Proceedings of the 2003 IEEE
International Conference on Robotics & Automation
Taipei, Taiwan, September 14-19, 2003

An Easily Reconfigurable Robotic Assembly System

Yusuke MAEDA[1] Haruka KIKUCHI[2] Hidemitsu IZAWA[1]
Hiroki OGAWA[3] Masao SUGI[1] *and* Tamio ARAI[1]

[1]Department of Precision Engineering, School of Engineering,
The University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, JAPAN
maeda@prince.pe.u-tokyo.ac.jp

[2]Mobile Agent Lab, Multimedia Labs, NTT DoCoMo, Inc.

[3]Dept. of Mechano-Informatics, School of Information Science and Technology,
The University of Tokyo

Abstract

In this paper, a flexible robotic assembly system with decentralized architecture is presented. The system is designed to achieve high reconfigurability so that it can adapt to changes in manufacturing environment; a new robot can be easily installed to the system and execute assembly tasks immediately, together with other devices. For easier reconfiguration, a semi-automated calibration method for positions of newly installed robots is integrated into the system.

Our implementation of the assembly system consists of conventional manipulators and a belt conveyor. In the experiment, a new mobile manipulator was installed successfully and performed an assembly operation jointly with other existing manipulators.

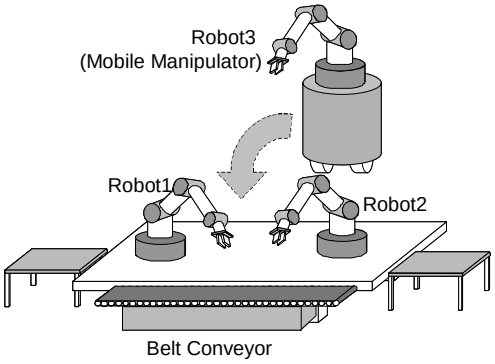


Figure 1: Plug-in of a New Robot

1 Introduction

In recent years, reconfigurability of manufacturing systems has been in great demand [1, 2]. Changes of system configuration at low cost will help the systems cope with predictable and unpredictable fluctuations in production environment. For example, a manufacturing system can adapt to the increase of production quantity if an additional device can be easily plugged into the system (Figure 1).

In regard to assembly systems, many researchers proposed decentralized control architectures to achieve flexibility, including reconfigurability [3–8]. Such decentralized software architectures are very helpful to achieve reconfigurability, but that covers only a part of the purpose. In real systems, physical problems of reconfiguration, such as conflict of workspaces, should be solved effectively. In order to achieve reconfigurability in real assembly systems, some researchers developed special homogeneous hardware [9–11], which is generally expensive.

The authors proposed a decentralized control architecture for robotic assembly cells that consists of heterogeneous conventional devices [12–14]. The architecture has functions that support easy reconfiguration; information renewals associated with physical reconfiguration are automated, and conflict of workspaces of the devices at reconfiguration is solved decentralizedly by negotiation. Collectively, we call the functions “Plug & Produce” [13–15].

As another physical problem, calibration of the position of a newly installed robot is required so that we can install the robot at an arbitrary position. Therefore, easy calibration is highly desirable. In our previous implementation [12], however, the calibration is performed manually; that is, a human operator has to move the gripper of a newly installed robot to point at guiding marks to localize the robot. The necessity of manual calibration may spoil the

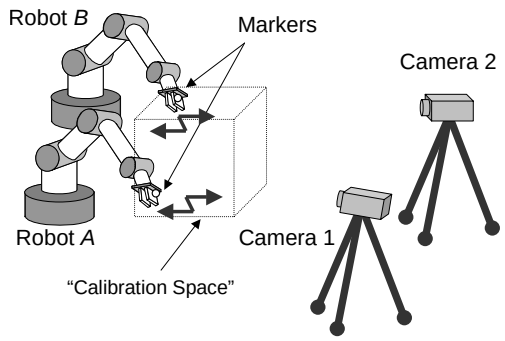


Figure 2: Calibration for Plug & Produce

system's reconfigurability.

In this paper, we present a new version of our assembly system. It has semi-automated calibration for newly installed robots, which enhances the reconfigurability of the system significantly. The management mechanism of positional information resulting from installation/removal of robots is also described. Experimental results of the system's behavior in installation of a new robot and an assembly task are shown.

2 Calibration for Plug & Produce

Calibration of the position of a newly installed robot is required for hand-over of parts and collision avoidance between the new robot and existing ones. We developed a semi-automated calibration method for Plug & Produce [16]. In the method, relative position/orientation between a newly installed robot and an existing robot is easily obtained. Here we describe the method briefly.

Our calibration method is based on the reconstruction of 3D coordinates of objects from camera images by the Direct Linear Transformation (DLT) method [17]. We obtain a homogeneous transformation matrix that represents positional relationship between the base frame of a newly installed robot and that of an existing one, by observing markers attached to the end-effector of each robot (Figure 2).

The procedure for the calibration of positional relationship between robot *A* and *B* is as follows:

1. A human operator places two cameras so that they can observe the markers on robots *A* and *B*.
2. Robot *A* moves autonomously in a limited space and the cameras observe the motion of its marker. Pairs of the image coordinates of the marker, $\boldsymbol{u}_{Ai} = [u_{1Ai}, v_{1Ai}, u_{2Ai}, v_{2Ai}]^T$, and 3D coordinates of the marker in *A*'s base frame, $\boldsymbol{x}_{Ai} = [x_{Ai}, y_{Ai}, z_{Ai}]^T$, are sampled as control points $(i = 1, \ldots, n_A)$.
3. Then robot *B* moves autonomously in the limited space and the cameras also observe the motion of its marker. Pairs of the image coordinates of the marker, $\boldsymbol{u}_{Bj} = [u_{1Bj}, v_{1Bj}, u_{2Bj}, v_{2Bj}]^T$, and 3D coordinates of the marker in *B*'s base frame, $\boldsymbol{x}_{Bj} = [x_{Bj}, y_{Bj}, z_{Bj}]^T$, are sampled as control points $(j = 1, \ldots, n_B)$.
4. A mutual positional relationship between the robots is calculated from $(\boldsymbol{x}_{Ai}, \boldsymbol{u}_{Ai})$ and $(\boldsymbol{x}_{Bj}, \boldsymbol{u}_{Bj})$, as a homogeneous transformation matrix $({}^A\boldsymbol{T}_B)$.

The features of the calibration method are as follows:

- No substantial modification to assembly devices are required. All we need is markers on the robots and external cameras.
- The calibration procedure can be mostly automated. Human operators only have to place two cameras where they can observe the markers.
- Positions of cameras can be unknown. Thus we can place cameras instantaneously.
- Transformation between the base frames of two manipulators is obtained. This is suitable for decentralized systems composed by autonomous agents.

In this method, calibration is made in a limited space (we call it "calibration space"). Note that reliable accuracy is achieved only around the calibration space, where inter-robot operation (e.g., hand-over of parts) should be performed.

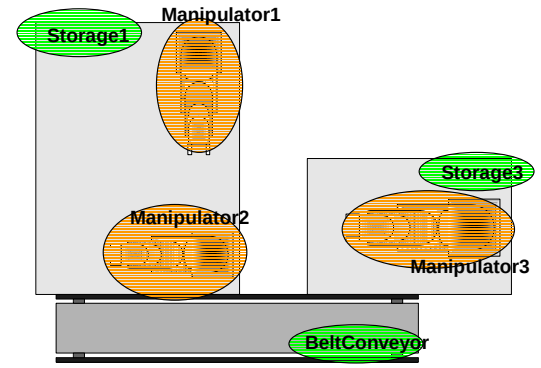
In our current implementation, the accuracy in the calibration space is about 0.5 [mm]. Details of the calibration including remarks on accuracy improvement are found in [16].

3 Management of Positional Information

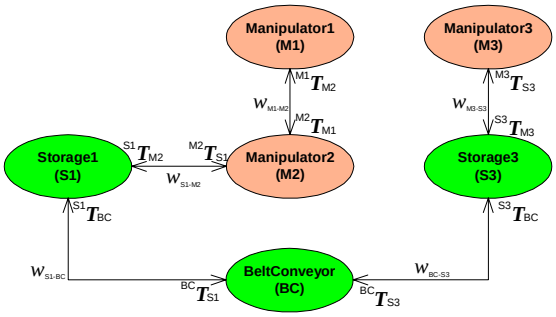
3.1 Graph Representation of Positional Relationships

Using the calibration method presented in the previous section, we can obtain information of positional relationships between robots. Our assembly system has decentralized architecture, therefore the positional information should be managed in a decentralized manner; therefore, we do not use global coordinates in a world coordinate frame, but just keep relative positional information between devices.

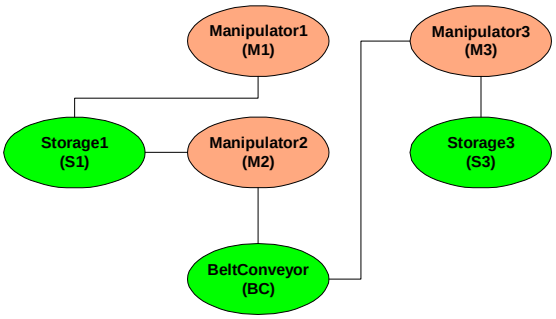
Let us consider an assembly cell like Figure 3(a). In this case, the positional information of assembly devices in the cell is represented as a "calibration graph," shown in Figure 3(b). Each node of the graph corresponds to each assembly device. Note that workbenches of the robots are



(a) Assembly Cell (Example)



(b) Calibration Graph



(c) Adjacency Graph

Figure 3: Graph Representation of Positional Relationships between Assembly Devices

also represented as "Storage" nodes. Only when the positional relationship between device X and Y is directly calibrated or defined, corresponding nodes are connected by an arc. In the arc, a homogeneous transformation matrix $({}^{X}\boldsymbol{T}_{Y})$ that represents the relative position/orientation between the devices is stored. By assigning a proper "cost" value to each arc according to the calibration accuracy, we can employ Dijkstra's algorithm to find the "shortest" path between two devices that are not directly calibrated. The path gives relative position/orientation between the devices. An advantage of not using global coordinates is that the system does not have to resolve positional inconsistency explicitly. That is, the following equation does not usually hold in our system because of the calibration error:

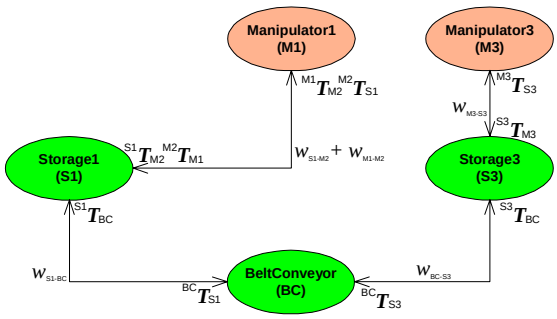


Figure 4: Updated Calibration Graph (Manipulator2 removed)

$$ {}^{X}\boldsymbol{T}_{Z} = {}^{X}\boldsymbol{T}_{Y}\,{}^{Y}\boldsymbol{T}_{Z}; \tag{1} $$

therefore if using global coordinates, we have to determine each global position/orientation of the assembly devices as a compromise solution of inconsistent equations.

In addition to the calibration graph, the system has another graph, called "adjacency graph" (Figure 3(c)). In the adjacency graph, an arc connects two device nodes when the devices are neighbors (that is, they can hand over a part directly). Arcs in the adjacency graph do not connect two manipulators directly. This is because a manipulator does not hand a part over to another manipulator directly, but places it on a shared domain temporarily and then another manipulator picks it up. The adjacency graph is required to determine transfer routes of parts between assembly devices.

When a new robot is installed in the system, both the calibration graph and the adjacency graph must be updated according to the calibration result. Similarly, when a robot is removed from the system, both graphs must be updated, too. In the case of removal, however, the calibration graph must be kept connected. For example, when we remove "Manipulator2" from the assembly cell in Figure 3, the calibration graph is altered as Figure 4.

3.2 Workspace Allocation

After the positional information is obtained, proper workspace (re-)allocation is necessary to avoid physical conflict between assembly devices and to hand over parts and products. In our system, the workspaces of assembly devices are classified as follows [13]:

Shared Domain. A domain where the neighboring devices can enter. This domain is used for hand-over of parts or products.

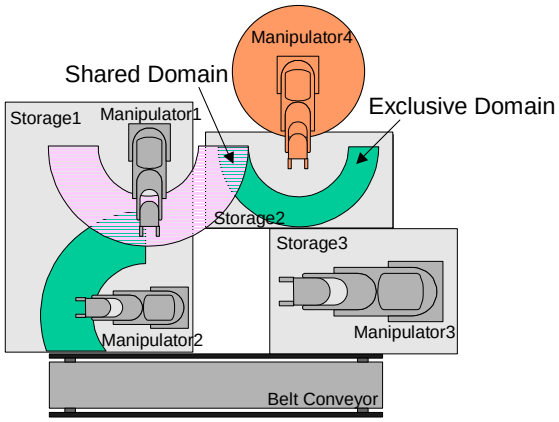


Figure 5: Shared/Exclusive Domains

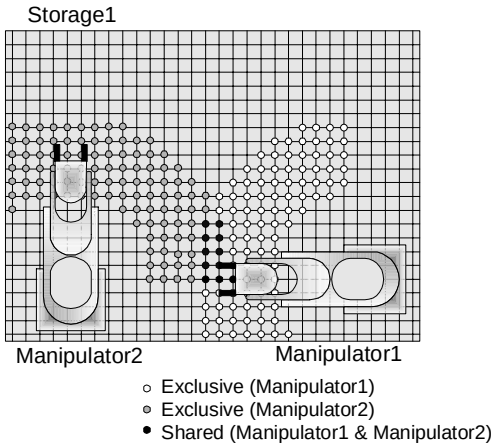


Figure 6: Grid Representation of Shared/Exclusive Domains

Exclusive Domain. A domain where other devices cannot enter. This domain is used for assembly and keeping parts or products.

Shared and exclusive domains are defined as in Figure 5.

In our implementation, all the workspaces are represented as grids, and each grid point is labeled as shared, exclusive, or neither (Figure 6). When a robot is installed or removed, the information stored in the grid points must be updated according to the calibration result and movable areas of the robots.

3.3 Procedure of Robot Installation / Removal

Now we describe how the positional information mentioned above is updated when a new robot is installed. The procedure of robot installation is as follows:

1. The new robot announces its rough position, which is given by a human operator, to all the devices and tells its possible neighbor devices to undertake no assembly operations for collision avoidance.
2. The new robot waits for the completion of all the assembly operations of its possible neighbors.
3. The calibration procedure described in Section 2 is performed. The procedure should be repeated for all the possible neighbors.
4. The calibration graph of the system is updated according to the calibration result.
5. Workspace information stored in the grid points is updated according to the calibration graph and movable areas of the robots.
6. The adjacency graph is updated according to the change of workspace information.
7. Now, the robot installation is completed. The new robot tells the neighbor devices to resume undertaking assembly operations.

The procedure for removing a robot from the system is similar:

1. The robot to be removed stops to undertake new assembly operations and completes all of its assigned operations.
2. The robot hands over parts in its exclusive domain, if any, to other robots.
3. Workspace information stored in the grid points is updated as a consequence of the removal of the robot.
4. The adjacency graph of the system is updated.
5. The calibration graph is updated.
6. Now, the removal of the robot is completed.

4 Experiment of Plug & Produce

4.1 Implemented Assembly Cell

Our implementation of the assembly system consists of three fixed manipulators, one mobile manipulator, and one belt conveyor (Figure 7). An LED is attached on the end-effector of each manipulator, as a marker for calibration. Two CCD cameras are also used for calibration. Each device is controlled by a Linux PC (Figure 8). All the assembly devices and storage are implemented as software agents (“holons” [13]). The agents are programmed in Java for high interoperability, while they were written in C++ in our previous implementation [12]. We use Java Communications API and JNI (Java Native Interface) for controlling assembly devices. All the agents can communicate with each other through the Ethernet.

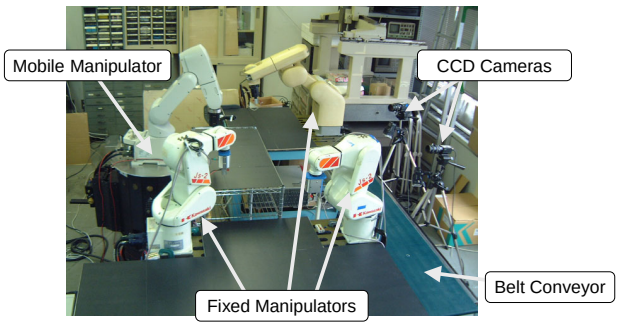


Figure 7: Implemented Assembly Cell

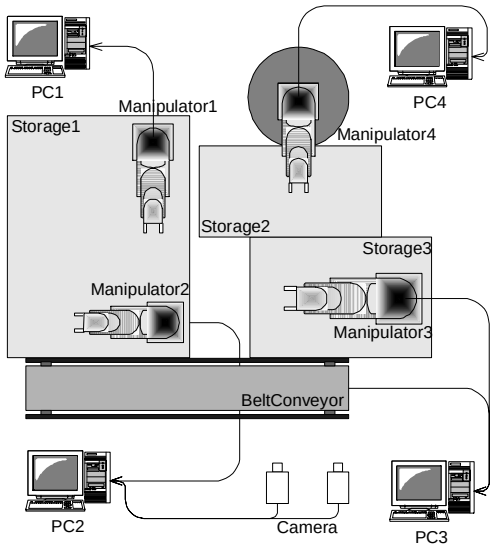
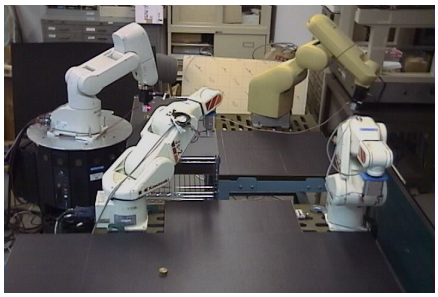


Figure 8: System Configuration

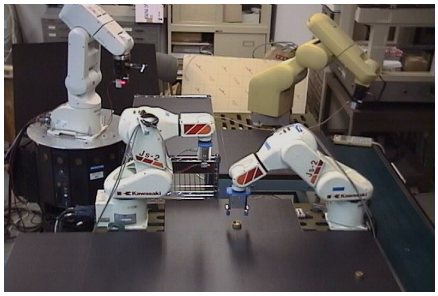
4.2 Experiment

Installation of a mobile manipulator to the assembly cell was performed (Figure 9). At first, the mobile manipulator ("Manipulator4") was placed next to a fixed manipulator ("Manipulator1"). Then each manipulator moved its gripper to eight points in turn ($n_A = n_B = 8$). These points formed a bounding box, whose location was determined based on the rough position of the newly installed Manipulator4. After the calibration procedure, a simple assembly task was requested. The task was just to collect part 'A' and 'B' and insert 'A' into 'B.' Necessary operations to complete the task were assigned to each device by negotiation and executed in order.

In the experiment, the new manipulator was successfully plugged into the system and participated in the assembly task. Figure 10 is a Gantt chart for the experiment. At first, calibration between Manipulator1 and Manipulator4 was executed. In our current implementation, the setting



(a) A Scene in Calibration



(b) A Scene in Assembly

Figure 9: Experiment of Plug & Produce

of tracking windows to observe markers for calibration is not yet automated, therefore a human operator had to do it (60 [s]–105 [s]). Calibration was completed at 125 [s], and then an operation of moving part 'B' was immediately assigned to Manipulator4, the newly installed robot. At 380 [s], the product was finally assembled.

This experimental result shows that "Plug & Produce" functionality works fine for robotic assembly systems.

5 Conclusion

In this paper, we presented a robotic assembly system with high reconfigurability. The "Plug & Produce" functionality of the system including semi-automated calibration of positions of newly installed robots enables flexible reconfiguration of the system. In the experiment, we could easily install a new manipulator to the system to participate in assembly tasks.

The system will not be optimal with regard to its reconfigurability until it has effective task allocation. Currently, we are improving our existing task allocation algorithm based on inter-agent negotiation.

Acknowledgments

This research is supported by IMS/HMS (Intelligent Manufacturing Systems/Holonic Manufacturing Systems) international research program. The authors would like to

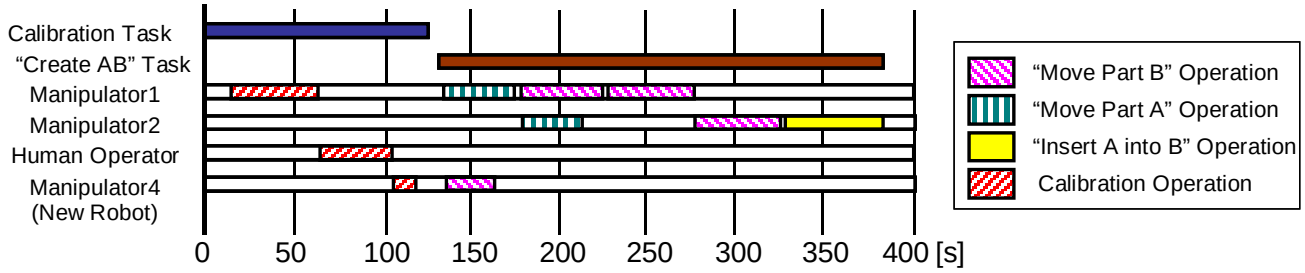


Figure 10: Gantt Chart for Plug-in and Assembly

thank Toshibumi Seki at the Toshiba Corporation for his advice on inter-agent communication, and Jay Bradley at the University of Edinburgh for his helpful comments.

References

[1] Y. Koren, U. Heisei, F. Jovane, T. Moriwaki, G. Pritschow, G. Ulsoy and H. Van Brussel: "Reconfigurable Manufacturing Systems," Annals of the CIRP, Vol. 48, No. 2, pp. 527–540, 1999.

[2] I.-M. Chen: "Rapid response manufacturing through a rapidly reconfigurable robotic workcell," Robotics and Computer Integrated Manufacturing, Vol. 17, No. 3, pp. 199–213, 2001.

[3] E. Oliveira: "Cooperative Multi-Agent System for an Assembly Robotics Cell," Robotics and Computer-Integrated Manufacturing, Vol. 11, No. 4, pp. 311–317, 1994.

[4] P. Valckenaers, H. Van Brussel, L. Bongaerts and F. Bonneville: "Programming, scheduling, and control of flexible assembly systems," Computers in Industry, Vol. 26, No. 3, pp. 209–218, 1995.

[5] A. A. Rizzi, J. Gowdy and R. L. Hollis: "Agile Assembly Architecture: An Agent Based Approach to Modular Precision Assembly Systems," Proc. of 1997 IEEE Int. Conf. on Robotics and Automation, pp. 1511–1516, 1997.

[6] J. S. Basran, E. M. Petriu and D. C. Petriu: "Flexible Agent-based Robotic Assembly Cell," Proc. of 1997 IEEE Int. Conf. on Robotics and Automation, pp. 3461–3466, 1997.

[7] J.-C. Fraile, C. J. J. Paredis, C.-H. Wang and P. K. Khosla: "Agent-Based Planning and Control of a Multi-Manipulator Assembly System," Proc. of 1999 IEEE Int. Conf. on Robotics and Automation, pp. 1219–1225, 1999.

[8] J.-L. Chirn and D. C. McFarlane: "A Component-Based Approach to the Holonic Control of a Robot Assembly Cell," Proc. of 2000 IEEE Int. Conf. on Robotics and Automation, pp. 3701–3706, 2000.

[9] K. Tamaki, M. Uno, T. Mita, K. Sugimoto, S. Sakaue and H. Onari: "A Restructurable Assembly Center Employing Mobile DD-Robots," Proc. of 24th Int. Symp. on Industrial Robots, pp. 119–126, 1993.

[10] S. Kondoh, Y. Umeda and H. Yoshikawa: "Development of Upgradable Cellular Machines for Environmentally Conscious Products," Annals of the CIRP, Vol. 47, No. 1, pp. 381–384, 1998.

[11] M. Hanai, T. Nakasai, H. Hibi, F. Kojima and Y. Fukuda: "New Autonomous Manufacturing System Adapted for Uncertainty in Market —APS: Adaptive Production System—," Proc. of the Second Int. Workshop on Intelligent Manufacturing Systems, pp. 15–22, 1999.

[12] T. Arai, Y. Aiyama, M. Sugi and J. Ota: "Holonic assembly system with Plug and Produce," Computers in Industry, Vol. 46, No. 3, pp. 289–299, 2001.

[13] M. Sugi, Y. Maeda, Y. Aiyama and T. Arai: "Holonic Robot System: A Flexible Assembly System with High Reconfigurability," Proc. of 2001 IEEE Int. Conf. on Robotics and Automation, pp. 799–805, 2001.

[14] M. Sugi, Y. Maeda, Y. Aiyama, T. Harada and T. Arai: "A Holonic Architecture for Easy Reconfiguration of Robotic Assembly Systems," IEEE Trans. on Robotics and Automation, to appear.

[15] T. Arai, Y. Aiyama and Y. Sasaki: "Holonic Storage: An Assembly and Storage Cell by Manipulation Using Environment," Proc. of 29th CIRP Int. Seminar on Manufacturing Systems, pp. 221–226, 1997.

[16] T. Arai, Y. Maeda, H. Kikuchi and M. Sugi: "Automated Calibration of Robot Coordinates for Reconfigurable Assembly Systems," Annals of the CIRP, Vol. 51, No. 1, pp. 5–8, 2002.

[17] R. Shapiro: "Direct Linear Transformation Method for Three-Dimensional Cinematography," Research Quarterly, Vol. 49, No. 2, pp. 197–205, 1978.