrees of freedom easier is to make a hybrid control structure consisting of several sub-controllers. Each sub-controller takes care of a part of degrees and by a combination of several sub-controllers a behavior is emerged. To make the robot adaptive, each sub-controller must have adaptivity at each task sub-space.

Our final goal is to build an autonomous adaptive robot that has many degrees of freedom in dynamic/unknown environments. We have proposed to use such a hybrid structure of adaptive controllers to control multi-degree-of-freedom robots (Hosoda et al., 1998; Miyashita et al., 1998a; Miyashita et al., 1998b). In these papers, however, each sub-controller is adaptive, but cannot finding out its redundancy with respect to the given sub-task. If a sub-controller can find out the redundancy, it can spare the redundant degrees of freedom to other controllers, and as a consequence, the over all hybrid structure becomes more adaptive.

In this paper, we propose a new method for an adaptive controller to finding out its redundancy automatically. The remainder of this article is organized as follows. First, we introduce the adaptive sub-controller that utilize a least-mean-square method. Then, by utilizing UD factorization of the covariance matrix, a method to derive the redundancy is proposed. Finally, preliminary simulation results are shown to demonstrate the effectiveness of the proposed method.

2 A Hybrid Control Architecture for multi-DOF robots

2.1 Sub-controllers

The i-th sub-task for a sub-controller is defined in its own task sub-space. Typically, it is an output space of an external sensor. The sensor output $\boldsymbol{x}_i$ is a function of actuator displacement $\boldsymbol{\theta}$,

$$\boldsymbol{x}_i = \boldsymbol{x}_i(\boldsymbol{\theta}). \tag{1}$$

Differentiating the equation with respect to the sampling rate, we can get

$$\Delta \boldsymbol{x}_i(k) = \boldsymbol{J}_i(k)\Delta\boldsymbol{\theta}(k), \tag{2}$$

in the k -th sampling step, where $\boldsymbol{J}_i \stackrel{\triangle}{=} \partial\boldsymbol{x}_i/\partial\boldsymbol{\theta}$.

The i-th sub-task may be define in terms of a desired value $\boldsymbol{x}_{id}$. If all the actuators of the robot are velocity-controlled, we can derive a velocity command for i-th task from eq.2,

$$\boldsymbol{u}(k) = \boldsymbol{J}_i(k)^+\boldsymbol{K}_i(\boldsymbol{x}_{id}(k)-\boldsymbol{x}_i(k))+(\mathbf{I}-\boldsymbol{J}_i(k)^+\boldsymbol{J}_i(k))\boldsymbol{\xi} \tag{3}$$

where $\mathbf{I}$, $\boldsymbol{K}_i$ and $\boldsymbol{\xi}$ are an identity matrix, a feedback gain matrix, and an arbitrary vector that denotes the redundancy of the controller with respect to the given task to follow $\boldsymbol{x}_{id}$. $\boldsymbol{A}^+$ denotes a pseudo-inverse of a matrix $\boldsymbol{A}$. By utilizing the last term in the right hand side in eq.(3), we can make a hybrid structure of the sub-controllers(Hosoda et al., 1998).

2.2 Adaptation of the controllers

If there is no *a priori* knowledge on the robot and the environment, the matrix $\boldsymbol{J}_i(k)$ is unknown. To make the robot adaptive to change of the environment, a mechanism to estimate the matrix is needed. An estimator utilizing least-mean-square method is(Hosoda et al., 1998)

$$\begin{aligned}\widehat{\boldsymbol{J}}_i(k) &= \widehat{\boldsymbol{J}}_i(k-1) + \{\Delta\boldsymbol{x}_i(k) - \widehat{\boldsymbol{J}}_i(k-1)\Delta\boldsymbol{\theta}(k)\} \\ &\quad \frac{\Delta\boldsymbol{\theta}(k)^T\boldsymbol{P}(k-1)}{\rho + \Delta\boldsymbol{\theta}(k)^T\boldsymbol{P}(k-1)\Delta\boldsymbol{\theta}(k)}\end{aligned} \tag{4}$$

where $(0 < \rho < 1)$ is a forgetting factor, and $\boldsymbol{P}(k)$ is a covariance matrix calculated as

$$\begin{aligned}\boldsymbol{P}(k) &= \frac{1}{\rho}\{\boldsymbol{P}(k-1) \\ &\left.\quad - \frac{\boldsymbol{P}(k-1)\Delta\boldsymbol{\theta}(k)\boldsymbol{\theta}(k)^T\boldsymbol{P}(k-1)}{\rho + \Delta\boldsymbol{\theta}(k)^T\boldsymbol{P}(k-1)\Delta\boldsymbol{\theta}(k)}\right\}\end{aligned} \tag{5}$$

We have already demonstrated that many kinds of robots can be controlled by this adaptive controller without any *a priori* knowledge on the robot and the environment(Hosoda and Asada, 1997).

2.3 Estimating the redundancy automatically

Utilizing a characteristic of the estimator (5), the robot can find the redundancy automatically. This may allows the control architecture being adaptive and robust.

If the robot is redundant to achieve the i-th sub-task, the variation of the actuator displacement $\Delta\boldsymbol{\theta}$ is not uniform over the task space. Therefore, by observing the covariance matrix $\boldsymbol{P}$, which represents the variation of $\Delta\boldsymbol{\theta}$, the robot can find out its redundancy by itself.

It is known if the variation of $\Delta\boldsymbol{\theta}$ is not uniform, the covariance matrix $\boldsymbol{P}$ cannot keep the positiveness, and therefore the least-mean-square based estimator will lose the numerical stability (Ifeachor and Jervis, 1993). To avoid such instability, a technique so called UD factorization is proposed(Bierman, 1976), which we show more details in the appendix. Following the method, the covariance matrix $\boldsymbol{P}$ is factorized

$$\boldsymbol{P} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T, \tag{6}$$

where $\boldsymbol{U}$ is an upper triangular matrix whose diagonal elements are all 1, and $\boldsymbol{D}$ is a diagonal matrix. Instead of updating $\boldsymbol{P}$ by eq.(5), we can update $\boldsymbol{U}$ and $\boldsymbol{D}$ (see appendix) so as to ensure the positiveness of the matrix $\boldsymbol{P}$ and ensure the numerical stability.

Note that if the positiveness of $\boldsymbol{P}$ is week, one of the diagonal element of $\boldsymbol{D}$ is near to 0. Therefore, if the control input for the robot $\Delta\boldsymbol{\theta}$ is not uniform, we can find out by observing $\boldsymbol{D}$.

Let $\boldsymbol{D}$ and $\boldsymbol{U}$ be

$$\boldsymbol{D} = \mathrm{diag}\begin{bmatrix} d_1 & \cdots & d_n \end{bmatrix}, \tag{7}$$

$$\begin{aligned}\boldsymbol{U} &= \begin{bmatrix} 1 & & * \\ & \ddots & \\ 0 & & 1 \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{u}_1 & \cdots & \boldsymbol{u}_n \end{bmatrix},\end{aligned} \tag{8}$$

respectively. If the robot is redundant, a diagonal element d_j of $\boldsymbol{D}$ becomes very small. If $d_j = 0$, the corresponding direction of $\Delta\boldsymbol{\theta}$ is not anymore effective to the estimation, that is, the direction is redundant with respect to the sub-task. Every term

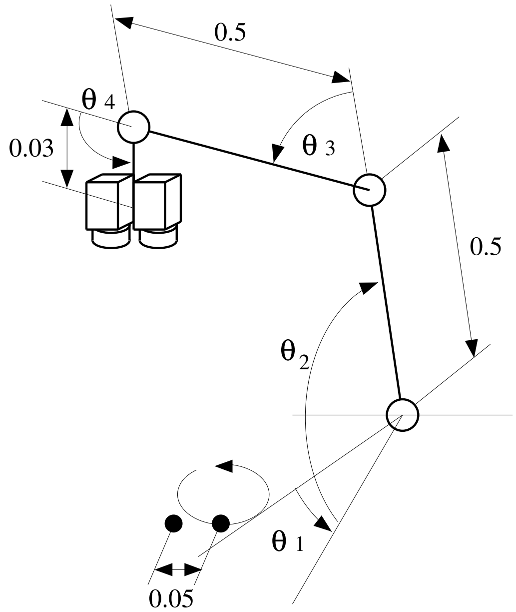

Figure 1: A 4 DOF manipulator with 2 cameras is used for the simulation (Unit is [m]).

concerning about $\Delta\boldsymbol{\theta}$ in the estimator (4) is multiplied with $\boldsymbol{P}$:

$$\begin{aligned} \boldsymbol{P}\Delta\boldsymbol{\theta} &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T\Delta\boldsymbol{\theta} \\ &= \boldsymbol{U}\begin{bmatrix} \ddots & & \\ & d_j & \\ & & \ddots \end{bmatrix}\boldsymbol{U}^T\Delta\boldsymbol{\theta}, \end{aligned}$$

Therefore, the redundant direction is given as a column vector of $\boldsymbol{U}^{-T}$. To eliminate the redundancy, we can remove this direction form estimated matrix $\widehat{\boldsymbol{J}}_i$.

3 Simulation

We demonstrate simulation results, to shown that the robot can find out the redundancy with respect to one sub-task, and can apply another task utilizing the redundant space.

3.1 Robot System

A robot used for the simulation had 4 DOFs with two cameras depicted in figure 1. By two cameras the robot observes two positions of image points. The first sub-task given to the robot is to keep these positions constant in the image planes, that is, visual servoing task. The points are moving along a circle whose diameter is 0.1[m] in 5[s], 10 times.

Two cameras are gazing at 2 points, therefore the image feature vector is $\boldsymbol{x} \in \Re^8$. The robot has 4 DOFs, $\boldsymbol{\theta} \in \Re^4$. Two points are moving on a surface along the circle, therefore the DOF needed for the task is 3.

The forgetting factor of the estimator $\rho = 0.8$. The initial matrix of $\boldsymbol{P}$ is an identity matrix. The initial matrix for the estimated Jacobian matrix is given as an arbitrary matrix

$$\widehat{\boldsymbol{J}}(0) = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \\ 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix},$$

because we assume that the controller does not have any *a priori* knowledge on the parameters of the robot and of the environment. Note that the rank of this matrix is 4. This means that all the 4 DOFs are utilized for the visual tracking task at the initial condition.

To help the understanding of the readers, we show the true Jacobian matrix:

$$\widehat{\boldsymbol{J}}(0) = \begin{bmatrix} 2300 & 0 & 0 & 0 \\ 100 & 0 & 1300 & 1300 \\ 2400 & 300 & 200 & 0 \\ 0 & 0 & 1300 & 1300 \\ 2400 & -300 & -200 & 0 \\ 100 & 0 & 1300 & 1300 \\ 2300 & 0 & 0 & 0 \\ 0 & 0 & 1300 & 1300 \end{bmatrix}.$$

Note that the rank of this matrix is 3.

3.2 Simulation Results

Here we show two simulation results:

case 1 applying visual servoing control without redundancy estimator, and

case 2 applying visual servoing control with redundancy estimator.

In case 1, the robot cannot find out its redundancy with respect to the first "visual servoing" subtask. In case 2, however, it will find redundancy. Utilizing the redundancy, the second subtask to make θ_3 converge to $-\pi/3$ is applied. The second subtask is performed to minimize the index

$$Q = \{\theta_3 - (-\pi/3)\}^2.$$

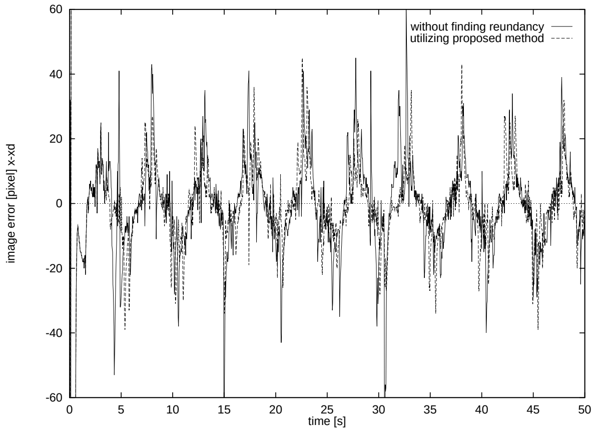

Figure 2: Simulation result 1 : image error in camera 1, image feature 1, x- coordinate ($x_1 - x_{1d}$)

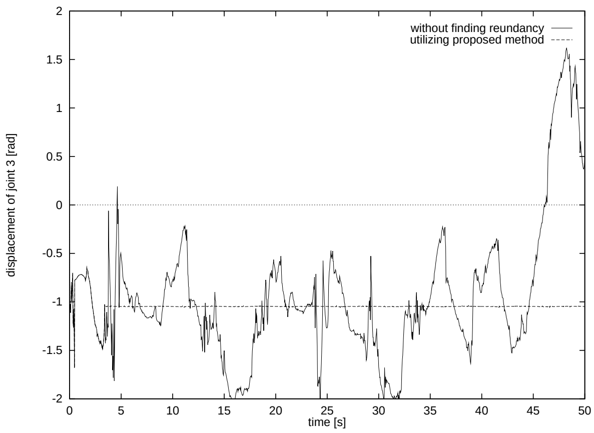

Figure 3: Simulation result 2 : joint 3 displacement θ_3

Simulation results are shown in figures 2 and 3. By utilizing the on-line estimator, the visual servoing task is achieved without knowing about structure and parameters of the robot and the environment in both two cases. In figure 3, around 3 [s], the proposed estimator finds out the redundancy with respect to the first task. After that, θ_3 is converging to $-\pi/3$ utilizing the redundancy.

3.3 Summary and Conclusions

In this paper, we have proposed an adaptive controller that can find out its redundancy by itself. By combining such controllers, it is very easy to construct a hybrid control architecture to control multi-degree-of-freedom robot. At first, only the first subtask controller is working, then it finds out its redundancy. Then it will spare the redundancy with second subtask controller, and so on. Note that the first controller does not simply inhibit the lower controllers, but spares the found redundancy with them. As a consequence, a behavior is emerged by a combination of several controllers.

The example demonstrated in this paper is so simple. To show the effectiveness of the controller, a more complicated robot with more complicated tasks has to be used. Now an experiment with 7 DOF robot arm with cameras is on going.

A UD factorization (Bierman, 1976)

If the initial UD factorization of $\boldsymbol{P}$ is given, updated matrices $\widehat{\boldsymbol{U}}$ and $\widehat{\boldsymbol{D}}$ are obtained from $\boldsymbol{U}$ and and $\boldsymbol{D}$:

$$\begin{aligned} \boldsymbol{f} &= \boldsymbol{U}^T \Delta\boldsymbol{\theta}, \\ v_i &= d_i f_i \qquad (i = 1, \cdots, n, n \text{is dimension of } P), \\ \alpha_1 &= \rho + v_1 f_1, \\ \hat{d}_1 &= d_1 \gamma / \alpha_1, \\ b_1 &= v_1, \end{aligned}$$

As for the $j = 2, \cdots, n$ -th elements, they are updated as follows, where $\triangleq$ denotes overwite in the program manner,

$$\begin{aligned} \alpha_j &= d_{j-1} + f_j v_j, \\ \hat{d}_j &= d_j \alpha_{j-1} / \alpha_j, \\ b_j &\triangleq v_j, \\ p_j &= -f_j / \alpha_{j-1}, \end{aligned}$$

$$\left.\begin{aligned} \widehat{U}_{ij} &= U_{ij} + b_i p_j \\ b_i &\triangleq b_1 + U_{ij} v_j \end{aligned}\right\} (i = 1, \cdots, j). \qquad (9)$$

$$d_i \triangleq d_i / \rho \qquad (i = 1, \cdots, n). \qquad (10)$$

References

Bierman, G. J. (1976). Measurement updating using the u–d factorization. *Automatica*, 12:375–382.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE J. of Robotics and Automation*, RA-2(1):14–23.

Hosoda, K. and Asada, M. (1997). Adaptive visual servoing for various kinds fo robot systems. In *Preprints of 5th International Symposium on Experimental Robotics*, pages 451–462.

Hosoda, K., Igarashi, K., and Asada, M. (1998). Adaptive hybrid control for visual servoing and force servoing in an unknown environment. *IEEE Robotics and Automation Magazine*, 5(4):39–43.

Ifeachor, E. C. and Jervis, B. W. (1993). *Digital Signal Processing – A Practical Approach –*. Addison-Wesley.

Miyashita, T., Hosoda, K., and Asada, M. (1998a). Hybrid structure of reflective gait control and visual servoing for walking. In *Proc. of the 1998 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 229–234.

Miyashita, T., Hosoda, K., and Asada, M. (1998b). Reflective walk based on lifted leg control and vision-cued swaying control. In *Proc. of 1998 BSMEE International Symposium on Climbing and Walking Robots (CLAWAR'98)*, pages 349–354.