A Controller Design Method Based on Functionality

Toshiaki Tsuji, Kouhei Ohnishi
Department of System Design Engineering,
Faculty of Science and Technology,
Keio University
Email: tsuji@sum.sd.keio.ac.jp,ohnishi@sd.keio.ac.jp

Asif Sabanovic
Mechatronics Program,
Faculty of Engineering and Natural Sciences,
Sabanci University
Email: asif@sabanciuniv.edu

***Abstract*— Robots in human environment need some redundancy for adaptation. It is therefore necessary to solve a complicated design issue of large-scale systems with hyper-DOF. The authors have proposed a design framework of functionality to solve the issue. Although the framework deals with task shifts and exception handling in a unified manner, it was limited to a multi-robot system in one-dimensional space. This study expands the framework to multi-DOF robots in three-dimensional space and shows a developed form. Cooperative Jacobian matrix is introduced for coordinate transformation. A new problem of interactions among function-based systems occurs along with the expansion. Disturbance observer is applied on each actuator to eliminate the interactions. The simplicity and explicitness of function-based controller design carry on despite the expansion since function-based systems are decoupled with disturbance observer.**

I. Introduction

Ability of motion control has recently improved due to development of mechatronics technology. From now on, motion control systems such as robots, electric vehicles and so on are expected to expand their applicable scope to human environment. Robots in human environment need some redundancy for adaptation. Furthermore, they are often required to execute a complicated task concurrent with adapting to environment. It is therefore necessary to solve a design issue of large-scale systems with a complicated task.

Decentralized control is a promising method for large scale systems. It is preeminent in many features such as flexibility, fault tolerance, expandability, and rapid response. Many studies applied it to robot control systems [1]–[4]. Among them, interesting concepts such as subsumption architecture [1], multi-agent system [2] and cell structure [3] have been proposed. Artificial intelligence is often introduced to solve the design issue of these methods. Decentralized control is also utilized for fault tolerant systems [5]. More explicit and simple framework in view of controller design is desired although the methods for decentralized control systems are interesting as concepts.

Decomposition block control [6] is one of the efficient solutions. It transforms a control system into BCD-form and simplifies the design problem. Arimoto and Nguyen showed that overall control input can be designed by linear superposition of all signals under the condition of unique stationary resolution of the controlled position variables [7]. Okada, Tatani and Nakamura proposed a method to symbolize the robot motion based on the singular value decomposition [8]. Lee and Li presented a decoupled design method that makes a bilateral control system behave as a common passive rigid mechanical tool [9]. Tsuji, Nishi and Ohnishi proposed a framework of controller design based on functionality [10]. Onal and Sabanovic implemented a sensitive bilateral control using sliding mode control applying functionality [11]. Function is a minimum component which is independent to each other and the idea follows the principle of superposition. The framework provides a unified design method that deals with both complicated task variation and exception handling. Although controller design becomes simple and explicit with the framework, the study was limited to one-dimensional space. This paper therefore extends the framework for robots in three-dimensional space. Disturbance observer [12] is applied to decouple functions. An extended form of function-based controller design is described.

II. Concept of functionality

A. Definition of function

This part describes the idea of functionality [13] to concretize the role of the control system. At first, a system role is defined as follows:

Definition 1 "System role" is a description on the requirement from a user to a robot control system.

The system role represents a momentary feature of the control system and does not include sequential information. The control system should be designed to satisfy this system role. It is, however, difficult to associate a system role with a controller directly since the system role consists of abstract words. The idea of function is introduced to concretize the system role as follows.

Definition 2 "Function" is a minimum component of a system role. Conversely, the system role is described as a combination of functions.

Robots often need to execute multiple actions concurrently as the operation becomes complicated. For example, robots have to move after they grasp a load to achieve a conveying

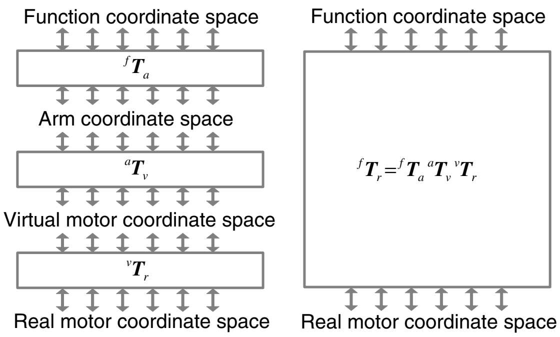

Fig. 1. Outline of coordinate transformation

operation. In short, the robots have to "move" and "grasp" at the same time. Robots may be obliged to execute and switch a wide variety of actions especially in a large scale system. The idea of functionality is to express these actions in the unit of function.

B. Coordinate transformation based on function

The controller design based on functionality needs coordinate transformation. This subsection describes an extended form of the coordinate transformation.

There exists many kinds of functions for tasks, exception handling, and so on. Various kinds of information such as arm tip position, motor angles and modal information are required for the functions. Multi-layered transformation is therefore introduced. An outline of the transformation is shown in Fig. 1.

The coordinate transformation introduced in [10] is to derive function coordinate space from workspace information of each robot. Note that workspace of a one-dimensional mobile robot coincides with space of motor angle. A Jacobian matrix is known for transformation from joint space to workspace. Transformation from real motor coordinate space to virtual motor coordinate space of sum and difference motor is introduced for a twin-drive system [14].

Several coordinate spaces are transformed through transformation matrices. $^f\boldsymbol{T}_r$, a transformation matrix from real motor coordinate space to function coordinate space, is derived by multiplying the matrices between each space.

At first, function coordinate space is transformed from arm coordinate space (i.e. workspace of each robot) as follows:

$$\boldsymbol{x}_f = {}^f\boldsymbol{T}_a \boldsymbol{x}_a \quad (1)$$
$$\dot{\boldsymbol{x}}_f = {}^f\boldsymbol{T}_a \dot{\boldsymbol{x}}_a \quad (2)$$
$$\ddot{\boldsymbol{x}}_f = {}^f\boldsymbol{T}_a \ddot{\boldsymbol{x}}_a \quad (3)$$
$$\boldsymbol{f}_f = {}^f\boldsymbol{T}_a \boldsymbol{f}_a \quad (4)$$
$$\boldsymbol{x}_a = [\boldsymbol{x}_{a1}, \boldsymbol{x}_{a2}, \cdots, \boldsymbol{x}_{am}]^T$$
$$\boldsymbol{f}_a = [\boldsymbol{f}_{a1}, \boldsymbol{f}_{a2}, \cdots, \boldsymbol{f}_{am}]^T$$

Here, $\boldsymbol{x}_{ai} \in \boldsymbol{R}^3$ and it denotes position of an end effector on the th robot. $\boldsymbol{f}_{ai} \in \boldsymbol{R}^3$ and it denotes external force on the end effector. The subscript denotes function coordinate space and the subscript denotes arm coordinate space. $^f\boldsymbol{T}_a \in \boldsymbol{R}^{N \times M}$, $^a\boldsymbol{T}_v \in \boldsymbol{R}^{M \times M}$, $^v\boldsymbol{T}_r \in \boldsymbol{R}^{M \times M}$, is total number of robots, is total DOF of robots, and is total DOF of functions.

$^f\boldsymbol{T}_a$ corresponds to the transformation matrix in [10]. It is composed of $1, 0$ and -1 to calculate sum and difference information of related arm tip variables.

As shown from (1) to (4), position, velocity, acceleration and external force are all transformed by $^f\boldsymbol{T}_a$. Position of arm tip is calculated by direct kinematics based on a real motor response. Force on arm tip is measured by a force sensor or reaction force observer (RFOB) [15]. Then, position and force information for function-based controller are derived from (1) and (4), respectively. Velocity and acceleration information on function coordinates are derived from a real motor response by (5) and (6).

$$\dot{\boldsymbol{x}}_f = {}^f\boldsymbol{T}_r \dot{\boldsymbol{x}}_r \quad (5)$$
$$\ddot{\boldsymbol{x}}_f = {}^f\boldsymbol{T}_r \ddot{\boldsymbol{x}}_r \quad (6)$$
$${}^f\boldsymbol{T}_r = {}^f\boldsymbol{T}_a {}^a\boldsymbol{T}_v {}^v\boldsymbol{T}_r \quad (7)$$

where, $^a\boldsymbol{T}_v$ is a transformation matrix similar to a Jacobian matrix. It transforms virtual motor coordinate space to arm coordinate space. $^v\boldsymbol{T}_r$ is a transformation matrix from real motor coordinate space to virtual motor coordinate space.

$^v\boldsymbol{T}_r$ is a specific transformation matrix only for a twin drive system. It is a unit matrix $\boldsymbol{I}$ for other general systems. In a one-dimensional system, the Jacobian matrix of each robot $^a\boldsymbol{T}_v$ is also a unit matrix $\boldsymbol{I}$.

$^f\boldsymbol{T}_r$ can be explained as an extended Jacobian matrix. It is extended for a twin-drive system and cooperative work of a multi-robot system. It is therefore called "cooperative Jacobian matrix". $^f\boldsymbol{T}_a$, which is simply named "transformation matrix" in [10], is called "function matrix" for distinction.

Control input $\boldsymbol{u}_f$ is derived from controllers on function coordinate space. Here, $\boldsymbol{u}_f$ is in acceleration dimension. Torque input in real motor coordinate is derived from (8).

$$\boldsymbol{\tau}_r = \boldsymbol{M}_n {}^f\boldsymbol{T}_r^+ \boldsymbol{u}_f \quad (8)$$
$${}^f\boldsymbol{T}_r^+ = ({}^f\boldsymbol{T}_r^T {}^f\boldsymbol{T}_r)^{-1} {}^f\boldsymbol{T}_r^T$$

Here, $\boldsymbol{M}_n \in \boldsymbol{R}^{M \times M}$ and it is the nominal value of the inertia matrix of robots. The condition for deriving torque input is

$$(\boldsymbol{M}_n {}^f\boldsymbol{T}_r^+) = \quad (9)$$

Therefore, if any of functions are dependent on each other, a new function should be added. On the other hand, if functions are overfull, one of the functions with the lowest priority should be halted. The entire block diagram is shown in Fig. 2.

C. Dynamics in function coordinate space

It is to be anticipated from the name of cooperative Jacobian matrix that the coordinate transformation is for kinematics of a large scale system. Virtual dynamics in a function coordinate interferes with each other, contrary to the method proposed in [10]. The interference occurs due to the generalization to three-dimensional systems.

Disturbance observer is applied to all of real motors in this method to cancel the interferences. It is known that the

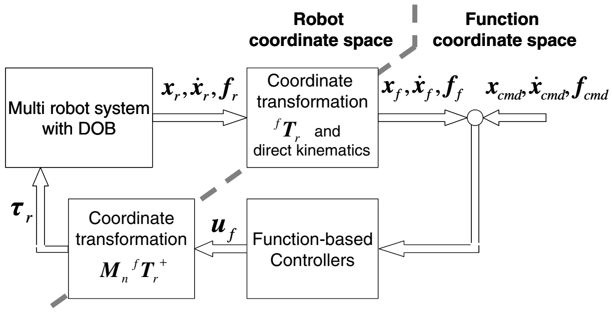

Fig. 2. Block diagram of function-based control system

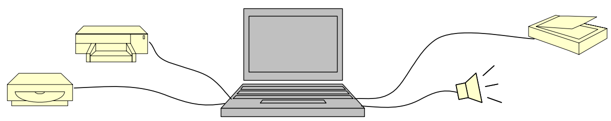

Fig. 3. Design as detachable component

plant works as a nominal system when acceleration control is acquired with disturbance observer [12]. Hence inputs from position/force controller based on functions are superposed without any interference.

D. Concept of function-based controller design

The main idea of function-based controller design is to design each controller as a detachable component. It is similar to design of peripheral equipment for PC as shown in Fig. 3. Many kinds of function-based controllers are designed in advance like peripheral equipment. Among them, requisite functions are exerted depending on the varying system role. Great patterns of tasks are realized with such a framework. Furthermore, the design is still simple and explicit. In sum, this framework is useful for control of robots adaptive to complicated environments since it solves the issues of task variation and exception handling of complicated systems [10].

III. Configuration of Function-Based Control System

A. Procedures of controller design

A design flow of function-based control system is shown in Fig. 4. Firstly, the system role is determined by a designer of the control system. Secondly, the designer divides the system role into functions. Thirdly, a priority order of functions is determined. Important functions should be secured even if the number of active functions alters. Then, the transformation matrix ${}^f\boldsymbol{T}_r$ is derived. The number of functions is modified so that rank of ${}^f\boldsymbol{T}_r$ agrees with total DOF of robots . Otherwise, (9) is unsatisfied. Finally, function-based controllers are designed individually.

B. Reconfiguration for alteration of system role

When the system role alters, combination of functions and its transformation matrix should be modified. At first, new combination of task functions should be given by the designer. Here, a task function is a function to acquire the system role while a performance-limit function is a function to deal with an exception. In the next place, the transformation matrix should be modified along with the functions. Majority of task functions control relative position or relative force between arm tips. In this study, ${}^f\boldsymbol{T}_a$ denotes the relation between arm tips. In sum, ${}^f\boldsymbol{T}_a$ should be modified in a similar way in [10] by modifying $\boldsymbol{T}$ when the system role alters.

C. Reconfiguration for exception handling

Reconfiguration for exception handling is more difficult compared to that for alteration of the system role. There are three reasons: exceptions occur all of a sudden; the control system should choose the combination of functions autonomously; not only ${}^f\boldsymbol{T}_a$ but also ${}^a\boldsymbol{T}_v$ or ${}^v\boldsymbol{T}_r$ should be modified since performance-limit functions that deal with exceptions are often based on a real motor output or a virtual motor output. A method to modify a transformation matrix is introduced below.

${}^f\boldsymbol{T}_r$ is described as follows:

$$ {}^f\boldsymbol{T}_r = \left[\begin{array}{cccc} {}^f\boldsymbol{t}_{r1}^T & {}^f\boldsymbol{t}_{r2}^T & \cdots & {}^f\boldsymbol{t}_{rN}^T \end{array} \right]^T \tag{10} $$

${}^f\boldsymbol{t}_{ri} \in \boldsymbol{R}^M$, it extracts the coordinate of the th function. It is called “function mode” and depends on the characteristics of the function. Function modes for task functions are derived all at once from (7).

On the other hand, performance-limit functions, which are activated in a special case also have their function modes. The function mode of the performance-limit function should be derived individually when the function is activated. The function mode of the performance-limit function is derived from various ways since performance-limits may exist in each layer of the multi-layered coordinate transformation. For example, a function mode of a velocity-limit function on the th real motor is derived as follows:

$$ \begin{aligned} {}^f\boldsymbol{t}_{r,PL}^T &= \left[\begin{array}{cccc} 1 & 2 & \cdots & M \end{array} \right] \\ & \begin{cases} i = 1 & = \\ i = 0 & \end{cases} \end{aligned} \tag{11} $$

Here, ${}^f\boldsymbol{t}_{r,PL}$ denotes a function mode of a performance-limit function.

A position-limit function for avoidance of a singular point is shown as another example of a performance-limit function. A joint angle of the twin drive system corresponds to a response value of a virtual differential motor. Hence a singular point

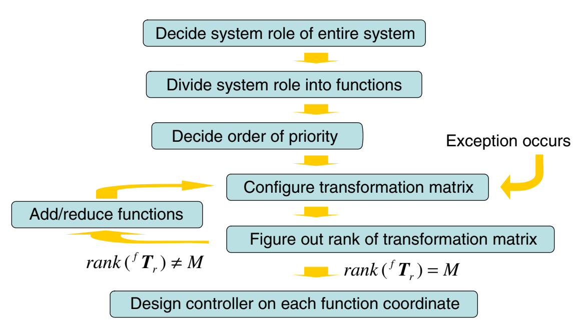

Fig. 4. Flow of controller design

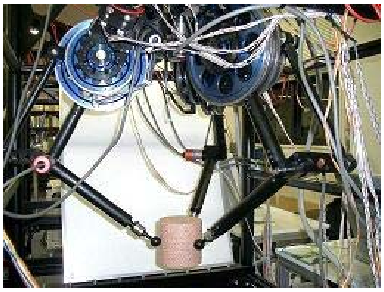

Fig. 5. Parallel link manipulators

is avoided by setting a position-limit on the virtual motor. A function mode of the position-limit function for the th virtual motor is derived as follows:

$$^{f}\boldsymbol{t}_{r,PL} = ^{v}\boldsymbol{t}_{rk} \quad (12)$$

where, $^{v}\boldsymbol{T}_{r} = \left[\begin{array}{cccc} ^{v}\boldsymbol{t}_{r1}^{T} & ^{v}\boldsymbol{t}_{r2}^{T} & \cdots & ^{v}\boldsymbol{t}_{rN}^{T} \end{array} \right]^{T}$.

A function mode of a position-limit function on an arm tip is derived as follows:

$$^{f}\boldsymbol{t}_{r,PL} = ^{a}\boldsymbol{t}_{rk} \quad (13)$$

where, $^{a}\boldsymbol{T}_{r} = \left[\begin{array}{cccc} ^{a}\boldsymbol{t}_{r1}^{T} & ^{a}\boldsymbol{t}_{r2}^{T} & \cdots & ^{a}\boldsymbol{t}_{rN}^{T} \end{array} \right]^{T}$. It is assumed that the position limit is set for the th element of $\boldsymbol{x}_{a}$.

A procedure for exception handling is shown as follows:

1) keep observing variables for discriminating exceptions;
2) select a relevant performance-limit function when one of the variables exceeds its limit;
3) derive $^{f}\boldsymbol{t}_{r,PL}$, a function mode of the performance-limit function;
4) derive $^{f}\boldsymbol{t}_{r,low}$, the function mode of the lowest-priority function;
5) if $^{f}\boldsymbol{t}_{r,PL} \cdot {}^{f}\boldsymbol{t}_{r,low} \neq 0$, substitute $^{f}\boldsymbol{t}_{r,PL}$ to $^{f}\boldsymbol{t}_{r,low}$ in $^{f}\boldsymbol{T}_{r}$;
6) if $^{f}\boldsymbol{t}_{r,PL} \cdot {}^{f}\boldsymbol{t}_{r,low} = 0$, select the function with the next-lowest priority, derive its function mode $^{f}\boldsymbol{t}_{r,low}$, and go to 5).

IV. Function-Based Controller Design for Cooperative Grasping Motion

A control system for parallel link manipulators is shown in this section as a typical example of a function-based system. A picture of manipulators is shown in Fig. 5. The modeling of the manipulators is shown in [16]. The entire system consists of three parallel link manipulators with 3 DOF. There are 6 motors on each manipulator since the manipulator consists of twin drive systems.

An overview of the work with the control system is shown in Fig. 6. An operator holds one of the manipulators and handles an object with it.

Three manipulators are fixed with orientation difference of 120 degrees respectively. Absolute position of the arm tip is presented by cylindrical coordinates as shown in (14).

$$\boldsymbol{x}_{ai} = [\quad _{i} \quad _{i} \quad _{i} \,]^{T} \quad (14)$$

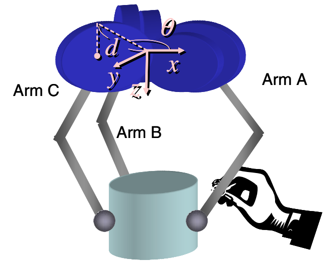

Fig. 6. Overview of work

where denotes distance from the z-axis based on the center of three manipulators, denotes up-down position, and denotes rotation angle in a horizontal plane.

A control system to execute the following operation was developed as a typical example of a human support operation with task variation.

Firstly in Step 1, the arm tips of the three manipulators move in compliance with external force only in the grasping mode, a mode that denotes sum of $_{A}$, $_{B}$ and $_{C}$. Step 2 starts after the operator inserts a cylindrical object between the three arm tips. In Step 2, the object is cooperatively grasped by the three arms while its position and attitude is kept constant under external force. In Step 3, the object moves in compliance with external force only in the pitching mode while it is grasped. Its position is kept constant at that time. In Step 4, it moves only in the up-down mode while its attitude is kept constant and it is grasped. Task functions for acquiring the system roles in Step 1 to Step 4 are shown in Table I. The overview of the coordinate transformation is shown in Fig. 7.

Here, RC, SC, VC, and GR denote functions of rigid coupling, spring coupling, velocity control, and grasping, respectively. Numbers in parentheses denote the priority order of the function. The grasping function has higher priority to secure the object. Velocity control functions on sum coordinates keep velocity of virtual sum motors constant to avoid stick-slip phenomenon [14]. Outputs of the functions have relatively

TABLE I

FUNCTIONS FOR PARALLEL LINK MANIPULATORS

	Step 1	Step 2	Step 3	Step 4
Based on d				
Mode 1(Grasping)	**SC** (1)	**GR** (1)	**GR** (1)	**GR** (1)
Mode 2	RC (2)	RC (2)	RC (2)	RC (2)
Mode 3	RC (3)	RC (3)	RC (3)	RC (3)
Based on θ				
Mode 1(Rolling)	RC (9)	RC (9)	RC (9)	RC (9)
Mode 2	RC (8)	RC (8)	RC (8)	RC (8)
Mode 3	RC (7)	RC (7)	RC (7)	RC (7)
Based on z				
Mode 1(Up-down)	RC (6)	RC (6)	RC (6)	**SC** (6)
Mode 2(Pitching)	RC (5)	RC (5)	**SC** (5)	RC (5)
Mode 3	RC (4)	RC (4)	RC (4)	RC (4)
Based on virtual sum motors				
Mode 1	VC (10)	VC (10)	VC(10)	VC(10)
⋮	⋮	⋮	⋮	⋮
Mode 9	VC (18)	VC (18)	VC(in	VC(18)

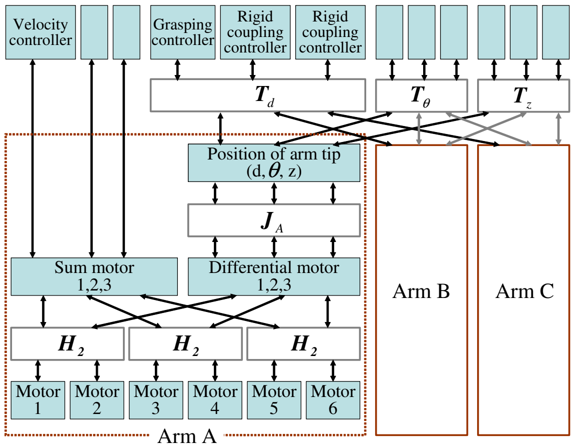

Fig. 7. Overview of entire coordinate transformation

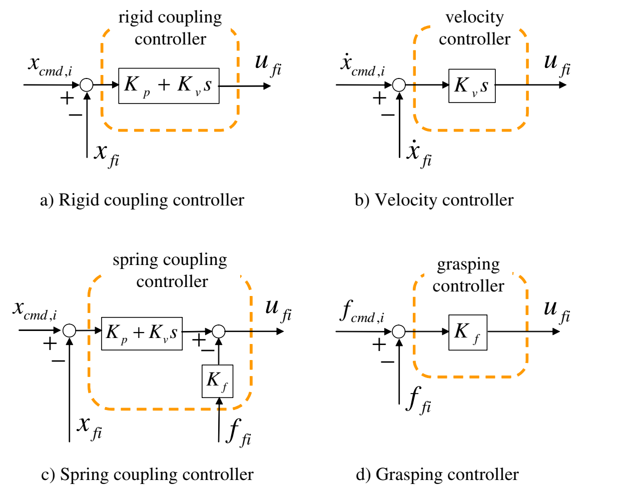

Fig. 8. Block diagram of functions

small effects on the operation. The velocity control functions therefore have lower priority. The priority order of other task functions is given arbitrarily. Performance-limit functions exist in addition to the task functions. Priority of performance-limit functions are set higher than that of task functions so that they are compulsively activated when exceptions occur.

The function matrix ${}^f\boldsymbol{T}_a$ for such functions is given as follows:

$$ {}^f\boldsymbol{T}_a = \begin{bmatrix} \boldsymbol{T}_d & & & \\ & \boldsymbol{T}_\theta & & \\ & & \boldsymbol{T}_z & \\ & & & \boldsymbol{I}_9 \end{bmatrix} {}^f\boldsymbol{S}_a \tag{15} $$

$$ \boldsymbol{T}_d = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \tag{16} $$

$$ \boldsymbol{T}_\theta = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \tag{17} $$

$$ \boldsymbol{T}_z = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \tag{18} $$

where, ${}^f\boldsymbol{S}_a$ is a permutation matrix to change an order of variables from an arm-based order to a function-based order. $\boldsymbol{I}_n$, an th order unit matrix, corresponds to virtual sum motor coordinates. $\boldsymbol{T}_d$ denotes a function matrix in coordinates while $\boldsymbol{T}_\theta$ and $\boldsymbol{T}_z$ denote that in and coordinates. The first row of $\boldsymbol{T}_d$, $\boldsymbol{T}_\theta$ and $\boldsymbol{T}_z$ are to derive sum of three manipulators' responses. The sum modes are named Mode 1. The second row and the third row are to derive the difference value of the manipulator A and others. The difference values correspond to Mode 2 and Mode 3.

${}^a\boldsymbol{T}_v$ in this study is shown as follows:

$$ {}^a\boldsymbol{T}_v = \begin{bmatrix} {}^a\boldsymbol{T}_{vA} & & \\ & {}^a\boldsymbol{T}_{vB} & \\ & & {}^a\boldsymbol{T}_{vC} \end{bmatrix} \tag{19} $$

$$ {}^a\boldsymbol{T}_{vA} = \begin{bmatrix} \boldsymbol{I}_3 & \\ & \boldsymbol{J}_A \end{bmatrix} \tag{20} $$

$$ {}^a\boldsymbol{T}_{vB} = \begin{bmatrix} \boldsymbol{I}_3 & \\ & \boldsymbol{J}_B \end{bmatrix} \tag{21} $$

$$ {}^a\boldsymbol{T}_{vC} = \begin{bmatrix} \boldsymbol{I}_3 & \\ & \boldsymbol{J}_C \end{bmatrix} \tag{22} $$

Here, $\boldsymbol{J}_A$, $\boldsymbol{J}_B$ and $\boldsymbol{J}_C$ denote Jacobian matrices for arm A, B and C, respectively.

${}^v\boldsymbol{T}_r$ in this study is shown as follows:

$$ {}^v\boldsymbol{T}_r = \begin{bmatrix} {}^v\boldsymbol{T}_{rA} & & \\ & {}^v\boldsymbol{T}_{rB} & \\ & & {}^v\boldsymbol{T}_{rC} \end{bmatrix} \tag{23} $$

$$ \begin{aligned} {}^v\boldsymbol{T}_{rA} &= {}^v\boldsymbol{T}_{rB} = {}^v\boldsymbol{T}_{rC} \\ &= {}^v\boldsymbol{S}_r \begin{bmatrix} \boldsymbol{H}_2 & & \\ & \boldsymbol{H}_2 & \\ & & \boldsymbol{H}_2 \end{bmatrix} \end{aligned} \tag{24} $$

$$ {}^v\boldsymbol{S}_r = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{25} $$

$$ \boldsymbol{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{26} $$

where, ${}^v\boldsymbol{S}_r$ is a permutation matrix to change an order of variables from real motors to virtual motors. $\boldsymbol{H}_2$ is a second-order Hadamard matrix.

Block diagrams of function-based controllers are shown in Fig. 8. Each function consists of a simple position/force controller.

V. Experiment

Experimental results are shown in this section. Table II shows control gains in the experiment. Figs. 9 and 10 show responses in coordinates and in coordinates, respectively. When the operator maneuvered the manipulator A in Step 1, all three manipulators moved only in grasping mode and accomplished open-close motion. An object was grasped in

TABLE II

CONTROL PARAMETERS

Position gain	K_p	600.0
Velocity gain	K_v	70.0
Force gain	K_f	8.0
Cutoff-frequency of DOB	G_{dis}	30.0
Cutoff-frequency of RFOB	G_f	15.0

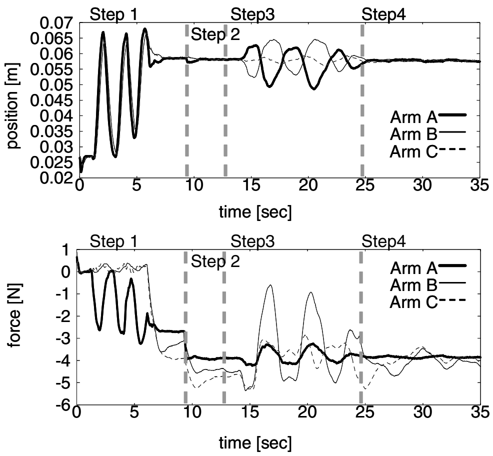

Fig. 9. Responses in d coordinate

Step 2 after the operator inserted it. The object was tilted in the pitching mode in Step 3 when the operator applied force in the direction. On the other hand, the object went up and down in Step 4 when the operator applied force in the same direction.

External force affected in all directions since the operator did not accurately maneuver. The object, however, moved only in the mode of spring coupling functions as shown from the force responses in Figs. 9 and 10. The direction of free motion was changed by modifying the combination of functions. Force responses in Fig. 9 show that grasping motion was retained then. Interaction between each mode was small due to acceleration control based on DOB.

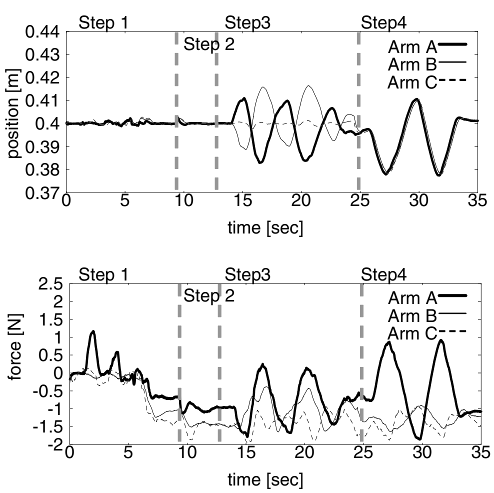

Fig. 10. Responses in z coordinate

VI. CONCLUSION

This study expanded the framework of function-based controller design to be more general. The expanded form deals with three-dimensional robots. A new problem of interactions among function-based systems occurs after the expansion. Disturbance observer is applied on each actuator to eliminate the interactions. The simplicity and explicitness of function-based controller design carry on despite the expansion since function-based systems are decoupled with disturbance observer.

ACKNOWLEDGMENT

This work is supported in part by NTT DoCoMo, Inc.

REFERENCES

[1] R. A. Brooks: "A Robust Layered Control System For A Mobile Robot", IEEE J. R & A, Vol. RA-2, No. 1, pp. 14–23, (1986)

[2] M. C. L. Sabatucci and A. Chella: "A Possible Approach to the Development of Robotic Multi-Agent Systems", Proc. IEEE/WIC Int. Conf. Intelligent Agent Technology, pp. 539–544, (2003)

[3] T. Ueyama, T. Fukuda, F. Arai, Y. Katou, S. Matsumura and T. Uesugi, "A Study on Dynamically Reconfigurable Robotic Systems (10th Report, Distributed Control Structure for Organization using an Evaluation of Network Energy for Group Structure of Cebot)", J. JSME, Part C, Vol. 58, No. 549, pp. 132–139, (1992) (in Japanese)

[4] H. Asama, M. K. Habib, I. Endo, K. Ozaki, A. Matsumoto and Y. Ishida, "Functional Distribution among Multiple Mobile Robots in An Autonomous and Decentralized Robot System", Proc. IEEE Int. Conf. R & A, pp. 1921–1926, (1991)

[5] Y. Fujimoto, T. Sekiguchi, "Fault-Tolerant Configuration of Distributed Discrete Controllers", IEEE Trans. on Industrial Electronics, Vol. 50, No. 1, pp. 86–93, (2003)

[6] J. E. Hernandez, A. G. Loukianov, B. Castillo-Toledo, V. I. Utkin, "Observer Based Decomposition Control of Linear Delayed Systems", Proc. IEEE Int. Conf. Decision and Control, pp. 1867–1872, (2001)

[7] S. Arimoto, P. T. A. Nguyen, "Principle of Superposition for Realizing Dexterous Pinching Motions of a Pair of Robot Fingers with Soft-tips," IEICE Trans. Fundamentals, Vol. E84-A, No. 1, pp. 39–47, 2001

[8] M. Okada, K. Tatani, Y. Nakamura, "Polynomial Design of the Nonlinear Dynamics for the Brain-Like Information Processing of Whole Body Motion," Proc. of IEEE Int. Conf. on R & A, pp. 1410–1415, 2002

[9] D. Lee and P. Y. Li, "Passive bilateral feedforward control of linear dynamically similar teleoperated manipulators" , IEEE Trans. Robotics & Automation, vol. 19, No. 3, pp. 443–456, 2003.

[10] T. Tsuji, K. Ohnishi, "A Controller Design Method of Decentralized Control System", IEEJ Int. Power Electronics Conf. (IPEC-NIIGATA), 2005.

[11] C. D. Onal, A. Sabanovic, "Bilateral Control with a Reflex Mechanism on the Slave Side", Proc. of the 31st Annual Conf. of the IEEE Industrial Electronics Society (IECON2005), pp. 195–200, 2005

[12] K. Ohnishi, M. Shibata, T. Murakami, "Motion Control for Advanced Mechatronics", IEEE/ASME Trans. Mechatronics, vol. 1, No. 1, pp. 56–67, 1996

[13] T. Tsuji, K. Natori, K. Ohnishi: "A Controller Design Method of Bilateral Control System," EPE-PEMC'04, Vol. 4, pp. 123–128, 2004.

[14] N. Hayashida, T. Yakoh, T. Murakami, K. Ohnishi, "A Friction Compensation in Twin Drive System", Proc. 6th Int. Workshop on Advanced Motion Control, pp.187–192, 2000

[15] T. Murakami, R. Nakamura, F. Yu, K. Ohnishi, "Force Sensorless Compliant Control Based on Reaction Force Estimation Observer in Multi-Degrees-of-Freedom Robot", Journal of RSJ vol. 11, No. 5, pp. 765–768, 1993. (in Japanese)

[16] T. Kageyama, K. Ohnishi, "An architecture of decentralized control for multi-degrees of freedom parallel manipulator", Proc. 7th Int. Workshop on Advanced Motion Control, pp. 74–79, 2002