A LEARNING ROBOT TO ACQUIRE ASSEMBLY TASKS THROUGH OBSERVATION OF HUMAN DEMONSTRATION
視覚により人と同等な組み立て作業能力を獲得するロボット

by

Jun Takamatsu
高松 淳

A Master Thesis
修士論文

Submitted to
the Graduate School of
the University of Tokyo
on February 6, 2001
in Partial Fulfillment of the Requirements
for the Degree of Master of Science
in Information Science

Thesis Supervisor: Katsushi Ikeuchi　池内 克史
Professor of Information Science

ABSTRACT

The realization of robot learning ability that is the equal of its human counterpart is one of the dreams of robotics researchers. Although the realization of such ability in a general setting is quite difficult, this paper describes how, by limiting the scope of the robot's assembly tasks and proposes, we propose to design and develop such a learning robot. First, the robot uses computer vision to observe a human performance; next, the robot learns to understand the human performance by the use of symbolic representations called "task-models". Those task models are then converted into "sub-skills", group of operations executable by the robot. Through the execution of a sequence of sub-skills, the robot can robustly perform the same tasks performed by the human demonstrator. We have implemented such ability in our robot test-bed and demonstrated its effectiveness.

論文要旨

人間と同等の学習能力を持つロボットをつくることは、ロボット研究の夢であるが、その現実は困難を極める。この論文では、組み立て作業に話を限定し、人間と同様に視覚学習から、組み立て作業手法を習得するロボットを提案する。まず、人間の行った動作を観察し、それから作業モデルとよばれる抽象的な表現でそれを記述する。

しかる後に、作業モデルをサブスキルと呼ばれるロボットの手先で実行可能な命令列に変換する。ロボットはこのサブスキルの列を実行することで人間により行われた作業を再現する。実際にこのような能力をロボット上に実装し、その効果について検証した。

Acknowledgements

The author thanks the members of the Ikeuchi laboratory, especially the members of robotics group, for advising, helping my experimentation, and offering various software and hardware to be used in this work.

Contents

Chapter 1

Introduction

Several methods for trying a robot to acquire human hand actions have been proposed. *Teach pendant* and *tele-operation* are methods to do by giving a robot the path of a manipulator through easy means. In the methods, because a manipulator of a robot moves only along the given path, a manipulator needs to have a high degree of rigidity and excellent position control ability, and the environment must be fixed. So, for industrial use, those conditions are satisfied, and thus it is better selection for robot execution.

Path planning method has proposed as an enhancement of these methods. In the method, we give only the goal of the task; a robot then plans the manipulation path automatically. [17] proposed the notion of the *configuration space*(c-space). Using c-space, we can plan the manipulation path without depending on shapes of objects. But it is very difficult to treat c-space; in reality, we can plan the path in planar case only. Now, however, a method for overcoming this problem by using *randomized algorithm* has been proposed in [16].

Generally speaking, the path planning method can be used only in an ideal situation as mentioned above. But real world situations change dynamically and do not continue to maintain ideal situations. Because assembly tasks, in particular, need precise position control, that is the serious problem. In this paper, we propose the method to solve this problem.

In assembly tasks, the transition between contact relations is very important. To achieve good transitions, we need to solve the following subproblems:

1. How do we obtain the possible transitions of contact relations?
2. How do we select the better transitions?
3. How do we move an object for realizing the aimed transitions?
4. How do we recover execution errors?

In [10], a method for solving the first problem, obtaining the possible transitions,is proposed. The method solves efficiently using the notion of *limit angle*. But it is difficult to compute a limit angle.
In [11], a method for the solving the problem of selecting the better transitions is proposed. The method solves by subtracting a *motion degree of freedom*(DOF). A recent paper[12] proposed a method to select the better transition between contact relations with the same DOF.
In [14], a method for solving the problem of realizing the aimed translations is proposed. The method formulates an object motion maintaining or detaching contact elements composed of the contact relation. Though the solution region of the formulation represents an possible movement, this method is proposed selecting more robust movement, too. For using the c-space analysis, this method applies in planar cases only. [15] [16] proposed in 3D cases using a randomized algorithm.
Several methods to overcome the fourth problem, recovering execution errors, by using force control and force feedback have been proposed. Those methods are roughly divided into *impedance control* and *force/position hybrid control*. Impedance control [34] [35] [36] enables a robot to move flexibly by controlling the rigidity of a manipulator, although, in pure position control, it is infinite. Force/position hybrid control [37] is the method to apply to position or force control to each DOF. Though the two methods enable a robot to perform assembly tasks robustly, it is difficult to implement the robot program to perform aimed tasks, and an excellent programmer and a long develop span are needed.
A high performance system for acquiring human hand actions needs to integrate those methods. In this paper, we would like to propose such an integration method.

1.1 Acquiring knowledge

The successful design and development of robots with learning abilities equivalent to those of humans is the dream of robot researchers. Because excellent programming skills are required to provide the robots with motion skills, the robots can be regarded as being only tools, tools that are difficult to use very well. How then, can robots acquire skills that are the equivalent of human skills? Unlike humans, robots do not have an innate ability for acquiring knowledge. It is true that humans have only the most elementary skills, e.g., sucking and swallowing; however, they acquire more complex skills by receiving external impulses. Based on this fact, *the assembly plan from observation* (APO) system was proposed[1] [2] [3]. The goal of the APO system is to design and develop robots that can acquire the ability to perform assembly tasks., an ability that equals the ability of humans to perform those same tasks.

We believe that humans obtain various knowledge of actions mainly from observation. Using symbolic representation, we recognize and memorize observation data using symbolic representation. We perform the action by converting symbolic representation to real motion. By repeating the actions, we improve our ability.

In the APO system, the robot can obtain various pieces of information from observation. The robot can recognize and memorize by using essential operations for assembly tasks referred as to *sub-skill.* The sub-skill is a symbolic representation in assembly tasks. The robot performs the action by a proper sub-skill execution with proper parameters. Though in the first attempt of the assembly task, the robot may decide the proper parameters, it may be able to make further progress by controlling the parameters.

The previous APO system could not completely realize the mimicking of the human learning system. In this paper, we propose a method to realize better mimicking, although it does not provide the robot with the ability for making progress.

1.2 Thesis outline

In Chapter 2, we describe how to recognize and memorize assembly tasks. In this chapter, we present essential sub-skills. In Chapter 3, we describe the robot vision system. Our robot vision system is more robust to observation errors using contact relations and their transitions. In Chapter 4, we describe the robot itself. In Chapter 5, we describe the implementation of sub-skills. In Chapter 6, we describes the experiment and the result. Finally, in Chapter 7, we present our conclusions about our work

Chapter 2

Recognizing assembly tasks

The goal of assembly tasks is to achieve transitions of contact relations. Because infinite contact relations and their transitions exist, in order to recognize assembly tasks, we need to analyze contact relations and their transitions. First, we introduce *the screw theory* and six types of degrees of freedom(DOF) for analyzing contact relations. Next, we propose the analysis of those transitions from possible transitions of six types of DOFs. In this paper, we assume that an object is polyhedral and rigid, and that only one object grasped by a hand or a manipulator (referred as to a *grasping object*) is allowed to move.

2.1 Configuration space

Analysis of contact relations is necessary for recognizing assembly tasks. Because a possible motion is decided by, and closely related to, a contact relation, we analyze the contact relation from possible motion in this situation. The object configuration has six DOFs (three DOFs in translation and three DOFs in rotation). The coordinator with six axes that correspond to six DOFs (referred as c-space)[17] [18], can be divided into two types of regions. One is the possible object configuration region and the other is the impossible (for example, when one object sticks to another). The region composed of an impossible object configuration is called *c-obstacle* and the surface of the region is called *c-surface*. For an example, but in the planar motion (two DOFs in translation and one DOF in rotation) case, in a c-space, a possible configuration of a rectangular object as shown to the left in Figure 2.1 is shown to the right in Figure 2.1.

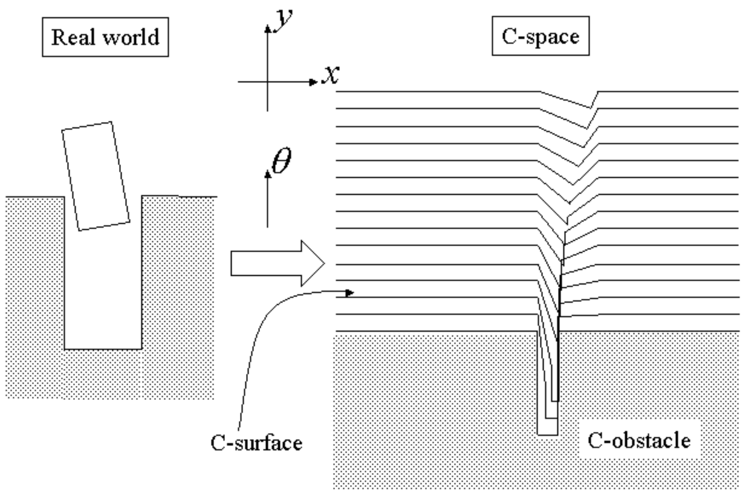


Figure 2.1: C-space

Using c-space, we can analyze a possible motion without regard to the shape of objects. Though computing c-surface and c-obstacle is very difficult (NP-complete problem), the use of the screw theory makes it very easy to compute local c-surface and c-obstacle representing impossible infinitesimal motion.

2.2 Screw theory

The screw theory is a convenient concept for representing three dimensional rigid body displacement[19]. Any rigid body displacement can be accomplished by a rotation about a unique axis and a translation along the same axis. The combined motion is called a screw displacement or twist. The axis is referred to as the screw axis, and the ratio of the translation to the rotation is designated as the pitch of the screw. The amount of rotation about the screw axis is called the amplitude of the screw.

A screw S is represented mathematically by two 3D vectors, $[S_0, S_1]$. S_0 is the direction of the screw axis and $S_1 = P \times S_0 + pS_0$. The vector $P \times S_0$ is the

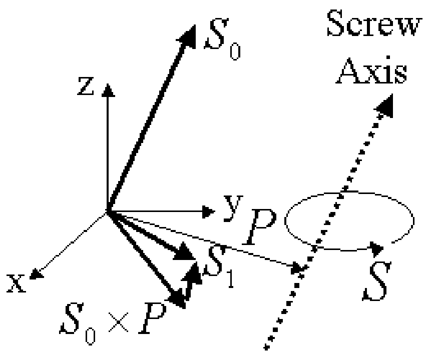


Figure 2.2: Screw representation

moment of the screw axis about the origin, P is a vector to the screw axis from the origin and the scalar p is the pitch of the screw. For pure rotations, the pitch $p = 0$, and the screw S will be $[S_0, P \times S_0]$. For pure translations, the pitch p is infinity, so the screw S will be [0, S_0].

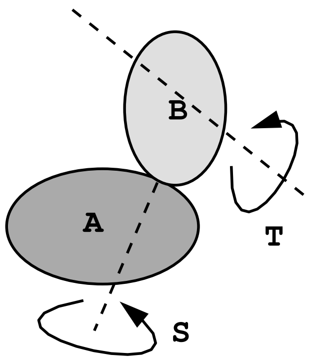


Figure 2.3: one object is contact with another

Consider two bodies in contact as shown in Figure 2.3. Let screw S=$(s_1, s_2, s_3, s_4, s_5, s_6)$ represent the line of contact in 3D space and screw T=$(t_1, t_2, t_3, t_4, t_5, t_6)$ represent the displacement of one object with respect to the other. Any screw T that can cause the sliding of object B on object A is called a reciprocal screw and is related to S by the equation (2.1).

$$s_1t_4 + s_2t_5 + s_3t_6 + s_4t_1 + s_5t_2 + s_6t_3 = 0 \tag{2.1}$$

Any screw T that can cause the detaching of object B from object A is called a repelling screw and is related to S by the inequality (2.2).

$$s_1t_4 + s_2t_5 + s_3t_6 + s_4t_1 + s_5t_2 + s_6t_3 0 \tag{2.2}$$

Thus, the relation that defines the feasible set of legal motions which do not violate the contact(sliding and repelling) can be written as the inequality (2.3).

$$s_1t_4 + s_2t_5 + s_3t_6 + s_4t_1 + s_5t_2 + s_6t_3 \geq 0 \tag{2.3}$$

2.3 Possible motion in screw theory

A contact relation is represented as the combination of contacts between vertex, edge, and face as shown in Figure 2.4. First, we would like to mention the method for representing an infinitesimal possible motion by the inequality in a single contact case using screw theory. Next, we would like to mention the method in a multiple contact case.

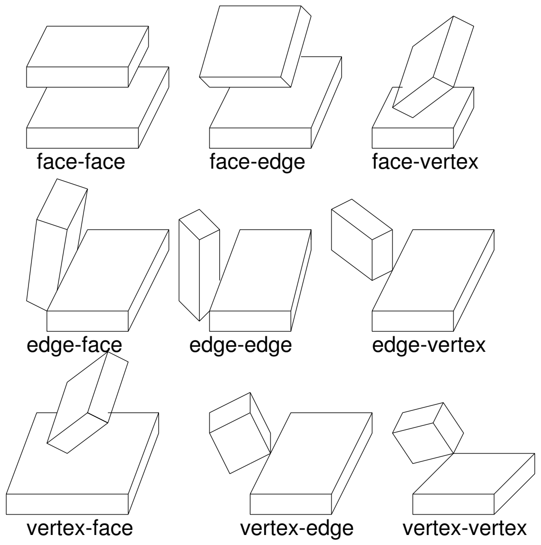


Figure 2.4: Nine kinds of contacts

2.3.1 Single contact

Vertex-face contact

In the vertex-face contact case as shown in Figure 2.5, the inequality (2.4) represents the possible infinitesimal motion of a grasping object in the screw representation, where $T_0 = (t_1, t_2, t_3)$ and $T_1 = (t_4, t_5, t_6)$.

$$n \cdot T_1 + (p \times n) \cdot T_0 \geq 0 \tag{2.4}$$

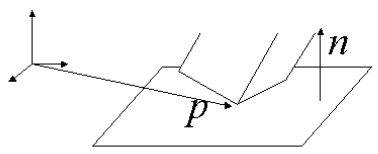


Figure 2.5: Vertex-face contact

Face-vertex contact

In the face-vertex contact case as shown in Figure 2.6, the inequality (2.5) represents the possible infinitesimal motion of a grasping object in the screw representation.

$$-n \cdot T_1 - (p \times n) \cdot T_0 \geq 0 \tag{2.5}$$

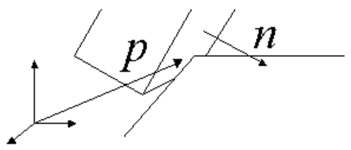


Figure 2.6: Face-vertex contact

Edge-edge contact

In the non-parallel edge-edge contact case as shown in Figure 2.7, the inequality (2.6) represents the possible infinitesimal motion of a grasping object in the screw representation, where $n = \pm\frac{e_1 \times e_2}{|e_1 \times e_2|}$ and the direction of the vector n is out to a grasping object.

$$n \cdot T_1 + (p \times n) \cdot T_0 \geq 0 \tag{2.6}$$

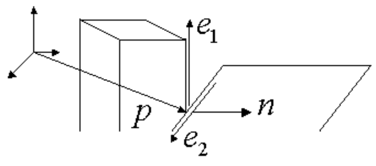


Figure 2.7: Non-parallel edge-edge contact

In the parallel edge-edge contact case as shown in Figure 2.8, the possible infinitesimal motion cannot be represented by the method mentioned above. But with regard to the parallel edge-edge contact as some vertex-face and face-vertex contacts, it can be represented. From this point, we will assume that an edge-edge contact means a non-parallel edge-edge contact.

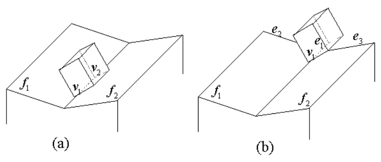


Figure 2.8: Parallel edge-edge contact

Another contact

Consider the edge-face contact case as shown to the left in Figure 2.9. The edge-face contact can be represented as the combination of finite vertex-face contacts, because the possible motion in the edge-face contact case as shown to the left in Figure 2.9 is equal to in the two vertex-face contacts case as shown to the right in Figure 2.9.

Other contacts can be represented as the combination of finite vertex-face, face-vertex, and edge-edge contact. So all contact relations can be represented in the screw representation using the method mentioned later. From now on, we will

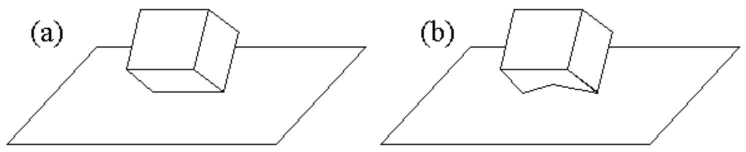


Figure 2.9: Face-face contact

assume that a contact relation is represented as the combination of three finite kinds of contacts.

2.3.2 Multiple contact

In a multiple contact case, a possible infinitesimal motion can be represented as a simultaneous inequalities (2.7), where n is the number of contacts and each equation represents one of the combinations of contacts.

$$\begin{pmatrix} a_{11} & \cdots & a_{16} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{n6} \end{pmatrix} \begin{pmatrix} t_1 \\ \vdots \\ t_6 \end{pmatrix} \geq \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2.7}$$

2.4 Motion DOFs

The previous APO system[1] divides a translation DOF into three types as follows: (Show in Figure 2.10)

Maintaining DOF A maintaining DOF is the DOF in which an object translates maintaining the contact relation.

Detaching DOF A detaching DOF is the DOF in which an object translates detaching the contact relation.

Constraining DOF A constraining DOF is the DOF in which an object cannot translate.

This paper treats the motion of an object not only as a translation, but also as a rotation. So we define three types of rotation DOFs as follows:

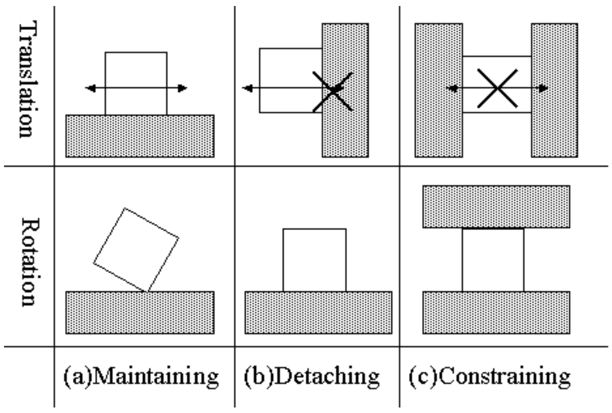


Figure 2.10: Three types of motion DOFs: maintaining, detaching, and constraining DOFs

Maintaining DOF A maintaining DOF is the DOF of the direction of a rotation axis[1] around which an object rotates maintaining contact relations.

Detaching DOF A detaching DOF is the DOF of the direction of a rotation axis around which an object rotates detaching contact relations.

Constraining DOF A constraining DOF is the DOF of the direction of a rotation axis around which an object cannot rotate.

2.5 Analyzing contact relations

In the previous section, we defined six types of motion DOFs for analyzing contact relations. Now, we would like to propose the method to compute those from a contact relation. First, we need to compute the possible infinitesimal motion from the equation 2.7. So we here introduce the tool for computing simultaneous inequalities, *the theory of polyhedral convex cones*[20]. Next, we propose the algorithm for computing motion DOFs.

[1]A motion in 3D space has six DOFs, and a translation has three DOFs. So we regard the remaining DOFs as rotation DOFs. But, because a translation is a rotation when an axis is infinitely far away, we use the DOF of *the direction a rotation axis*.

2.5.1 The theory of polyhedral convex cones(PCC)

Consider the system of m linear inequalities $AX \geq 0$ shown in inequalities (2.8). X is a n-dimensional vector. The solution space of X, denoted by A^* is called a polyhedral convex cone[20].

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ & \ddots & \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \geq \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2.8}$$

For examining a PCC, we turn first to the face-structure of the PCC $A^* = \{X | A_1 X \geq 0, \cdots, A_p X \geq 0\}$. To each subset H (which may be the null set $\emptyset$) of the indices 1, $\cdots$,p there corresponds a subset F_H of A^* defined by the conditions $A_h X 0$ for each h in H, $A_h X = 0$ for each h not in H. F_H is the (open) face of A^* corresponding to H. Since H can be chosen in 2^p possible ways, and since non-vacuous faces corresponding to distinct subsets of indices are clearly disjoint, we see that A^* is partitioned into 2^p faces, some of which may be vacuous. If F^H is not vacuous, then it is the intersection $F_H = O_H \cap L_H$ of the (open) set O_H of all vectors X satisfying $A_h X 0$ for each h in H and the linear subspace L_H consisting of all vectors X satisfying $A_h X = 0$ for each h not in H. The dimension d_H of L_H is given by $d_H = n - r_H$, where r_H is the maximal number of linearly independent equations in the system of equations determining L_H. We say that F_H is a face of dimension d_H.

2.5.2 The algorithm for computing motion DOFs

Let simultaneous inequality as shown in the equation (2.7) be given. The possible solution region represents how an object can move. In the result of the theory of polyhedral convex cones, we can analyze the solution region by the range of the dimension of the face of PCC. First, We would like to propose the method to compute the range of the dimension of faces of PCC. Next, we would like to compute motion DOFs using that.

The method to compute the range of the dimension of faces of PCC

For analyzing simultaneous inequalities, we need to compute the range of the dimension of faces of PCC. First, the simultaneous inequalities(2.8) can be transformed to the equivalent simultaneous equalities(2.9) using non-negative variables.

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} u_1 \\ \vdots \\ u_m \end{pmatrix} \quad (\forall i, u_i \geq 0) \qquad (2.9)$$

Next, the following algorithm is applied:

1. Searching for a combination of rows which are not independent.
2. Obtaining the equation as shown in the equation (2.10) including only u_i by erasing x_1, ..., and x_n from the combination.
$$c_1 u_{i_1} + \cdots + c_l u_{i_l} = 0 \qquad (2.10)$$
3. Solving $u_{i_1} = \cdots = u_{i_m} = 0$, when $\forall k, c_k 0$ or $u_{i_k} = 0$, or $\forall k, c_k < 0$ or $u_{i_k} = 0$.
4. Repeating Steps 1 to 3 until the number of the elements which $u_i = 0$ gets does not increase.

Then, the equalities composed of the simultaneous equalities (2.9) are divided into two types of equalities; one is the equality in which $u_i = 0$ referred as to *equality-part*, and the other is the equality in which $u_i \geq 0$ referred as to *inequality-part*. In short, we obtain optimal simultaneous equalities and inequalities (2.11) with the same solution region.

$$BX = 0 \quad CX \geq 0 \qquad (2.11)$$

Since if $\exists i, \{i \in H | u_i = 0\}$, F_H is vacuous, the range of the dimension of faces of PCC is from $m - r_{all}$(i.e. $H = \emptyset$) to $6 - r_{eq}$(i.e. $u_i \geq 0 \Leftrightarrow i \in H$), where r_{all} is the rank of the matrix $\begin{bmatrix} B \\ C \end{bmatrix}$ and r_{eq} is the rank of the matrix B.

Computing translation DOFs

For computing motion DOFs, we need to determine how a object can translate and rotate. In order to determine possible translation, we need to compute the solution region of simultaneous inequalities(2.12), because the vector $[0, S_1]$ represents translation in the screw representation.

$$\begin{pmatrix} a_{14} & a_{15} & a_{16} \\ & \vdots & \\ a_{n4} & a_{n5} & a_{n6} \end{pmatrix} \begin{pmatrix} t_4 \\ t_5 \\ t_6 \end{pmatrix} \geq \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \tag{2.12}$$

We determine translation DOFs from the rank and the range of the dimension of faces of PCC using Table 2.1.

Table 2.1: Translation DOFs

rank	the dimension of faces of PCC	maintaining	detaching	constraining
0	{3}	3	0	0
1	{2}	2	0	1
	{2,3}	2	1	0
2	{1}	1	0	2
	{1,2}	1	1	1
	{1,2,3}	1	2	0
3	{0}	0	0	3
	{0,1}	0	1	2
	{0,1,2}	0	2	1
	{0,1,2,3}	0	3	0

Computing rotation DOFs

Since the simultaneous inequalities(2.7) represent the infinitesimal possible motion including translation and rotation, it is difficult to compute rotation DOFs only. First, we compute all motion DOFs using Table 2.2.

Table 2.2: Translation DOFs

rank	the dimension of faces of PCC	maintaining	detaching	constraining
0	{6}	6	0	0
1	{5}	5	0	1
	{5,6}	5	1	0
2	{4}	4	0	2
	{4,5}	4	1	1
	{4,5,6}	4	2	0
3	{3}	3	0	3
	{3,4}	3	1	2
	{3,4,5}	3	2	1
	{3,4,5,6}	3	3	0
4	{2}	2	0	4
	{2,3}	2	1	3
	{2,3,4}	2	2	2
	{2,3,4,5}	2	3	1
	{2,3,4,5,6}	2	4	0
5	{1}	1	0	5
	{1,2}	1	1	4
	{1,2,3}	1	2	3
	{1,2,3,4}	1	3	2
	{1,2,3,4,5}	1	4	1
	{1,2,3,4,5,6}	1	5	0
6	{0}	0	0	6
	{0,1}	0	1	5
	{0,1,2}	0	2	4
	{0,1,2,3}	0	3	3
	{0,1,2,3,4}	0	4	2
	{0,1,2,3,4,5}	0	5	1
	{0,1,2,3,4,5,6}	0	6	0

Next, we analyze with regard to the rotation axis. Rotation axes are divided into three types as follows: (shown in Figure 2.11)

- Rotating both clockwise and counter-clockwise around an axis (referred as to *type 1 axis*).
- Rotating either clockwise or counter-clockwise around an axis (referred as to *type 2 axis*).
- Not rotating around an axis (referred as to *type 3 axis*).

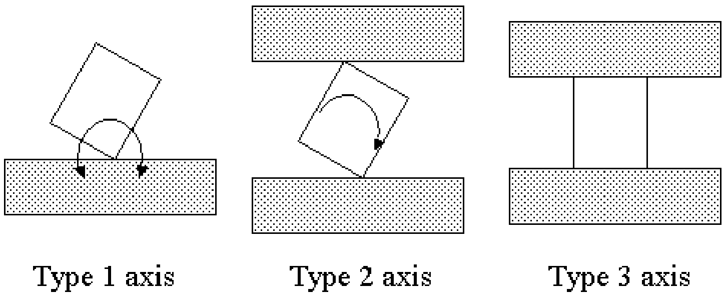


Figure 2.11: Definition of three types of axes

First, the simultaneous inequality(2.7) is transformed into the equivalent simultaneous equality(2.13) using non-negative variables.

$$\begin{pmatrix} a_{14} & a_{15} & a_{16} \\ & \vdots & \\ a_{n4} & a_{n5} & a_{n6} \end{pmatrix} \begin{pmatrix} t_4 \\ t_5 \\ t_6 \end{pmatrix} = \begin{pmatrix} u_1 \\ \vdots \\ u_m \end{pmatrix} - \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ & \vdots & \\ a_{n1} & a_{n2} & a_{n3} \end{pmatrix} \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \tag{2.13}$$

$$(\forall i, u_i \geq 0)$$

$\begin{pmatrix} a_{14} & a_{15} & a_{16} \\ & \vdots & \\ a_{n4} & a_{n5} & a_{n6} \end{pmatrix}$ is denoted by A_r

Next, the below algorithm is applied:

1. Solving $u_1, \cdots, u_m$ using the algorithm mentioned above.

Table 2.3: the axis DOFs

rank	the dimension of faces of PCC	Type 1 axis	Type 2 axis	Type 3 axis
0	{3}	3	0	0
1	{2}	2	0	1
	{2,3}	2	1	0
2	{1}	1	0	2
	{1,2}	1	1	1
	{1,2,3}	1	2	0
3	{0}	0	0	3
	{0,1}	0	1	2
	{0,1,2}	0	2	1
	{0,1,2,3}	0	3	0

2. Searching a combination of rows of matrix A_r which are not independent.
3. Obtaining the equality (2.14) including only u_i, t_1, t_2 and t_3 by erasing t_4, t_5, and t_6 from the combination.

$$d_1t_1 + d_2t_2 + d_3t_3 = c_1u_{i_1} + \cdots + c_lu_{i_l} \tag{2.14}$$

4. Memorizing the inequality $d_1t_1 + d_2t_2 + d_3t_3 \geq 0$, if $\forall k, c_k 0 \quad or \quad u_{i_k} = 0$.
5. Memorizing the inequality $-d_1t_1 - d_2t_2 - d_3t_3 \geq 0$, if $\forall k, c_k < 0 \quad or \quad u_{i_k} = 0$.
6. Repeating Steps 2 to 5 until a new inequality is not memorized.

The simultaneous inequalities composed of memorized inequalities represent the possible axis direction. Applying the theory of PCC, axis direction DOFs can be divided into the three types mentioned above using Table 2.3.
Finally, a rotation DOF can be computed using the equation(2.15), where m_t and m_a represent the numbers of maintaining DOFs in translation and all motion,

m_r, d_r, and c_r represent the numbers of maintaining, detaching, and constraining DOFs in rotation, ax_3 represents the numbers of DOFs of the type 3 axis.

$$\begin{aligned} m_r &= m_a - m_t \\ d_r &= 3 - m_r - c_r \\ c_r &= ax_3 \end{aligned} \tag{2.15}$$

2.6 Singular contact

Consider the case as shown in Figure 2.12. The infinitesimal possible motion computed by the method mentioned above is not correct, because the region of the possible motion near a present object configuration in c-space is not convex; rather, it is concave. So it is impossible to represent the motion in a simultaneous inequalities.

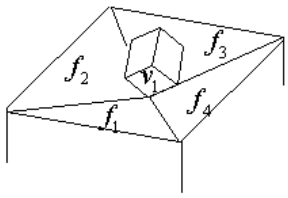


Figure 2.12: Singular contact relation

We should classify contact relations into the *singular* contact relations and the non-singular contact relations. First, we draw the condition of a singular contact relation. Next, we propose the analysis method for it.

2.6.1 Condition of a singular contact relation

In a singular contact relation, it is satisfied that the region of the possible configuration near a present object configuration is concave in c-space. That means that the convex vertex is in contact with the convex vertex or edge. In short, the contact elements composed of a contact relation include two kinds of contacts referred as to *singular contact* as shown in Figure 2.13.

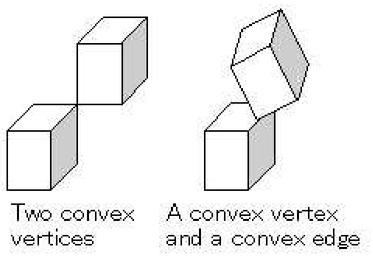


Figure 2.13: Two types of singular contacts

2.6.2 Possible infinitesimal motion in singular contact relations

For the analysis, we need to formulate a possible infinitesimal motion. Using a richer representation than simultaneous inequalities enables us to do so as follows:

Single singular contact

Two convex vertices contact

In the two convex vertices contact case as shown to the left in Figure 2.13, the contact relation is composed of some vertex-face contacts. When $A_1T \geq 0, \cdots$, and $A_nT \geq 0$ represent corresponding single contacts in the screw representation, where n is the number of composed contacts, A_i is a 1×6 matrix, and $T =^t (t_1, \cdots, t_6)$, the possible infinitesimal motion can be formulated as shown in the equation (2.16).

$$A_1T \geq 0 \; or \; \cdots \; or \; A_nT \geq 0 \tag{2.16}$$

Convex vertex-convex edge contact

In the convex vertex-convex edge contact case as shown to the right in Figure 2.13, the contact relation is composed of two vertex-face contacts. When $A_1T \geq 0$ and $A_2T \geq 0$ represent corresponding single contacts in the screw representation, the infinitesimal possible motion can be formulated as shown in the equation (2.17).

$$A_1T \geq 0 \; or \; A_2T \geq 0 \tag{2.17}$$

All singular contact relations

All singular contact relations can be represented as the equation (2.18).

$$\left(\bigcap_i A_i T \geq 0\right) \bigcap \left(\bigcap_i \bigcup_j B_{ij} T \geq 0\right) \tag{2.18}$$

2.6.3 Analyzing singular contact relations

In the singular contact relation case, we apply the non-singular analysis method to the contact relation removing singular contacts from original singular contact relation. In short, given the equation (2.18) representing the singular contact relation, we apply the non-singular analysis method to the simultaneous inequality as shown the equation (2.19).

$$\bigcap_i A_i T \geq 0 \tag{2.19}$$

In this case, the DOFs are referred as to *singular maintaining, singular detaching*, and *singular constraining* DOFs..

2.7 Analyzing transitions of DOFs

Transitions of contact relations leads transitions of motion DOF of a grasping object. For existing six kinds of DOFs – maintaining, detaching, constraining, singular maintaining, singular detaching, and singular constraining – there is the possibility that ${}_6C_2 = 15$ transitions exist.

Now, assuming not to exist the motion maintaining singular contact[2], ${}_3C_2 = 3$ transitions between singular maintaining, singular detaching, and singular constraining can be removed.

Next, considering the case of transitions, maintaining or detaching to (singular) constraining, the *entrance* DOF, singular maintaining or singular detaching, exists between these transitions as shown in Figure 2.14. Therefore, these transitions are impossible..

[2]In this paper, our subgoal is to recognize human assembly tasks, and a human avoids the motion like that in assembly tasks. So we believe that this assumption is proper.

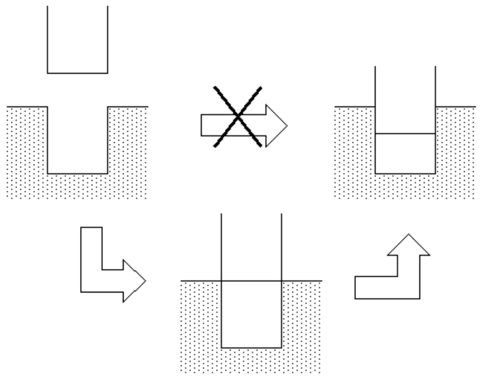


Figure 2.14: impossible transitions directly

Only eight possible transitions between these six types of DOFs exist as shown in Figure 2.15. Among these transitions, three transitions – maintaining to detaching, maintaining to singular maintaining, and maintaining to singular detaching – have the characteristic that the direction of the motion of an object is the same as the direction of the changed DOF. We call this type of transitions *class 1 sub-skills*. The other five transitions have the characteristic that the direction of the motion is different from the direction of the changed DOF. We call this type of transitions *class 2 sub-skills*.

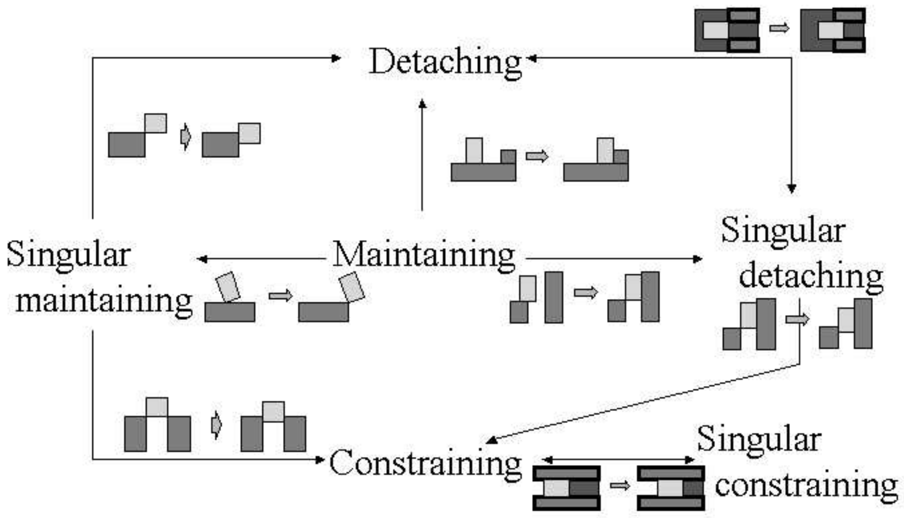


Figure 2.15: Possible transitions

2.8 Sub-skill

2.8.1 Class 1 sub-skill

Maintaining to detaching

The motion as shown in Figure 2.16 leads the transition, maintaining to detaching. We call these sub-skills *make contact in translation* and *in rotation.*

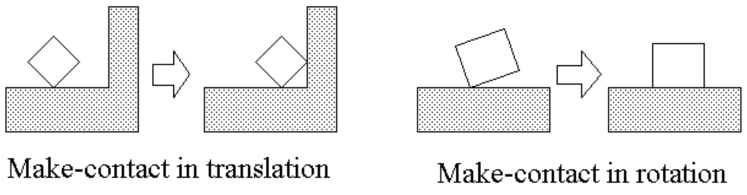


Figure 2.16: Maintaining to detaching

Maintaining to singular maintaining

The motion as shown in Figure 2.17 leads the transition, maintaining to singular maintaining. We call these sub-skills *slide in translation* and *in rotation.*

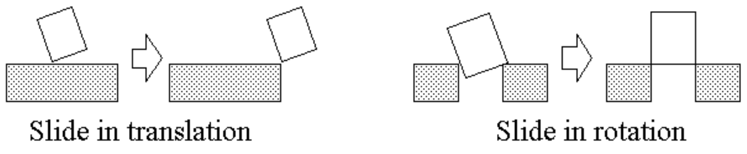


Figure 2.17: Maintaining to singular maintaining

Maintaining to singular detaching

The motion as shown in Figure 2.18 leads the transition, maintaining to singular detaching. This motion is like the motion combined with make-contact and slide and, in the implementation, this motion can be treated as a make-contacts sub-skill.

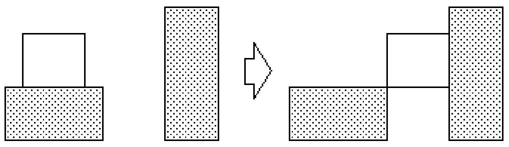


Figure 2.18: Maintaining to singular detaching

2.8.2 class 2 sub-skill

Singular maintaining to constraining

The motion as shown in Figure 2.19 causes the transition, singular maintaining to constraining. Because the transition requires precise control of an object configuration, it is very subject to error.

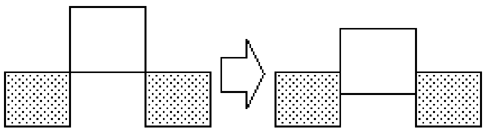


Figure 2.19: Singular maintaining to constraining

Singular detaching to constraining

The motion as shown in Figure 2.20 causes the transition, singular detaching to constraining. In this case, because the transition requires precise control, it is also subject to error. But because there is the supposed DOF, singular detaching (in this case, we can complete this sub-skill to insert the object pushing below), it is easier to complete this sub-skill than to complete the sub-skill, singular maintaining to constraining.

Singular maintaining to detaching

The motion as shown in Figure 2.21 causes the transition, singular maintaining to detaching. Though it is very difficult to realize the singular contact relation, in this case this transition is robust to object configuration errors.

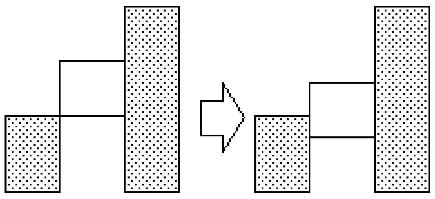


Figure 2.20: Singular detaching to constraining

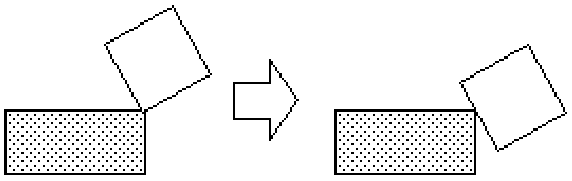


Figure 2.21: Singular maintaining to detaching

2.9 Assigning sub-skills

We now describe the method for assigning a proper sub-skill from a transition. The definition of class 1 sub-skills lists the rules for assigning class 1 sub-skill as follows:

- Assign a make-contact sub-skill in translation (rotation), if the transition, maintaining to (singular) detaching, occurs in translation (rotation).
- Assign a slide sub-skill in translation (rotation), if the transition, maintaining to singular maintaining, occurs in translation (rotation).

Although the former rule is correct, the latter rule is not correct. Why?
For example, with regard to the parallel edge-edge contact case as shown in Fig 2.22, the DOF around the edge can be considered to be a*not-singular* maintaining DOF in fact, because an object can rotate maintaining the contact relations. So, a slide sub-skill is assigned the transition; then, maintaining to *true singular maintaining*, occurs. An object cannot move maintaining a contact relation in a true singular maintaining DOF (referred to as a *restricted DOF*. Of course, a restricted

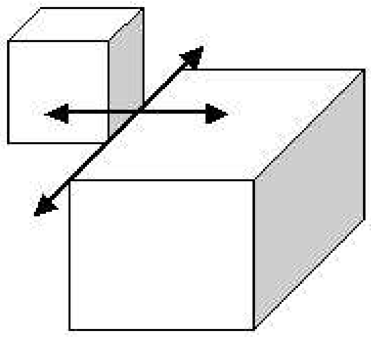


Figure 2.22: True singular maintain?

DOF includes a detaching and a constraining DOF. The number of restricted DOFs is equal to the rank of the matrix (2.20), given the equation (2.18) representing the possible infinitesimal motion of an object in the screw representation[3]

$$\begin{bmatrix} A_i & \cdots & B_{11} & \cdots \end{bmatrix} \tag{2.20}$$

And the number of the true singular maintaining DOFs m_s can be computed using the equation (2.21), where d, c, and r are the numbers of the singular detaching, singular constraining, and restricted DOFs.[4]

$$m_s = r - c - d \tag{2.21}$$

The correct rules for assigning for class 1 sub-skills are as follows:

- Assign a make-contact sub-skill in translation (rotation), if the transition, maintaining to (singular) detaching, occurs in translation (rotation).
- Assign a slide sub-skill in translation (rotation), if the transition, maintaining to true singular maintaining, occurs in translation (rotation).

The same rules apply for assigning class 2 sub-skills.

[3]Considering the motion is maintaining a contact relation only, simultaneous equations representing a possible infinitesimal motion are the same as one in non-singular contact.

[4]Of source, in non-singular case, $r = c + d$.

Chapter 3

Robot Vision System

In this system, 2D and 3D images are obtained by the multi-baseline real time stereo system. (Shown in Figure 3.1) By processing these images, trajectories of an grasping object and a configuration of an environmental object are obtained. In this chapter, we describe the image processing method using this system.

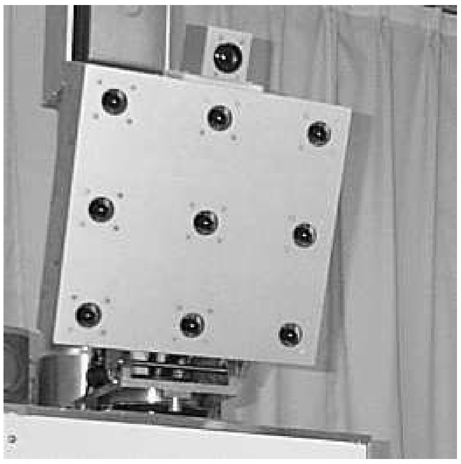


Figure 3.1: Multi-baseline real time stereo system

3.1 Extracting assembly parts

First, we extract assembly parts only. Because the light source changes little during human assembly tasks, we extract these parts using *background subtraction* (Shown in Figure 3.2).

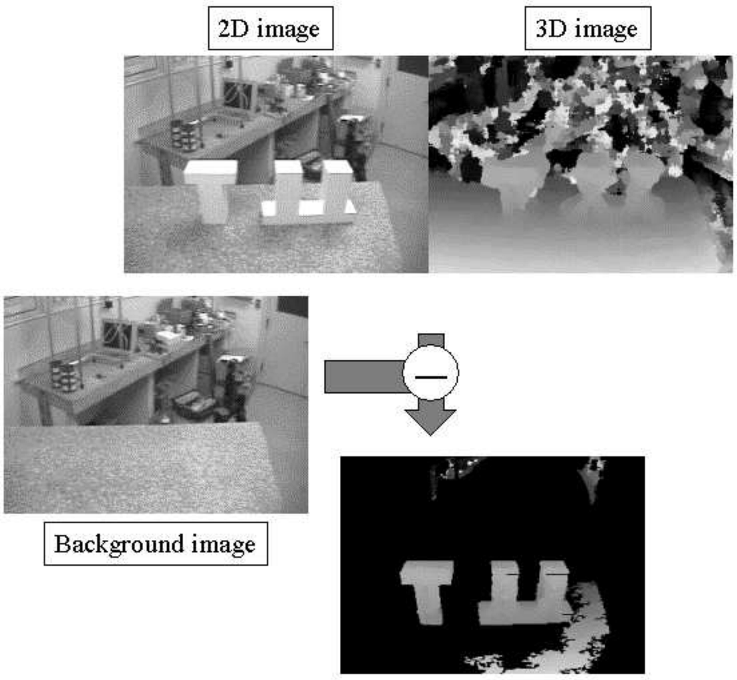


Figure 3.2: Extracting assembly parts using background subtraction

3.2 Classifying a grasping object and an environmental object

Next, we classify a grasping object and another object referred to as an environmental object. Because an environmental object is fixed, we can divide it into a grasping and an environmental object as shown in Figure 3.3 from the histogram of a pixel of a 3D image through a time sequence.

3.3 3D template matching system

Then, we obtain object configurations from those 3D images by using the 3D template matching system [4], which is a kind of iterative closet point method, but it can obtain more precise object configuration by *weighted least square* method. Figure 3.4 shows obtained object configurations. For errors of vision systems,

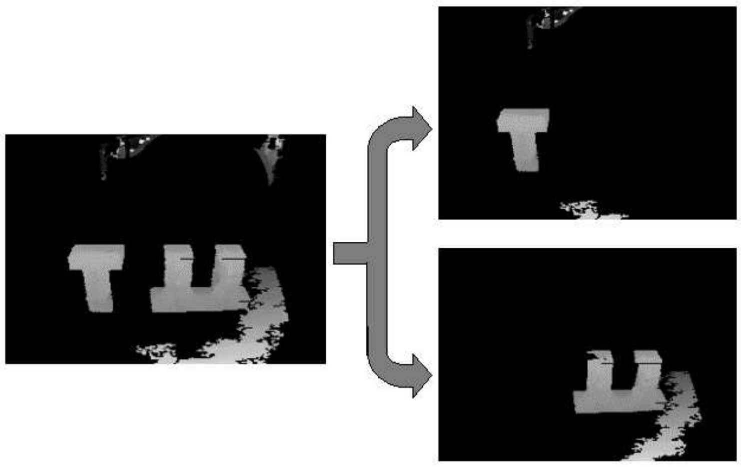


Figure 3.3: Dividing parts into a grasping and an environmental object

we cannot obtain correct object configurations for the screw theory recognition mentioned above. Fortunately, we can obtain it by using the contact relation that is more robust to errors of vision system[2].

3.4 Vision error correct

As mentioned above, an obtained object configuration from the robot vision system includes some errors to prevent the screw theory recognition mentioned above. These errors need to be corrected. We now describe the method for removing the errors by using a contact relation represented as a finite combination of vertex-face, face-vertex, and edge-edge contacts. Generally speaking, it is impossible to obtain the object configuration satisfying the given contact relation, because that is the equivalent of solving a non-linear redundant equation. But in this system, because of obtaining an approximate answer, we can solve this problem by using the iterative least square method[2]. The problem solving method is as follows:

1. Formulate a condition of an object configuration maintaining each contact.
2. Transform this formula to a linear equation by using Taylor expansion.
3. Solve these simultaneous equations by using the least square method.

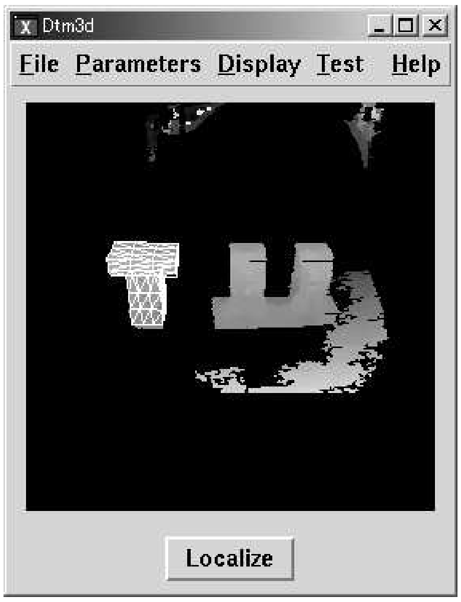


Figure 3.4: 3D template matching system

4. Repeat Steps 1 to 3 until the correct answer is obtained.

3.4.1 Formulation of a condition of a contact

As mentioned above, an object configuration obtained from the robot vision includes some errors. So we correct the errors by moving the object. When an object rotates γ, β,and α around z-axis, y-axis, and x-axis, and translate x, y, z along x-axis, y-axis, and z-axis, a surface normal n and a vertex v move to n' and v' as shown in the equation (3.1).

$$\begin{aligned} n' &= Rn \\ v' &= Rv + T \end{aligned} \tag{3.1}$$

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{pmatrix} T = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Single contact

Vertex-face contact

In the vertex-face contact case as shown in Figure 3.5, a vertex v' after error correction is computed using the equation (3.2).

$$v' = Rv + T \tag{3.2}$$

Because vertex v' is on the face f, the equation (3.3) is obtained.

$$\Delta e_{vf} = n \cdot (Rv + T - f) = 0 \tag{3.3}$$

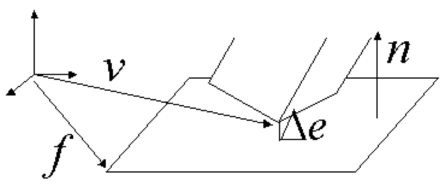


Figure 3.5: Vertex-face contact

Face-vertex contact

In the face-vertex contact case as shown in Figure 3.6, a vertex f' on the face and a face normal n' after error correction are computed using the equation (3.4).

$$\begin{aligned} n' &= Rn \\ f' &= Rf + T \end{aligned} \tag{3.4}$$

Because vertex v is on the face f', the equation (3.5) is obtained.

$$\begin{aligned} \Delta e_{fv} &= Rn \cdot (v - Rf - T) \\ &= Rn \cdot v - n \cdot f - Rn \cdot T = 0 \ \ (\because {}^{t}RR = E) \end{aligned} \tag{3.5}$$

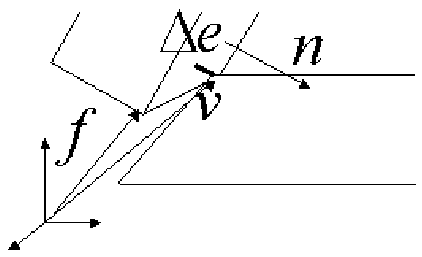


Figure 3.6: Face-vertex contact

Edge-edge contact

In the edge-edge contact case as shown in Figure 3.7, a direction vector e_1' and a point p_1' on the edge after error correction are computed in Equation 3.6.

$$\begin{aligned} e_1' &= Re_1 \\ p_1' &= Rp_1 + T \end{aligned} \tag{3.6}$$

Because the edge E_1 is IN contact with the edge E_2, the equation (3.7) is obtained.

$$\Delta e_{ee} = \frac{(Re_1 \times e_2)\cdot(p_2 - Rp_1 - T)}{|Re_1 \times e_2|} = 0 \tag{3.7}$$

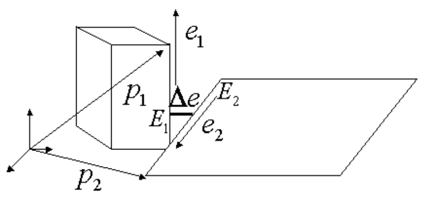


Figure 3.7: Edge-edge contact

Multiple contact

In a multiple contact case, simultaneous equations composed of equations corresponding to each single contact represent an satisfied object configuration.

3.4.2 Taylor Expansion

Applying Taylor Expansion to the equation (3.3), (3.5), and (3.7), the equations 3.8 , 3.9, and 3.10 are obtained, where $r =^t (\alpha, \beta, \gamma)$ and $P = \frac{p_2 - p_1 + \{n \cdot (p_2 - p_1)\} n}{|e_1 \times e_2|}$.

$$\Delta e_{vf} \quad = \quad n \cdot (v - f) - [-{}^t n \; {}^t(n \times v)] \begin{bmatrix} T \\ r \end{bmatrix} \tag{3.8}$$

$$\Delta e_{fv} \quad = \quad n \cdot (v - f) - [{}^t n \; {}^t(v \times n)] \begin{bmatrix} T \\ r \end{bmatrix} \tag{3.9}$$

$$\begin{aligned} \Delta e_{ee} \quad &= \quad n \cdot (p_2 - p_1) \\ &- \quad [{}^t n \; {}^t\{p_1 \times n + (e_2 \times p) \times e_1\}] \begin{bmatrix} T \\ r \end{bmatrix} \end{aligned} \tag{3.10}$$

All of the equations can be represented in the form as shown in the equation (3.11).

$$\Delta e = c - [{}^t a {}^t b] \begin{bmatrix} T \\ r \end{bmatrix} \tag{3.11}$$

3.4.3 Least square method

As mentioned above, we obtained the formulation for vision error correction. Now, we need to compute the answer satisfying each equation $\Delta e = 0$. Unfortunately, for approximation by Taylor expansion, the answer may not exist. So, the condition all $\Delta e = 0$ is transform to the equation (3.12), and we obtain the answer by using *least square method*.

$$\sum (\Delta e)^2 = 0 \tag{3.12}$$

Because the answer makes the value of the equation (3.12) the smallest, it satisfies the condition as shown in the equation (3.13), where $q = \begin{bmatrix} T \\ r \end{bmatrix}$.

$$\frac{\partial}{\partial q} \sum (\Delta e)^2 = 0 \tag{3.13}$$

We need to solve the equation (3.14), where n is the number of contacts.

$$\begin{aligned} \frac{\partial}{\partial q} \sum (\Delta e)^2 \quad &= \quad \sum_i^n \frac{\partial}{\partial q} (c_i - [{}^t a_i \; {}^t b_i] q)^2) \\ &= \quad \sum_i^n \begin{bmatrix} a_i \\ b_i \end{bmatrix} (c_i - [{}^t a_i \; {}^t b_i] q) = 0 \end{aligned}$$

$$\left[\sum_i^n \begin{pmatrix} a_i \\ b_i \end{pmatrix} [{}^t a_i \ {}^t b_i]\right] q = \sum_i^n c_i \begin{bmatrix} a_i \\ b_i \end{bmatrix} \tag{3.14}$$

Because the rank of the matrix $\sum_i^n \begin{bmatrix} a_i \\ b_i \end{bmatrix} [{}^t a_i \ {}^t b_i]$ may not be six, the equation may not be solved by using the inverse matrix. But by using *singular value decomposition* method, we can solve it.

Singular value decomposition

Given an $m \times n$ matrix A but $m \geq n$, singular value decomposition of A is

$$A = UW\,{}^tV,$$

where U is an $m \times n$ column-orthogonal matrix, V is an $n \times n$ orthogonal matrix, and W is an $n \times n$ diagonal matrix with positive or zero elements.

$$W = diag(w_i)$$

Now, given the simultaneous equations $Ax = b$, x and b is a $m \times 1$ matrix, the answer is

$$x =^t UW^*Vb,$$

$$\text{where } W^* = diag(w_i^*), w_i^* = \begin{cases} 1/w_i & (w_i \neq 0) \\ 0 & (w_i = 0) \end{cases}$$

3.5 Possibility of transition between contact relations

Though we describe the method to correct vision errors using obtained contact relations, the contact relations obtained under the situation existing vision errors may include some errors. So we now propose a method to remove those errors under the assumption that all transitions must be possible.

First, we consider the relationship between possible transitions and the connection between c-surfaces in c-space, Next, we propose a fast, practical algorithm.

3.5.1 Possible transitions and connection of c-surface

Consider the transitions between contact relations as shown to the left in Figure 3.8. The arrows represent possible direct transitions between two contact relations. C-surfaces of corresponding contact relations are shown to the right in the Figure 3.8.

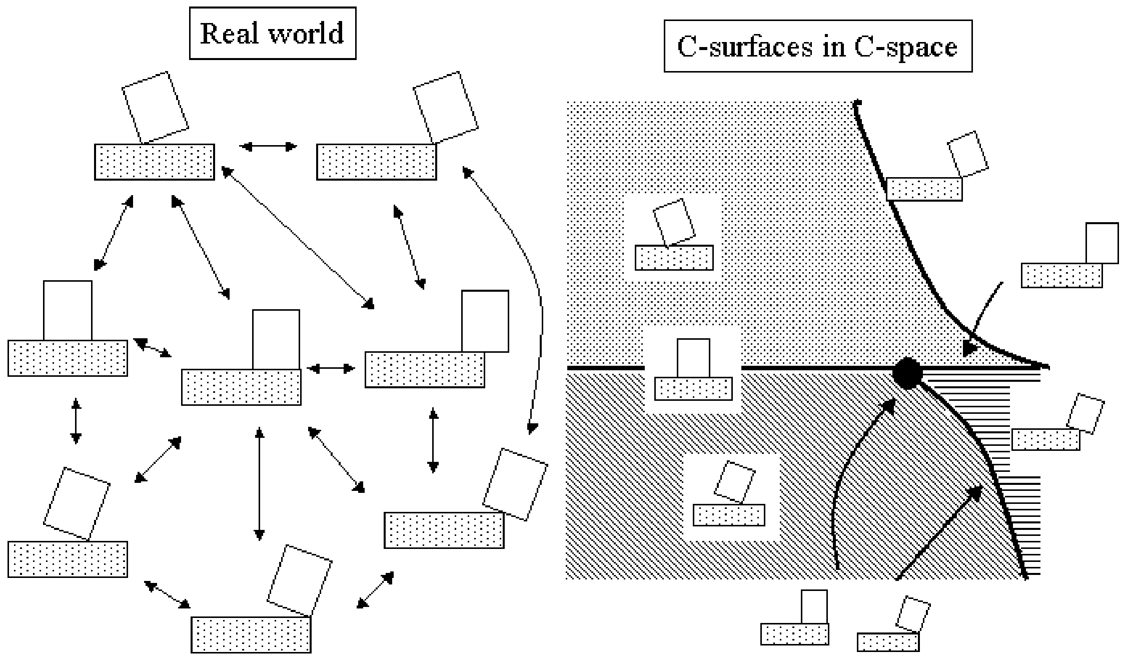


Figure 3.8: Relationship between possible transitions and c-surface

Possible direct transitions exist between two contact relations A and B if, and only if,

$$\{a_i\} \in \text{c-surface of } A \text{ and } b \in \text{c-surface of } B,\ lim_{i\to\infty}\{a_i\} = b$$

$$\text{or}$$

$$\{b_i\} \in \text{c-surface of } B \text{ and } a \in \text{c-surface of } A,\ lim_{i\to\infty}\{b_i\} = a,$$

in short, c-surfaces A and B are connected directly each other.

3.5.2 Fast algorithm for deciding the possibility of a direct transition

As mentioned above, though we can decide whether the transition is possible or not from the connection between two c-surfaces, treating a c-surface is very difficult. So we propose a fast practical algorithm.
First, we consider the method to formula a c-surface. The method is as follows:

1. The contact relation is represented as the combination of the vertex-face, face-vertex, and edge-edge contacts.
2. Each element is represented as the equations (3.3), (3.5), and (3.7), and the range of an object configuration maintaining each contact is represented as an inequality.
3. The c-surface is represented as the simultaneous equations obtained.

Notice that **the solution region of each contact equation is different from the others**. From this fact, we can learn the next assumption.

Proposition 1 *Two c-surfaces with the same DOFs are not directly connected.*

Proof If two c-surfaces intersect each other, the contact relations corresponding to the intersection must exist and contradict to connect directly. We assume two c-surfaces do not intersect. But a c-surface surrounds another c-surface that has not the same but different DOFs.

Proposition 2 *Given two contact relations A and B, the corresponding c-surfaces C_a and C_b are not connected directly, if $\exists a \in A, a \not{i}nB$ and $\exists b \in B, b \not{i}nA$*[1]

Proof Because the DOFs of C_a and C_b are different, we suppose the DOFs of C_a are less. We assume c-surfaces C_a and C_b are connected. Because the DOFs of C_b are greater, it is satisfied that $\{a_i\} \in C_a$ and $b \in C_b$, $lim_{i\to\infty}\{a_i\} = b$. Considering $\exists p \in A, p \not{i}nB$, the contact p is detached while the transition from C_a to C_b. But

[1] In this paper, a parallel edge-edge contact is represented as the combination of several vertex-face and face-vertex contacts. Here, a parallel edge-edge contact needs to be represented as the combination added to the edge-edge contact.

the solution region of the c-surface maintaining the contact p is a closed set; that contradicts that $\{a_i\} \in C_a$ and $b \in C_b$, $lim_{i \to \infty}\{a_i\} = b$.

Proposition 3 *Given two contact relations A and B satisfying that $\forall a \in A, a \in B$, and the corresponding c-surface C_a and C_b are not connected directly, if constraining DOFs in contact relation B translate maintaining or detaching DOFs in contact relation A.*

Proof Shown in Figure 2.15

Proposition 3 is a necessary condition, but not a sufficient condition. For example, considering the transition as shown in Fig 3.9, Proposition 3 is obviously satisfied , but it is an impossible direct transition[2] But practically, it is no problem, because this situation does not happen in real assembly tasks.

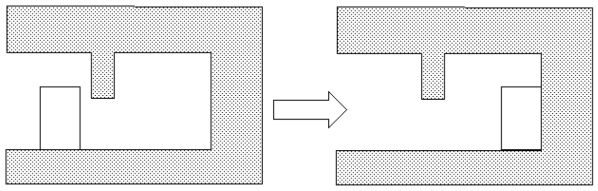


Figure 3.9: Possible transition?

So given two contact relations $A = \{a_1, \cdots, a_m\}$ and $B = \{b_1, \cdots, b_n\}$, the fast algorithm for deciding if the direct transition exists is as follows:

[2]In this case, it is a mistake to represent assembly tasks based on the mere contact relation.

```
int i,j,check;
// From Proposition 1 and 2
if ($m == n$) return “no”;
// the number of the elements composed of A
//          is less than B
if ($mn$) swap(A,B);
for ($i = 1; i <= m; i++$){
    if ($a_i \notin$ B) return “no”;
}
return “yes”;
```

Chapter 4

Platform

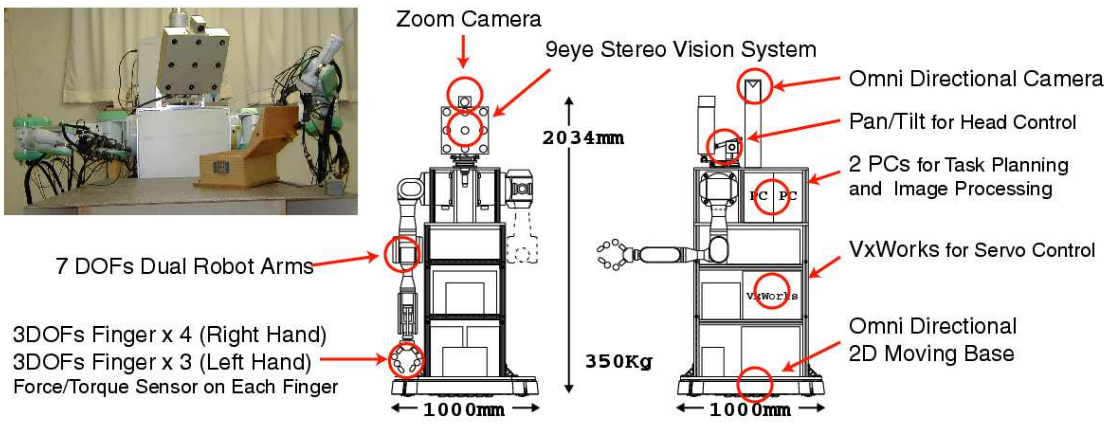

Zoom Camera
9eye Stereo Vision System
Omni Directional Camera
Pan/Tilt for Head Control
2 PCs for Task Planning and Image Processing
PC PC
VxWorks for Servo Control
VxWorks
Omni Directional 2D Moving Base
7 DOFs Dual Robot Arms
3DOFs Finger x 4 (Right Hand)
3DOFs Finger x 3 (Left Hand)
Force/Torque Sensor on Each Finger
2034mm
350Kg
1000mm
1000mm

Figure 4.1: Platform

We have developed a robot (Figure 4.1) as an experimental platform for robot learning and performing human hand-action tasks. For that purpose, we designed the robot to have similar capabilities and body parts as those of humans, including vision, two arms and upper torso.
The main features of this robot are summarized as follows:

- It is equipped with a 9-eye stereo vision system and other camera systems for object recognition (vision).

- It has dual 7 DOFs robot arms. The right arm has a hand with 4 fingers and the left arm has a hand with 3 fingers. Each finger has 3 DOFs and a

Force/Torque sensor on its tip (arms and hands).

- The omni directional moving base allows the robot move freely on a 2D plane in order to move the view point and the arms in any position (upper torso).
- CORBA-based [40] software architecture enables the robot to be programmed easily on multi-machines connected by a network and to be connected from new exterior devices such as data gloves.

4.1 Vision

Vision systems are the substitutes for the human visual sensation. This robot has 3 different visual components. The first is a 9-eye stereo vision system (described in detail later) for 3D recognition. The second is a camera (EVI-400, Sony) which has zooming capability. It is intended for 2D recognition in variable resolutions. These two vision systems are mounted on the robot head and are driven by Pan/Tilt mechanisms so that the robot can focus its attention on any points in front of it. The third component is an omni directional camera which is mounted on top of the robot body. This camera is used to detect the approach of a human or to determine an attention point which can then be analyzed by the other 2 cameras.

4.1.1 Nine-eye Stereo Vision System

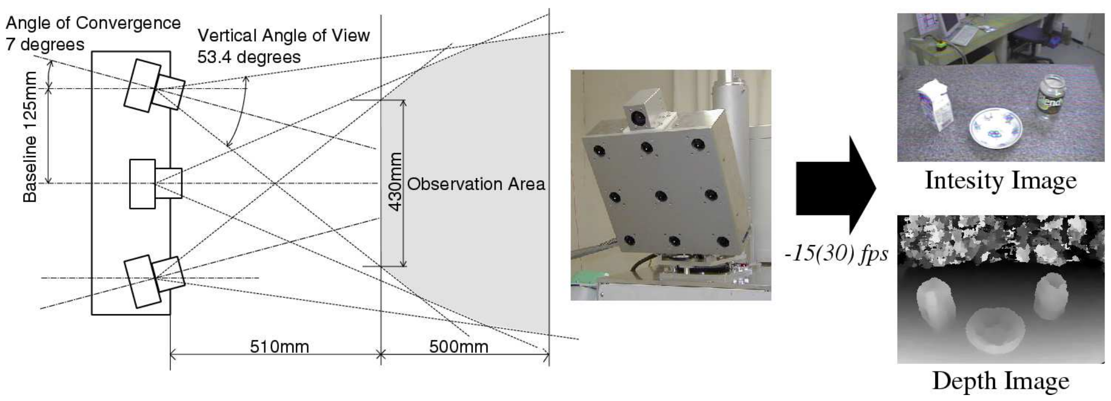


Figure 4.2: Stereo Vision System

The purpose of the 9-eye Stereo Vision System is to analyze the environment around the robot precisely in real time. The system is a product of Komatsu Ltd. [39] who adopted a multi-baseline approach [38] and has the following features.

- **Robust stereo matching**

  The stereo vision system processes 8 stereo pairs (the center camera and the exterior cameras) simultaneously and chooses the most reliable stereo pair on each pixel in the depth data.

- **Real-time processing on a hardware chip**

  The stereo calculation is processed on a hardware chip and can produce the resulting depth data (280×200 points) in real time (15fps, up to 30fps).

- **Easy customization of camera configuration**

  Users can easily rearrange the camera configuration to match their requirements. We extended the baseline and tilted the exterior cameras inward so that the stereo system can produce high resolution depth images of short distances, which are suitable for our robot. The measurable range is changeable and in this study we set up the valid range from 510mm to 1010mm, which is the closest. Figure 4.2 shows the camera configuration and the produced intensity and depth image.

4.2 Arms and Hands

As described in the previous chapter, we focus on the learning and performing of human hand-actions by a robot; the manipulation capability of the robot is essential in performing the same task as that performed by a human. This robot has dual 7DOFs PA10 robot arms (made by Mitsubishi Heavy Industry Ltd.) which have enough DOFs to move the robot hand through a wide area of 3D space (position and orientation). As a substitution for a human hand, a robot hand with fingers is attached to the tip of the robot arm. The right hand has 4 fingers and the left hand has 3 fingers. Each finger has 3 joints (3 DOFs) and is equipped with a 6 axes Force/Torque sensor on its tip. These fingers are arranged to face each other so as to enhance the grasping and manipulation capability. Figure 4.3 shows the limits of the working area of the robot arm in the level of the robot shoulder.

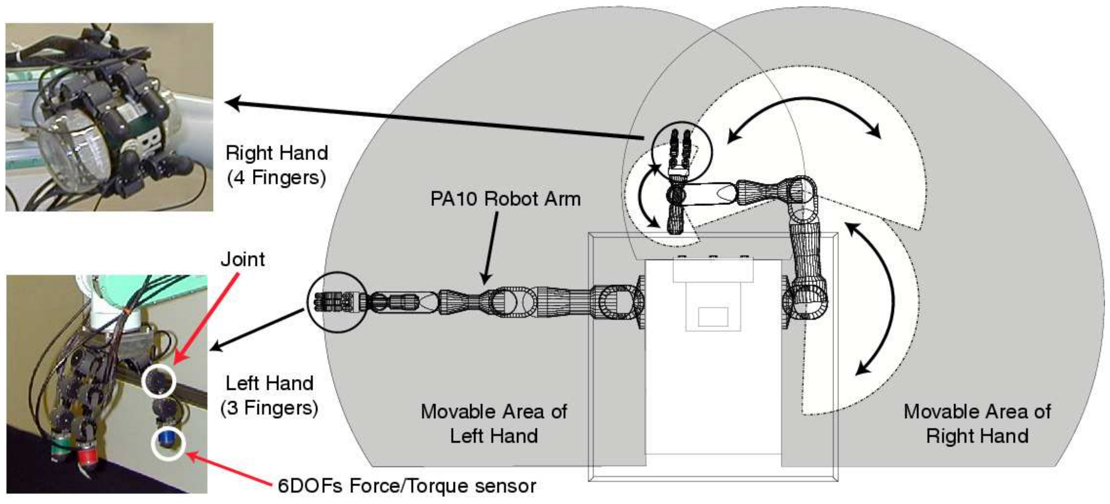


Figure 4.3: Arm and Hand Configuration

4.3 Upper Torso

Motion of the upper torso contributes to extending the robot arms and vision capability in 3D space. When humans perform certain hand-actions, they may move their heads here and there and try to see the target object from different angles. At the same time, they can twist or bend their upper torsos to exceed the limit of the working area of the arms.

To give the robot similar ability, we utilized the holonomic omni-directional vehicle [42] at the bottom of the robot. With this vehicle, the robot can move and rotate in any direction at any position on the floor. So the robot can change its point of view and the task space dynamically according to the task context.

4.4 CORBA based Software Architecture

4.4.1 Software Architecture

The robot is controlled by distributed software components on different machines connected by a network. Each of the hardware devices with which the robot is equipped has its specific control software process (abbreviated to “CS”, Component Server), and a brain process chooses the necessary CSs and accesses each CS across a network (LAN) to control the hardware resources in the robot. These hardware resources include PA10 robot arms, 9-eye stereo vision system, data gloves, etc. (Table. 4.1). The main reasons for constructing the robot software architecture by these distributed components are described below:

1. The robot can be shared by several laboratory members (users) at the same time. For example, one user processes the data from data gloves while another user carries out an experiment using the arms and the stereo vision, and a third user can perform recognition of the environment using the omni directional camera. Users must run their own brain programs and access only the necessary CSs.
2. If a software architecture is constructed in a monolithic form, a serious error in a part of the program brings about the termination of the entire program. CSs are independent processes, so, for example, a vital error in an image processing program (typically executed in the Stereo Vision CS) will not stop the arm movement in an abnormal state.

Each CS manages a specific hardware device directly (Table 4.1), so it must reside in the same machines in which the device is installed, while a brain can be on any machine. Each CS provides a set of APIs in order to be accessed by a brain process. We implemented these APIs by means of Common Object Request Broker Architecture (CORBA) technology.

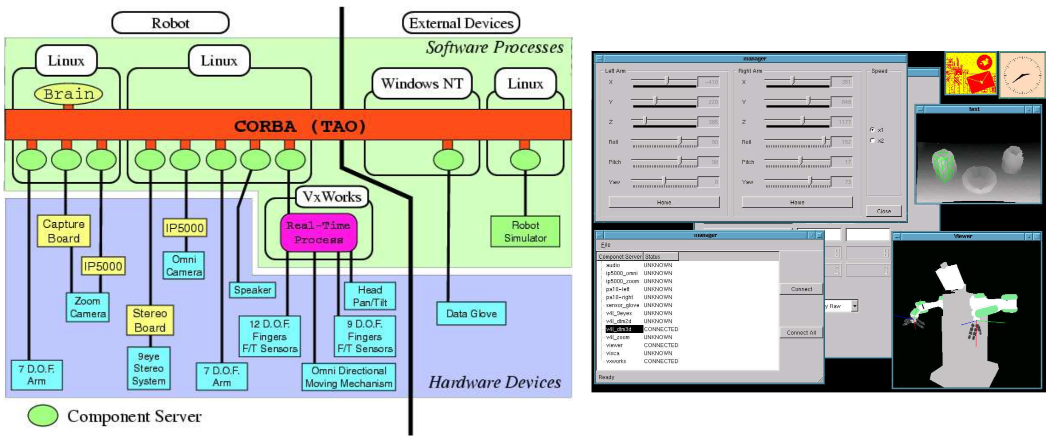
Table 4.1: Component Servers

Component Server	Control Devices Devices	Functions
Audio	Speaker	Speech synthesis system
IP5000	IP5000 board	Image processing
PA10	PA10 manipulators	Calculation of inverse kinematics of the arms and Controller of the PA10 manipulators
Sensor Glove	Cyber Glove	HMM based gesture recognition
2DTM	Zoom camera	Image processing by 2D Template Matching
3DTM	9-eye stereo vision	Image processing by 3D Template Matching
Viewer		Robot motion simulator
Visca	Zoom camera	Camera controller by Visca(TM) protocol
VxWorks	Fingers, neck, moving base	Control command generator for devices which require real-time servo control on VxWorks

4.4.2 CORBA

For a communication middle-ware between brains and CSs, we adapted CORBA [40] technology, which is a distributed object computing infrastructure being standardized by the Object Management Group (OMG).
In CORBA, a communication between a client object (brain) and a server object (CS) is handled by an Object Request Broker (ORB). Using an ORB, a client can transparently invoke a method on a server object, which can be on the same machine or across a network. The ORB accepts the call and is responsible for finding an object that can implement the request, pass it the parameters, invoke its method, and return the results. The interface of the server object is strictly defined by Interface Definition Language (IDL) in an architecture-independent manner and the client does not have to be aware of where the object is located, its programming language, its operating system, or any other system aspects that are not part of an object's interface. In so doing, the ORB provides interoperability between

applications on different machines in heterogeneous distributed environments.
The robot consists of different operating systems (Linux and Windows NT) on separate machines and the combination of brains and CSs are varied according to tasks and number of users. For this reason, we utilize CORBA technology as a base infrastructure of the robot software and define interfaces (APIs) of CSs by CORBA. By defining interfaces in IDL, we can easily attach new exterior devices (such as data gloves) to the robot.
We utilized TAO ORB [41] as an implementation of CORBA. TAO is the result of research on high-performance and real-time CORBA which is freely available. TAO supports a wide range of operating systems (including Linux and Windows NT) on several hardware architectures. The supported programming language is C++; therefore, the robot software architecture has been developed in C++. The left side of Figure 4.4 shows our software architecture described in this section. The right of Figure 4.4 shows the GUI controller of the robot. The robot can either behave autonomously or be controlled by a human with those GUI controllers.



Componet Server	Status
audio	UNKNOWN
ip5000_omni	UNKNOWN
ip5000_zoom	UNKNOWN
pa10-left	UNKNOWN
pa10-right	UNKNOWN
sensor_glove	UNKNOWN
v4l_9eyes	UNKNOWN
v4l_dtm2d	UNKNOWN
v4l_dtm3d	CONNECTED
v4l_zoom	UNKNOWN
viewer	CONNECTED
visca	UNKNOWN
vxworks	CONNECTED

Figure 4.4: Robot Architecture

Chapter 5

Execution assembly tasks

The method proposed above enables the robot to recognize human assembly tasks from observation using robot vision. The goal described in this chapter is that the robot acquires the ability for performing the same assembly tasks from the recognition results. The robot executes a proper sub-skill sequence for performing the tasks.
Many researchers have proposed methods to enable a robot to perform assembly tasks. If each object configuration is obtained precisely and a robot does not cause execution errors at all, a robot can perform assembly tasks using ideal trajectory. But in the real world, various errors preventing the completion of assembly tasks exist; a robot needs to have the ability for recovering errors. So, methods for recovering using contact information have been proposed. The methods are roughly divided into two types, *impedance control* and *force/position hybrid control*. In this paper, we describe the implementation of sub-skills based on a control model such as the force/position hybrid control. First, we describe the control model. Next, we describe the implementation of sub-skills.

5.1 Control model

We implemented sub-skills based on the position control mode of the pa10 library. In this mode, we can control a manipulator by obtaining the desired manipulator configuration.

For example, consider the case of moving the box on the table as shown to the left in Figure 5.1. In this case, we apply position control in a vertical direction and force control in a horizontal direction. Because the stiffness of the fingers is low, it can be considered that a hardware spring exists between the box and a manipulator in a vertical direction. When the value of a force sensor is f and the current manipulator position p, the next manipulator position p' is as shown in the equation (5.1), where s is a proper positive value, k is the coefficient of the spring, and f_d is a desired force value.

$$p' = p + s \begin{pmatrix} 1 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & k \end{pmatrix} \begin{pmatrix} 0 \\ f - f_d \end{pmatrix} \tag{5.1}$$

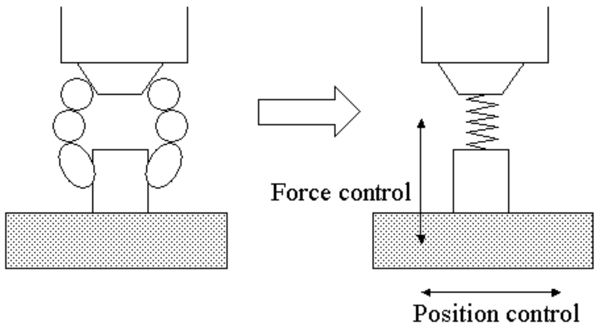


Figure 5.1: Control model

In general, a manipulator configuration pr_t in time t is computed by the equation (5.2), where pi_t is an ideal trajectory in time t, K is a proper 6×6 matrix, and f and f_d are current and desired force/torque values.

$$pr_{t+1} = pr_t + (pi_{t+1} - pi_t) + K(f - f_d) \tag{5.2}$$

5.2 Decision of position or force control

For the assumption that an object cannot be moved maintaining singular contact relations, we consider the method to decide to which applying position or force control in each DOF in non-singular contact relation only.

For moving freely in the maintaining DOFs and not obtaining counter force, it is a mistake to apply force control. Rather, position control is applied in the maintaining DOF.

Because in the detaching or constraining DOFs, position control using ideal trajectories can not be adapted flexibly, force control is applied. The desired force (torque) is set to zero in the constraining DOF, because friction and positioning errors do not occur.

On the other hand, because in the detaching DOF, an object can be easily moved by detaching a contact relation, we need to decide the proper desired force (torque). In translation, a desired force is set as pushing in the reverse direction against the detaching direction. (Show in Figure 5.2.)

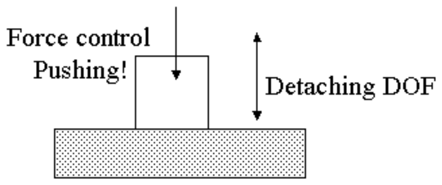


Figure 5.2: Force control strategy in the detaching DOF in translation

In rotation, a desired torque cannot set easily, because of the existing two types of detaching DOFs as follows: (shown in Figure 5.3)

- A rotation axis in the detaching DOF is a type 1 axis.
- A rotation axis in the detaching DOF is a type 2 axis.

In the latter, a desired torque around a proper axis is set to the proper positive value in the same way as in the translation. In the former, a desired torque around a proper axis is set to zero, because a torque around the axis occurs when detaching a contact relation. (Shown in Figure 5.4.)

5.3 Translation sub-skill

For applying the control model, ideal trajectories are needed. In the translation sub-skill, because an object orientation is not changed basically while performing

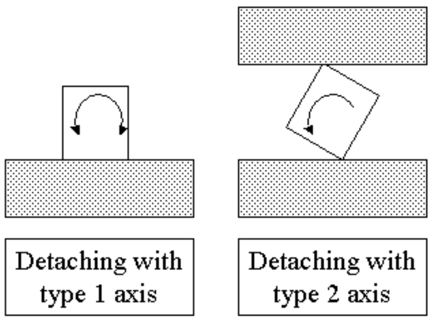


Figure 5.3: Two types of detaching DOFs in rotation

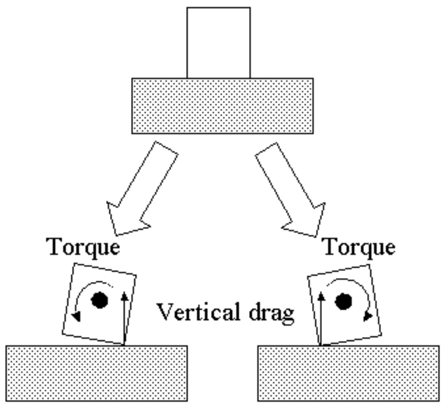


Figure 5.4: Torque in detaching a contact relations

the sub-skill, the ideal trajectory is a line. Because of using subtraction of ideal trajectories only in the control model, we need a line direction only. A line direction ΔT (but $|\Delta T|$ is 1) satisfies the equation (5.3), where m is the number of contacts and n_i is the face normal if an i-th contact is a vertex-face contact or face-vertex contact, and is the face normal including two edges if an i-th contact is an edge-edge contact.

$$A\Delta T = 0 \quad A = \begin{pmatrix} {}^t n_1 \\ \vdots \\ {}^t n_m \end{pmatrix} \tag{5.3}$$

If the rank of the matrix is three, the answer of ΔT is 0 only, so an object cannot be translated. If the rank is two, a line direction is computed by the equation (5.4) where n_i and n_j are independent of each other.

$$\Delta T = \pm \frac{n_i \times n_j}{|n_i \times n_j|} \tag{5.4}$$

When the rank is 1 or 0, consider the simultaneous equality as shown in the equation (5.5), where the number of the contacts happening after performing sub-skill and n'_i is the face normal of an i-th new occurring contacts after the sub-skill.

$$A'\Delta T = 0 \quad A' = \begin{pmatrix} {}^t n_1 \\ \vdots \\ {}^t n_m \\ {}^t n'_1 \\ \vdots \\ {}^t n'_l \end{pmatrix} \tag{5.5}$$

Now, the vector $n'_1, \cdots$, and n'_l can be divided into two types as follows:

- independent of the vector $n_1, \cdots$, and n_m (referred as to *type 1*)
- not independent (referred as to *type2*)

The type 1 vectors are useful for deciding the ideal line direction.

- if rank $A = 0$ and rank $A' = 1$, $\Delta T = \pm n'_1$.
- if rank $A = 1$ and rank $A' = 2$, $\Delta T = \pm \frac{n' - (n_1 \cdot n')n_1}{|n' - (n_1 \cdot n')n_1|}$, where n' is a type 1 vector.

If the rank $A' -$ the rank A is more than one, another transition detection by searching near contact relations mentioned later, or visual feedback is applied.

5.4 Rotation sub-skill

Because motion removing translation is rotation, it is very difficult to treat all rotations. But in assembly tasks performed by humans, rotation is often a limited planar (two DOFs in translation and one DOF in rotation) rotation in which the axis direction is fixed. First, we describe the planar rotation. Next we propose the method for mapping rotation motion performed by humans to planar rotation.

5.4.1 Planar rotation

In the plane, contact relations can be represented as the combination of vertex-edge and edge-vertex contacts. An object cannot rotate maintaining contact relations in which more than two contact elements includes[27]. So, we consider two cases as follows:

- single contact
- two contacts.

Single contact

In the single contact case as shown in Figure 5.5, an object can rotate around a fixed axis on a contact point. In that case, subtraction of the ideal trajectory is decided by a contact point only.

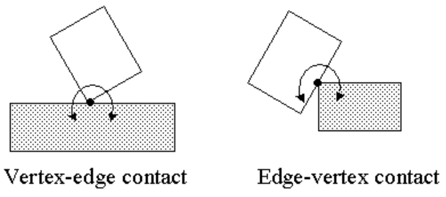


Figure 5.5: Rotation sub-skill: single contact case

Two contacts

In the two contact case as shown in Figure 5.6, an object configuration (x, y, θ) satisfies the equation (5.6) and (5.7), where n is the right angle vector toward the edge, v is a position of vertex, and e is a position of a point on the edge.

$$\text{vertex-edge contact} \quad : \quad n \cdot (Rv + T - e) = 0 \tag{5.6}$$

$$\text{edge-vertex contact} \quad : \quad Rn \cdot (v - Re - T) = 0 \tag{5.7}$$

but, $R = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}, T = \begin{pmatrix} x \\ y \end{pmatrix}$

Because if the value of θ is given, the value of x and y is obtained by solving first order simultaneous equalities, subtraction of ideal trajectories can be obtained.

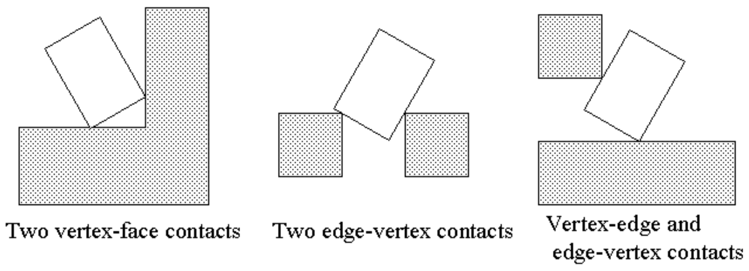


Figure 5.6: Rotation sub-skill: two contact case

5.4.2 Mapping a human performance to planar rotation

As mentioned above, we concentrate on treating only rotation around the axis whose direction is fixed. First, we decide on an axis direction using the screw theory. Using the method for computing rotation DOFs, we obtain the condition about an axis direction as shown in the equation (5.8), where n is a positive value. But in this case, because we consider the maintaining rotation only, all inequalities are converted to equalities[1]

$$A \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} \quad A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ & \vdots & \\ a_{n1} & a_{n2} & a_{n3} \end{pmatrix} \tag{5.8}$$

The answer to these simultaneous equalities can be found by the same way in translation.

Next, because of obtaining an ideal trajectory, the object configuration when maintaining all contacts is formulated using the equation (5.9), (5.10), and (5.11). (Shown in the section 3.4.1)

$$\text{vertex-face contact} \quad : \quad n \cdot (Rv + T - f) = 0 \tag{5.9}$$

[1]In the same contact relation, the obtained equations are different. But the answer includes a direction of a desired rotation axis.

$$\text{face-vertex contact} \quad : \quad Rn \cdot (v - Rf - T) = 0 \tag{5.10}$$

$$\text{edge-edge contact} \quad : \quad (Re_1 \times e_2) \cdot (p_2 - Rp_1 - T) \tag{5.11}$$

$$R = \begin{pmatrix} u^2 + (1-u^2)\cos\theta & uv(1-\cos\theta) - w\sin\theta & uw(1-\cos\theta) + v\sin\theta \\ uv(1-\cos\theta) + w\sin\theta & v^2 + (1-v^2)\cos\theta & vw(1-\cos\theta) - u\sin\theta \\ uw(1-\cos\theta) - v\sin\theta & vw(1-\cos\theta) + u\sin\theta & w^2 + (1-w^2)\cos\theta \end{pmatrix}$$

$T = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$, where the vector (u, v, w) represents the direction of rotation axis.

If the value of θ, u, v, w is given, the value of $x, y,$ and z is obtained by solving obtained simultaneous equations as shown in the equation (5.12), where n is the number of obtained equation.

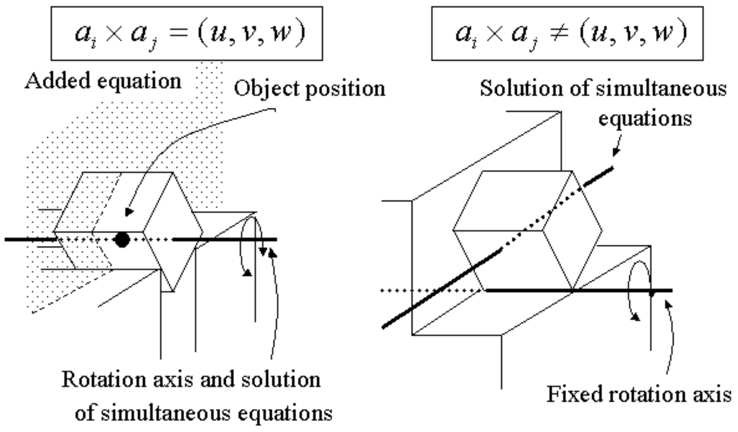
$$AT = \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \tag{5.12}$$

If the simultaneous equations (5.12) cannot be solved, it is impossible that a rotation motion around the axis whose is fixed, and ideal trajectories are solved by the randomized algorithm [16].

If the rank of the matrix A is 3, the answer representing an object configuration on ideal trajectories is only one. If the rank of the matrix A is 2, we assume that a_i, a_j are row elements of the matrix A and a_i is independent of a_j. If $a_i \times a_j = (u, v, w)$, the answer obtained by solving the simultaneous equations added to the equation (5.13), that is only one, considers an object configuration on ideal trajectories, where c is the current object position, along the rotation axis. If $a_i \times a_j \neq (u, v, w)$, an object can be rotated around a fixed axis. (Shown in Figure 5.7)

$$(a_i \times a_j) \cdot (T - c) = 0 \tag{5.13}$$

If the rank of the matrix A is 1, an object can rotate around a fixed axis. The axis is probably obtained from the axis direction and the contact point, because humans perform planar rotation motions. The answer obtained by solving the simultaneous equations added to the equation (5.14) and (5.15), where c_1, c_2 are



$a_i \times a_j = (u,v,w)$
Added equation
Object position
Rotation axis and solution of simultaneous equations
$a_i \times a_j \neq (u,v,w)$
Solution of simultaneous equations
Fixed rotation axis

Figure 5.7: Ideal trajectories

the vectors satisfying that $c_1 \cdot a = 0, c_2 \cdot a = 0, c_1 \cdot c_2 = 0$, supposing a is a row element of the matrix.

$$c_1 \cdot (T - c) = 0 \tag{5.14}$$

$$c_2 \cdot (T - c) = 0 \tag{5.15}$$

5.5 Make-contact sub-skill

A make-contact sub-skill is ended when an object cannot translate (rotate) in the aimed direction. In short, when the counter force (torque) is more than a proper threshold value, a make-contact sub-skill is ended.

5.6 Slide sub-skill

A slide sub-skill is ended when some singular contact occurs. Because a vertex-face, face-vertex, or edge-edge contact is changed to convex vertex-convex edge or two convex-vertices contacts, at the end of the sub-skill, the vertical drag generates the face or the face including two edges gets smaller. So, when a proper selected vertical drag is less than a proper threshold value, a slide sub-skill is ended.

5.7 For more robust sub-skills

The proposed control model enables the robot to perform assembly tasks robustly. But lack of efficiency, for example friction, sometimes prevents the completion of assembly tasks. The ability for detecting errors aids in completion First, we discuss the cause of the failure. Next we propose the method to detect errors.

5.7.1 Cause of failure

The main execution error is not maintaining an aimed contact relation. In short, it happens that an object moves in the direction of detaching DOFs. Conversely, we can detect unexpected contact transitions happening because of execution errors, or by moving an object virtually in the direction of detaching DOFs.

For example, consider the planar two contact case as shown in Figure 5.8. The left contact relation in Figure 5.8 is an original. The possible motion of this contact relation is represented as two inequalities, $f_1 \geq 0$ and $f_2 \geq 0$, in the screw representation. The solution region satisfies simultaneous equations; $f_1 = 0$ and $f_2 = 0$, represents a situation in which an object maintains an original contact relation. But the solution region satisfies simultaneous equations, $f_1 0$ and $f_2 = 0$, and represents the situation where an object maintains the contact corresponding to the equation f_1 and detaches the contact corresponding to the equation f_2. The solution region satisfies simultaneous equations, $f_1 = 0$ and $f_2 0$, and represents the situation where an object maintains the contact corresponding to the equation f_2, and detaches the contact corresponding to the equation f_1. Because the above two solution regions are not vacuous, we can detect two new contact relations. Conversely, by deciding whether the solution region is vacant, we can detect the cause of failure before execution.

The algorithm is as follows:

1. represent an original contact relation using the screw theory.

2. apply the motion DOF analyzer for obtaining detaching DOFs to that contact relation. As a result, we obtain the simultaneous inequalities (5.16), where

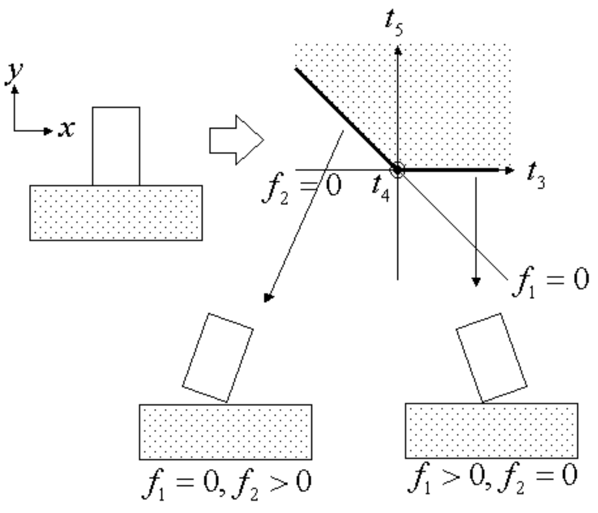


Figure 5.8: Contact relations happening because of execution error

b_i, c_i is a 1×6 matrix.

$$\begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} \begin{pmatrix} t_1 \\ \vdots \\ t_6 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix} \begin{pmatrix} t_1 \\ \vdots \\ t_6 \end{pmatrix} \geq \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \tag{5.16}$$

3. decide that the possible solution region of the simultaneous inequality (5.17) for all H, where H is a subset of $\{1, 2, \cdots, m\}$, l is the number of elements including H, H' is a set satisfying $H \bigcup H' = \{1, 2, \cdots, m\}, H \bigcap H' = \emptyset$, h_i, h'_i are i-th element values.

$$\begin{pmatrix} b_1 \\ \vdots \\ b_n \\ c_{h_1} \\ \vdots \\ c_{h_l} \end{pmatrix} \begin{pmatrix} t_1 \\ \vdots \\ t_6 \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} c_{h'_1} \\ \vdots \\ c_{h'_{l-m}} \end{pmatrix} \begin{pmatrix} t_1 \\ \vdots \\ t_6 \end{pmatrix} \geq \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix} \tag{5.17}$$

Given the simultaneous inequalities (5.16), the solution region is vacuous, if $\exists i, (1 \geq i \geq l - m)$, $c_{h'_i}$ is not independent of $b_1, \cdots, b_n, c_{h_1}, \cdots, c_{h_m}$.

5.7.2 Detecting errors and class 2 sub-skills

Basically, a robot can detect errors by means of a Force/Torque sensor. In short, if the value of force (torque) in the direction applying the force control is less than a proper threshold, a robot detects errors. But in the slide sub-skill, a robot does not decide that the force value gets less than proper threshold means the end of sub-skill or errors. (shown in Figure 5.9.)

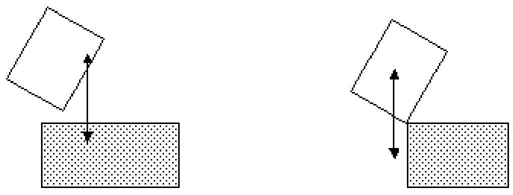


Figure 5.9: The end of sub-skill or error?

This situation corresponds to a class 2 sub-skill. A first solution is using visual feedback. A second solution is to try to perform the next sub-skill. Because the end of a slide sub-skill is the start of a next sub-skill, if the robot cannot continue the next class 1 sub-skill, the robot decide that a previous slide sub-skill is not finished.

Chapter 6

Experiments

We apply this system to the peg-in-hole assembly task. First, we perform the assembly task in front of a nine eye stereo system. The system obtains 2D and 3D data as shown in Figure 6.1.

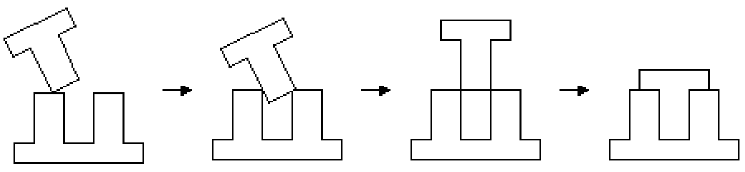


Figure 6.1: The peg in hole assembly task

Applying the image processing, the robot obtains the trajectories of the peg and the configuration of the hole before removing the vision error as shown in Figure 6.2. After removal, the robot obtains correct trajectories as shown in Figure 6.3
The robot applies the task analyzer to the transitions of contact relations obtained by the trajectories and 3D CAD models of the peg and the hole. As a result, the robot recognizes the assembly tasks as shown in Figure 6.4
Using the result, the robot plans a proper sequence of sub-skills as shown in Figure 6.5
Finally, following the plan, the robot executes as shown in Figure 6.6.

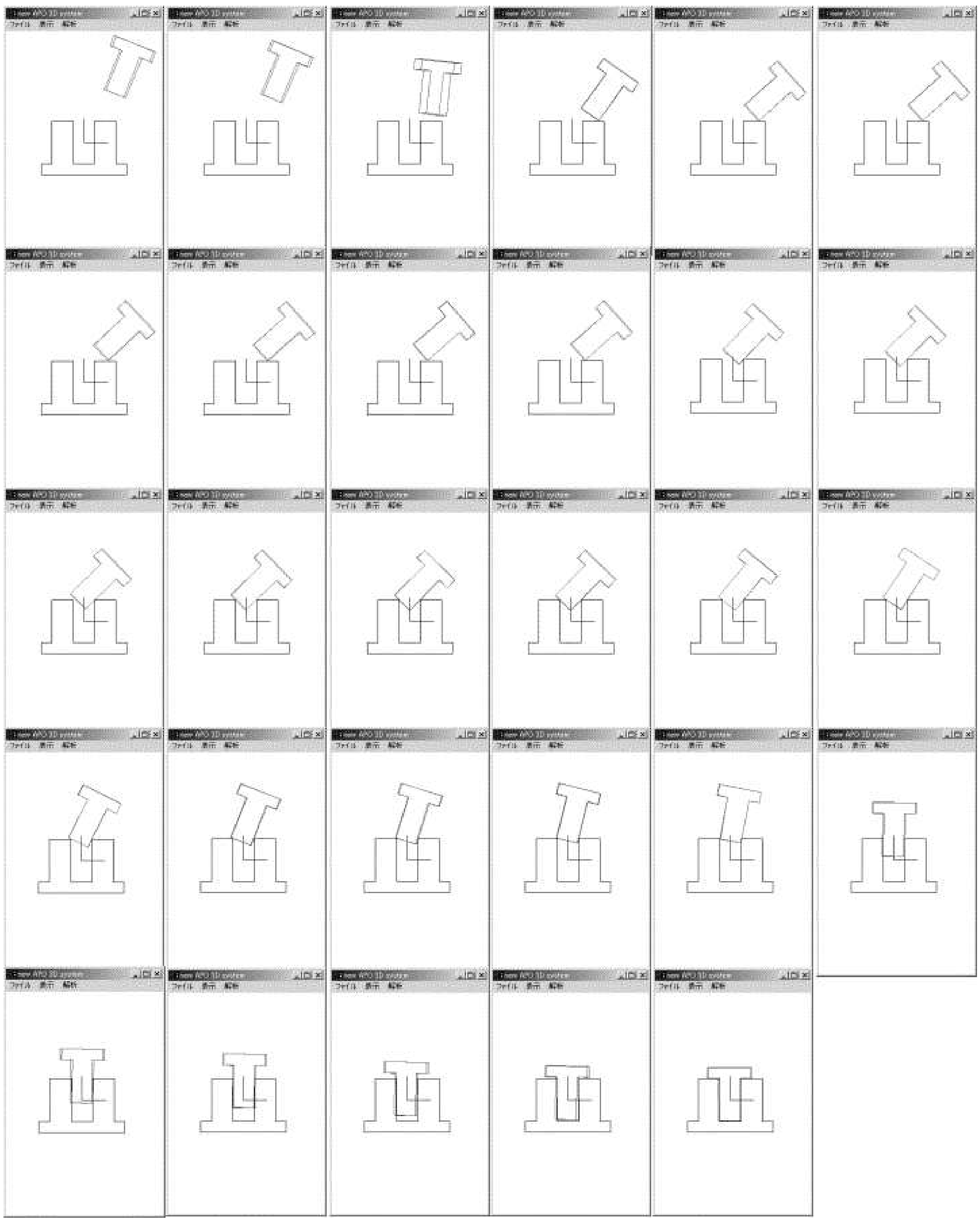


Figure 6.2: The trajectories of the peg including some errors

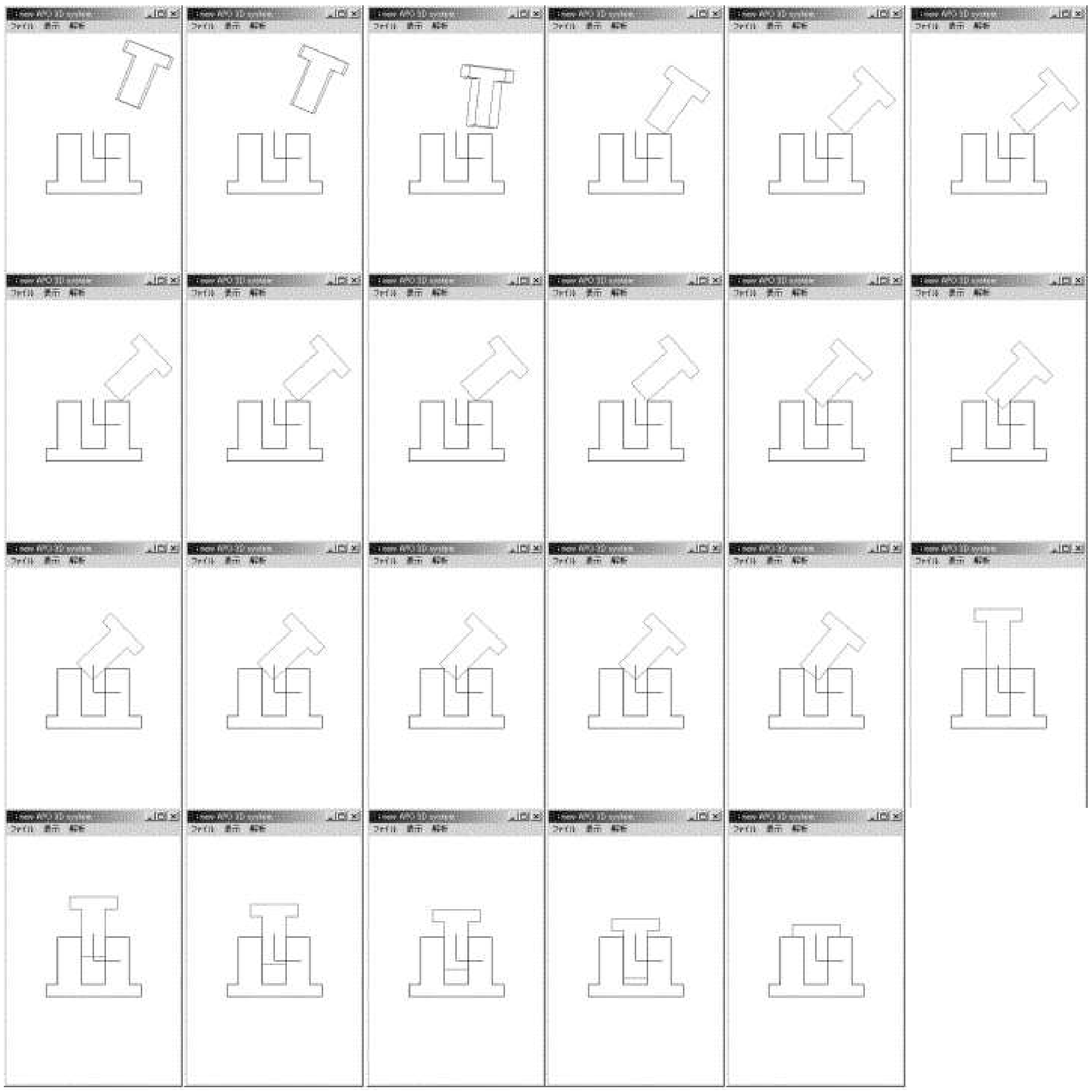


Figure 6.3: The correct trajectories of the peg

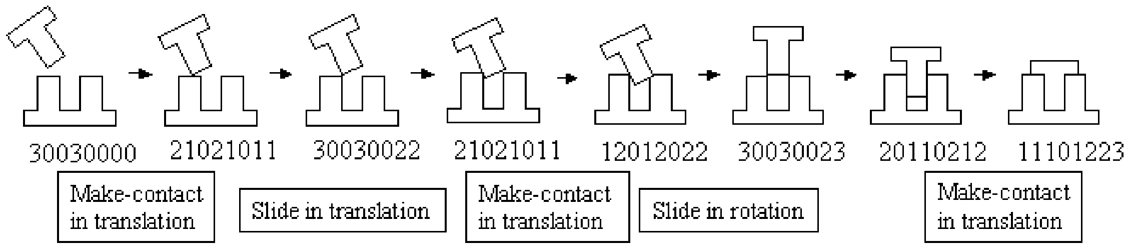


Figure 6.4: Result of the assembly task recognition

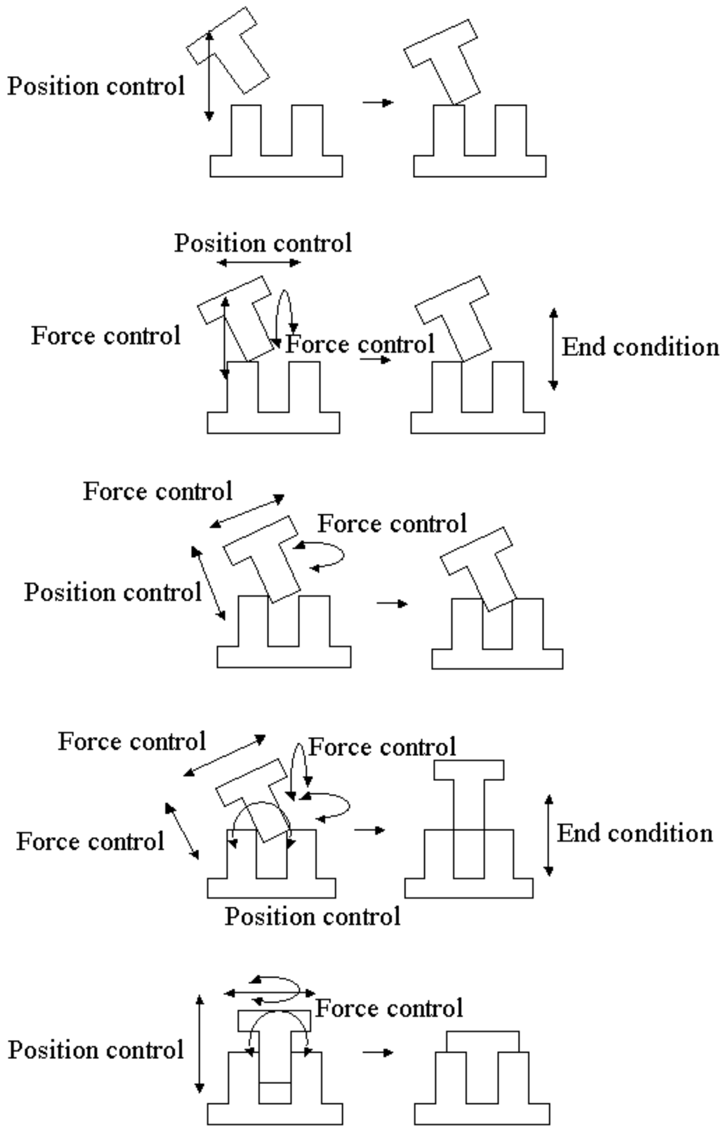


Figure 6.5: Task plan

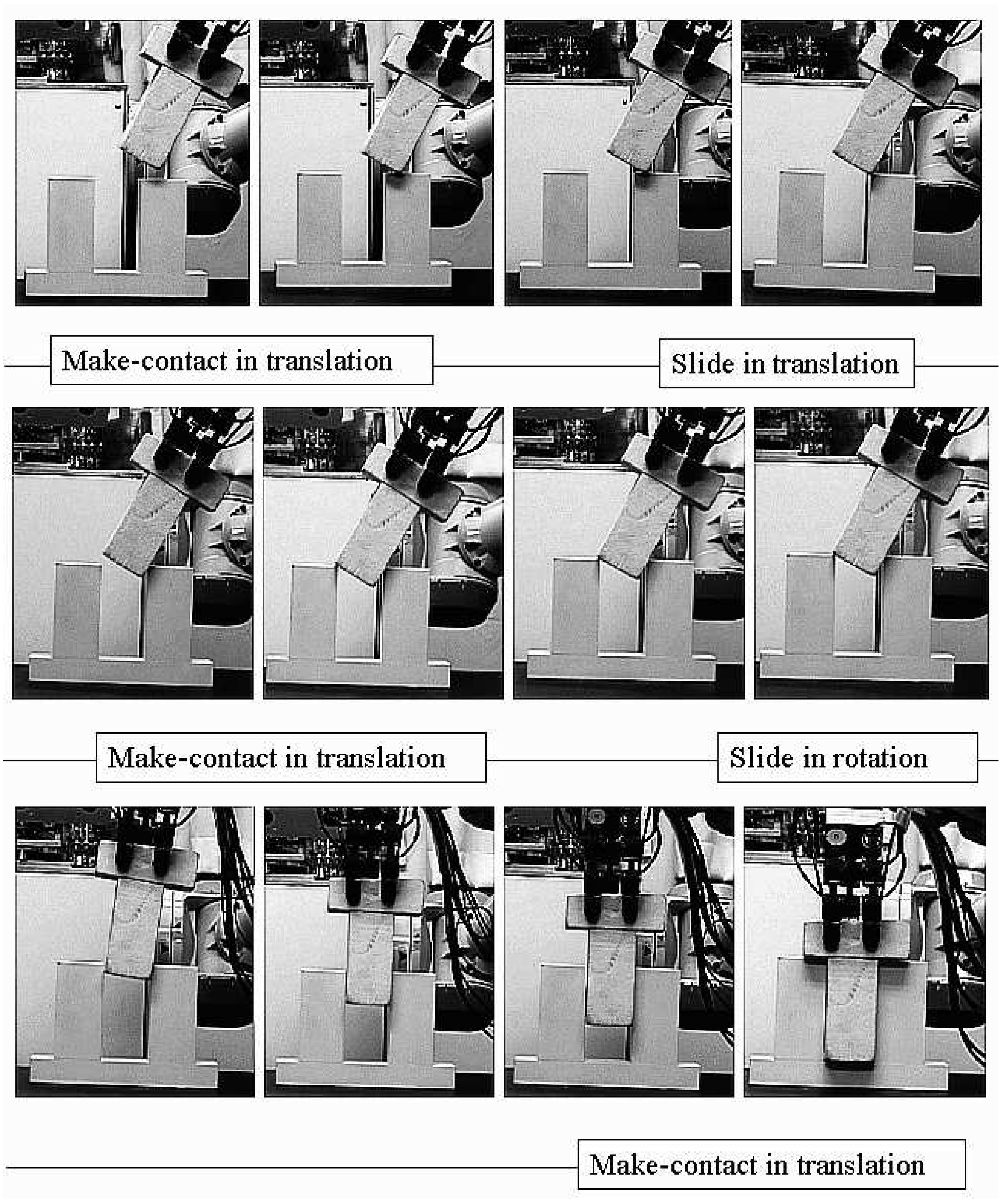

Make-contact in translation
Slide in translation
Make-contact in translation
Slide in rotation
Make-contact in translation

Figure 6.6: Task execution

Chapter 7

Conclusions and future work

7.1 Conclusions

We proposed a system for recognizing and executing assembly tasks from observation automatically.
First, we proposed the observation system. Though other observation systems can obtain the trajectories of objects, our system can remove the observation errors, using local contact relations and possible transitions between them. As a result, the robot can acquire better eyes.
Next, we proposed the recognition system. Though other recognition systems output only ideal transitions or transitions of contact relations, our system analyzes assembly tasks using a notion of motion DOFs. Our system has the *true* ability to recognize general assembly tasks.
Finally, we proposed the execution system. Though in other researches, making the program for execution needs an excellent human programmer, in our system, the implementation of sub-skills is required. Once sub-skills are implemented, the execution system can make the program perform assembly tasks automatically.
Furthermore, the proposed integrated system from observation to execution through recognition has yielded impressive results. In the assembly task field, the robot has abilities nearly equivalent to those of humans, we believe.

7.2 Future work

In the execution module, our system has several problems. First, we have not yet implemented torque and visual feedback. Next, we have not yet established the method to recover when a not-planning contact relation happens while execution. In this experiment, we selected the transitions happening that merely[1]. We would like to improve the sub-skill implementation.
In this paper, we refer to a necessary assembly operation as *sub-skill*, because an assembly skill, for example, a whole peg-in-hole operation, can be composed of a combination of sub-skills. So, we would like to make the robot perform assembly tasks composed of more steps by using skill sub-skill cooperation.
In this paper, we assumed several limitations as follows:

- An object is rigid and polyhedral.
- Only one grasping object is permitted to move.

By relaxation of these limits, we would like to make the robot to acquire various human actions.

[1]You may wonder why we make the robot perform the *unnatural* peg-in-hole operation. But for blind people, the peg-in-hole operation is natural. Blind people select certain operations without using visual feedback; it is interesting that the robot with poorer vision than people performs the peg-in-hole operation more successfully

References

[1] K. Ikeuchi and Takashi Suehiro, "Toward an Assembly Plan from Observation Part I: Task Recognition with Polyhedral Objects", IEEE Transactions on Robotics and Automation, Vol 10, No. 3, June, 1994.

[2] Takashi Suehiro and Katsushi Ikeuchi, "Towards an Assembly Plan from Observation: Part II: Correction of Motion Parameters based on Fact Contact Constraints", IEEE International Conference on Intelligent Robots and Systems, pp. 2095 - 2102, July 7 - 10, 1992.

[3] Jun Miura and Katsushi Ikeuchi, "Task-Oriented Generation of Visual Sensing Strategies in Assembly Tasks", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20, No.2, February, 1998.

[4] Mark. D. Wheeler, "Sensor Modeling, Probabilistic Hypothesis Generation, and Robust Localization for Object Recognition", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. 28, pp 293 - 331, 1986.

[5] Kentaro Kawamura, Mark D. Wheeler, Osamu Yamashita, Yoichi Sato, and Katsushi Ikeuchi, "The Method of Object Localization in Electric Distribution Systems by Using 3D Mesh Models and M-estimation", Journal of the Robotics Society of Japan, Vol. 18, No. 4, pp. 600 - 611, 2000 (in Japanese).

[6] Hiromu Onda, Tsukasa Ogasawara, Hirohisa Hirukawa, Kosei Kitagaki, Akira Nakamura, and Hideo Tsukune, "A Telerobotics System using Planning Functions Based on Manipulation Skills and Teaching-by-

Demonstration Technique in VR", Journal of the Robotics Society of Japan, Vol. 18, No. 7, pp. 979 - 994, 2000 (in Japanese).

[7] Masayuki Tsuda, Tomoichi Takahashi, and Hiroyuki Ogata, "Generation of an Assembly-Task Model Analyzing Human Demonstration", Journal of the Robotics Society of Japan, Vol. 18, No. 4, pp. 535 - 544, 2000 (in Japanese).

[8] Masayuki Tsuda, Hiroyuki Ogata, and Yoshito Nanjo, "Creating Groups of a Local Contact-State Transition Model for a Robotic Assembly Task using Human Demonstration", Journal of the Robotics Society of Japan, Vol. 18, No.4 pp. 545 - 554, 2000 (in Japanese).

[9] Takeshi Matsuoka, Tsutomu Hasegawa, Kyuhei Honda, and Toshihiro Kiriki, "Manipulation System by Multi-Fingered Hand Based on Task Observation and Task Evaluation", Journal of the Robotics Society of Japan, Vol. 17, No. 5, pp. 696 - 703, 1999 (in Japanese).

[10] Shinichi Hirai, Haruhiko Asada, and Hidekatsu Tokumaru, "Kinematic Analysis of Contact State Transitions in Assembly Operations and Automatic Generation Of Transition Network", Journal of the Society of Instrument and Control Engineers, Vol. 24, No. 4, April, 1988 (in Japanese).

[11] Yasuyoshi Yokokohji, "Classification of Contact States and Planning of Assembly Sequence", Journal of the Robotics Society of Japan, Vol. 11, No. 2, pp. 185 - 191, 1993.

[12] Yong Yu and Tsuneo Yoshikawa, "Evaluation of Contact Stability between Objects", Journal of the Robotics Society of Japan, Vol. 18, No. 7, pp. 1026 - 1033, 2000 (in Japanese).

[13] Tomas Lozano-Perez, Matthew T. Mason, and Russell H. Taylor, "Automatic Synthesis of Fine-Motion Strategies for Robots", The International Journal of Robotics Research, Vol. 3, No. 1, Spring, 1984.

[14] Brenan J. McCarragher and Haruhiko Asada, “A Discrete Event Approach to the Control of Robotic Assembly Tasks”, IEEE International Conference on Robotics and Automation, pp 331 - 336, 1997.

[15] Jing Xiao and Xuerong Ji, “A Divide-and-Merge Approach to Automatic Generation of Contact States and Planning of Contact Motion”, IEEE International Conference on Robotics and Automation, pp 750 - 756, April, 2000.

[16] Xuerong Ji and Jing Xiao, “Towards Random Sampling with Contact Constraints”, IEEE International Conference on Robotics and Automation, pp 1390 - 1395, April, 2000.

[17] Q. J. Ge and J. M. McCarthy, “An Algebraic Formulation of Configuration-Space Obstacles for Spatial Robots”, IEEE Conference on Robotics and Automation, pp. 1542 - 1547, 1990.

[18] Q. Ge and J. M. McCarthy, “Equations for Boundaries of Joint Obstacles for Planar Robots”, IEEE International Conference on Robotics and Automation, Vol. 1, pp 164 - 169, 1989.

[19] M. S. Ohwovoriole and B. Roth, “An Extension of Screw Theory”, Journal of Mechanical Design, Vol. 103, pp 725 - 735, October, 1981.

[20] H. W. Kuhn and A. W. Tucker, “Linear Inequalities and Related Systems”, Annals. of Mathematics Studies, Vol. 38, 1956

[21] Jun Takamatsu, Hirohisa Tominaga, Koichi Ogawara, Hiroshi Kimura, and Katsushi Ikeuchi, “Extracting Manipulation Skills from Observation”, IEEE International Conference on Intelligent Robots and Systems, Vol. 1, pp. 584 - 589, 2000.

[22] Hirohisa Tominaga, Jun Takamatsu, Koichi Ogawara, Hiroshi Kimura, and Katsushi Ikeuchi, “Symbolic representation of trajectories for Skill Generation”, IEEE International Conference on Robotics and Automation, Vol. 4, pp. 4077 - 4082, 2000.

[23] Jun Takamatsu, Hiroshi Kimura, and Katsushi Ikeuchi, “Classifying Contact States fro Recognizing Human Assembly Tasks”, IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, pp. 177 - 182, August, 1999.

[24] Shinichi Hirai, “Kinematics and Statics of Manipulation Using the Theory of Polyhedral Convex Cones and Their Application to the Planning of Manipulative Operations”, Journal of the Robotics Society of Japan, Vol. 17, pp. 68 -83, 1999 (in Japanese).

[25] Hirohisa Hirukawa, Toshiro Matsui, and Kunikatsu Takase, “A Fast Algorithm for the Analysis of the Constraint for Motion of Polyhedra in Contact and its Application to Departure Motion Planning”, Journal of the Robotics Society of Japan, Vol. 9, No. 7, pp 841 - 848, 1991.

[26] George V. Paul and Katsushi Ikeuchi, “A Quasi-Linear Method for Computing and Projecting onto C-Surfaces: Planar Case”, IEEE International Conference on Robotics and Automation, pp 2032 - 2039, April, 1997.

[27] Jing Xiao and Lizin Zhang, “Contact Constraint Analysis and Determination of Geometrically Valid Contact Formations from Possible Contact Primitives”, IEEE Transactions on Robotics and Automation, Vol. 13, No. 3, June, 19997.

[28] Yutaka Ishiyama, Yasushi Sumi, and Fumiaki Tomita, “3-D Motion Tracking of 3-D Objects Using Stereo Vision”, Journal of the Robotics Society of Japan, Vol. 18, No. 2, pp. 213 - 200, 2000 (in Japanese).

[29] M. Oshima, “Studies of Object Recognition Using three-Dimensional Information”, Researches of the Electrotechnical Laboratory, Vol. 826, July, 1982 (in Japanese).

[30] David Johnston and Jing Xiao, “On Relating the Disconnectedness of a Contact Formation to the Geometric Properties of its Constituent Objects”, IEEE International Conference on Robotics and Automation, pp. 2284 - 2289, April, 2000.

[31] Kosei Kitagaki, Takashi Suehiro, and Tsukasa Ogasawara, “Monitoring of a Pseudo Contact Point for Fine Manipulation”, IEEE International Conference on Intelligent Robots and Systems, pp. 757 - 762, 1996.

[32] Takashi Suehiro, “Study on Advanced Manipulation Systems”, Researches of Electrotechnical Laboratory, Vol. 912, June, 1990 (in Japanese).

[33] Takashi Suehiro and Kunikatsu Takase, “Representation and Control of Motion in Contact and Its Application to Assembly Tasks”, Proceedings of International Symposium on Robotics Research, pp. 367 - 374, 1989.

[34] Matthew T. Mason, “Compliance and Force Control for Computer Controlled Manipulators”, IEEE Transactions on Systems, Man, And Cybernetics, Vol. SMC-11, No. 6, June, 1981.

[35] Sung C. Kang, Yong K. Hwang, Mun S. Kim, Chong W. Lee, and Kyo-Il Lee, “A Compliant Motion Control for Insertion of Complex Shaped Objects using Contact”, IEEE International Conference on Robotics and Automation, pp. 841 - 846, April, 1997.

[36] D. E. Whitney, “Quasi-Static Assembly of Compliantly Supported Rigid Parts”, Journal of Dynamic Systems, Measurement, and Control, Vol. 104, pp. 65 - 77, March, 1982.

[37] Stefano Chiaverini and Lorenzo Sciavicco, “The Parallel Approach to Force/Position Control of Robotic Manipulators”, IEEE Transactions on Robotics and Automation, Vol. 9, No. 4, August, 1993.

[38] S. Kimura, T. Shinbo, E. Kawamura, H. Yamaguchi, and K. Nakano, “A New Real-Time Stereo Vision System Using 2D Convolvers”, Technical Report of IEICE, Vol. 97, No. 501, PRMU97-207, pp. 1- 8, January, 1998.

[39] http://www.komatsu.co.jp/research/study56.htm

[40] Common Object Request Broker Architecture, OMG, July, 1995.

[41] http://www.cs.wustl.edu/ schmidt/TAO.html

[42] S. Hirose and S. Amano, “The VUTON: High Payload High Efficiency Holonomic Omni-Directional Vehicle,” Proceedings of International Symposium on Robotics Research, pp.253-260, 1993.