Towards an Assembly Plan from Observation:
Fine localization based on face contact constraints

Takashi Suehiro Katsushi Ikeuchi

August 1991
CMU-CS-91-168

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

This research was sponsored by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson AFB, Ohio 45433-6543 under Contract F33615-90-C-1465, ARPA Order No. 7597.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. government.

Keywords: assembly plan, robot programming, teach by showing, teleoperation, object recognition, uncertainty, hand-eye, grasping

Abstract

We have been developing a novel method to program a robot, an APO (assembly-plan-from-observation) method. A human performs assembly operations in front of the APO system's TV camera. The APO system recognizes such assembly operations and generates an assembly plan to repeat the assembly operations using its robot arm.

Since an important purpose of assembly operations is to achieve face contacts between objects, our previous system was based on face contact relations. Our system recognized object configurations after each of assembly operation and then extracted face contact relations from the observed object configurations. By associating a face contact relation with the operation necessary to achieve the relation, the system was able to construct a plan to repeat the assembly.

In this system, two kinds of information were extracted from object configurations: 1) face-contact relation and 2) motion parameters necessary to move objects around. The system works well while noise-free input. Usually, however, object configurations contain some degree of error.

Since a face contact relation is a topological relation, it can be obtained reasonably well from noisy object recognition results. However, motion parameters are obtained directly by converting object configurations. Under the presence of error, due to error-contaminated motion parameters, the system may fail to perform an assembly operation, although a face-contact relation and thus an assembly operation is correctly recovered.

This paper proposes a method to correct erroneous motion parameters based on a face contact relation. We assume that a given face contact relation reflects the actual face contact correctly. Face contact constraint equations will be defined for each pair of contact faces. A face contact equation requires that a vertex of one face is on the plane including the other face. Motion parameters are determined by solving face contact constraint equations simultaneously using the singular value decomposition method.

In order to maintain the relationship among motion parameters of previous operations, we define operation dependency lists (ODL), symbolic lists of homogeneous transformations to represent operations. An ODL is calculated for each assembly operation, and is attached to each object. By using ODLs instead of object configurations, we can apply the same method to obtain correct motion parameters for former operations.

We implement this method in the APO system, apply the method to several assembly examples, and verify the effectiveness of the method.

Contents

1 Introduction

Several methods to program a robot have been proposed. Such methods include the following: teach-by-showing, teleoperation [15, 13, 6], textual programming[5], and automatic programming [8, 12, 11].

Among these four representative methods, teleoperation and automatic programming are the most promising. The automatic programming method aims to develop techniques to make a program automatically from geometric models by geometric reasoning. The teleoperation method requires master and slave manipulators. First, a human operator uses the master manipulator to perform a task; then the master manipulator's control signals are fed into the slave's and the same task is performed. Both of these methods are promising because they do not require expert programmers.

Yet, these methods are often inconvenient and impractical. For example, an automatic programming system often has to consider an infinite number of possible operations in order to determine the appropriate operation. It often occurs that the system has to solve highly non-linear equations. A teleoperation system typically does not possess any geometric reasoning mechanism. Due to this, the system cannot adjust erroneous human input. Also a very minor change of a program requires complete re-programming.

To remedy these problems, we have proposed a novel method which combines the automatic programming and teleoperation. We added a vision capability, observing human operations, to an automatic programming system. In this paradigm, since the approximated parameters are obtained from observation, the system can convert highly non-linear equations into linear equations around the rough estimation from observation. By observing human operations, the system can obtain a rough idea for choosing the appropriate operation from the possible operations. Since the APO system can clean up erroneous input data using the geometric reasoning capability, the APO method is more stable than teleoperation. The system generates not a numerical control signal, but a symbolic assembly plan. Thus, a minor change can be achieved by replacing a corresponding part in the symbolic assembly plan.

In particular, we proposed a system to observe a human performing assembly tasks, and a geometric reasoner to analyze and recognize such tasks from observation, and to generate the same assembly sequence for a robot. We will refer this paradigm as Assembly Plan from Observation (APO).

The central issue in achieving such an APO system is the type of representation

which will be used for describing an assembly task. The main purpose of an assembly task is to put together two separate parts into one subassembled part. By such an assembly task, one particular class of face contact relation is established between the two parts. Thus, we have decided to use face contact relations as the basic representation in the previous system.

We have constructed a procedure tree which relates face contacts with the necessary assembly operations for achieving them. A vision system recognizes objects and builds up object representations in a geometric model. Then, the system extracts face contact relations from the object representations. By consulting the procedure tree with the extracted contact relations, the system can infer human assembly operations. The system also collects motion parameters necessary to complete the assembly operations from the object representations.

In the previous system, the APO system extracts the following two kinds of information from object representations:

- *Topological face-contact relation-* Determining the face contact relations from the configurations of faces given by object configurations and then consulting the procedure tree to identify the operation such as "move-to-contact".

- *Numerical motion parameters-* Determining the motion parameters from the object configurations to complete the assembly task command such as "move-to-contact from $(0,0,0,0,0,0)$ roughly until $(1,1,1,0,0,0)$".

Our system works well when a vision system provides accurate object configurations. Usually, however, object configurations obtained by a vision system contain some degree of error, because human operation is not accurate enough and because vision systems introduce some positional errors in its recognition results.

Among the previous two kinds of information, the topological face contact relation can be extracted correctly under the presence of noise. The APO determines face contact relations through examining face equations of candidate faces. Although these face equations contain some degree of error, we can obtain correct face contact relations by using some approximated threshold values and evaluating the small errors as zeros.

On the other hand, the numerical motion parameters are affected directly by the presence of noise. If the APO system uses noisy motion parameters, it fails to achieve an assembly operation.

In this paper, we assume that the face contact relations given by the APO system reflect the actual face contacts between objects; the recovered topological relations are correct. By using these correct face contact relations, we will consider how to correct motion parameters.

This problem is related to the much larger problem: how to determine object configurations from observation while maintaining specified relation among objects. Several methods [10, 9, 4] have been reported to determine object configurations by matching its geometric model with observed data in the context of object recognition. They mainly, however, determine a single object configuration by least square fitting between observed features and model features. Lowe determines consistent feature (face or edge) configurations from observation in the context of object representation. He focuses on fitting these features to observed data; he does not consider the problem of fitting features while maintaining internal relations among features. Smith and Cheeseman consider the propagation of uncertainty through a relational graph among objects [14]. However, this technique focuses on propagation of uncertainty, and does not consider adjusting object configurations based on known relations.

Durrant-Whyte [2]'s problem is the most similar to ours. He considers how to update object positions in the world model under several sensor observations while maintaining relations among objects. Each object is connected with a spring to other objects; he focuses on finding equilibrium conditions in the network of objects. Since in our problem, each object is related with non-linear redundant relations, we have to solve such difficult non-linear redundant equations to determine the configurations of all objects.

In the context of assembly planning, Barrow and Popplestone tried to determine object configurations based on face contact relations [1]. However, no effective solutions have been reported.

Section 2 develops the basic face contact equations. Then, the section explains why we cannot solve the basic equations directly. Finally, we will explain how to remedy this problem by introducing information given by observations. Then, the section discusses the difference between correction in motion parameters and correction in object configurations. Finally, section 4 shows the implementation of the system.

2 Face Contact Constraints

2.1 Problem Statements

Let us suppose a human operator puts object A on object B. From observation, our system generates the internal geometric representations of objects A and B. In the real world, these two objects contact each other at face Fa and face Fb. Due to the error of observation, a gap between the faces occurs in the internal geometric representation as shown in Figure 1.

This paper assumes that the face contract relation, face Fa on face Fb, is determined correctly [1]. Under this assumption, this paper aims to adjust the object configuration in the internal geometric representations so that the two objects contact each other through face Fa and Fb.

2.2 Face contact equation

Mathematically, the gap may be represented as the distance between one face and one vertex of the other face. Part Fc denotes the projection of face Fa onto face Fb in Figure 1. The distance between one of the vertices of Fc and the plane including Fa is,

$$e = \mathbf{n} T_a^{-1} T_b \mathbf{v}, \tag{1}$$

where T_a and T_b are the homogeneous transformations of objects A and B with respect to the world coordinate system. $\mathbf{n} = (n_x, n_y, n_z, -d)$ is the face equation of Fa with respect to the object A's coordinate system. $\mathbf{v} = (x, y, z)^t$ is the position vector of the vertex with respect to the object B.

If the two objects contact each other, the gap does not exist. The distance is zero. Namely,

$$e = 0. \tag{2}$$

The face contact condition is that at least three vertices satisfy these equations. This condition and Eq. 1 are referred to as *face contact constraint* and *face contact equation*, respectively.

[1] This is possible, because the gap itself is small. By setting up a threshold value for considering two faces contacted to each other, the system can recover the face contract relation correctly

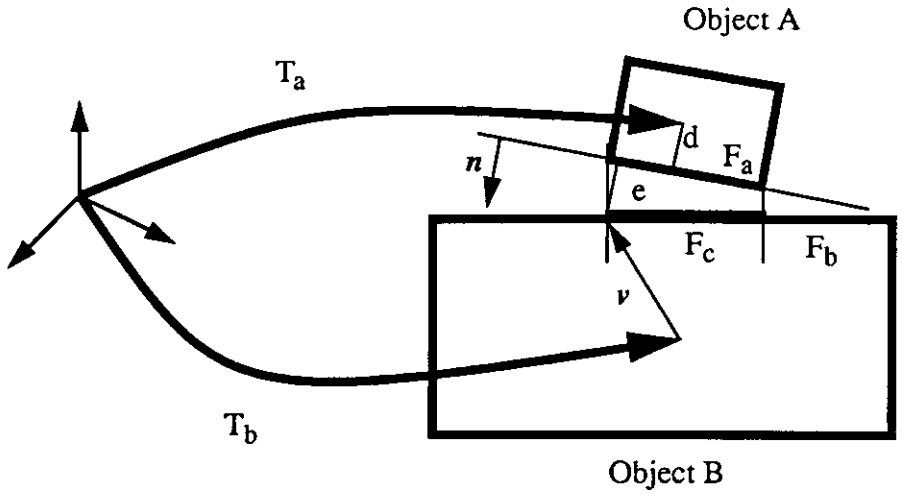


Figure 1: Face Contact

Basically, we will determine the homogeneous transformation, *Ta*, so that the face contact constraint holds at the contact face. This problem is not as simple as it appears due to the following two reasons:

Redundant degrees of freedom

In the example of Figure 1, the position of object *A* and its rotation about the axis perpendicular to the paper cannot be determined uniquely by the face contact constraint. Namely, the face contact equations contain redundant degrees of freedom.

Simultaneous non-linear equations

Let us consider the example in Figure 2. The objects contact each other in a cyclic manner. We have to simultaneously solve the face contact equations given by *A-B*, *B-C*, *C-A*, *B-D*, and *C-D*. Moreover, as shown in Equation 1, each equation is non linear. We have to solve these non-linear simultaneous equations.

The reminder of this section will consider how to solve these two problems using approximate solutions derived from observation.

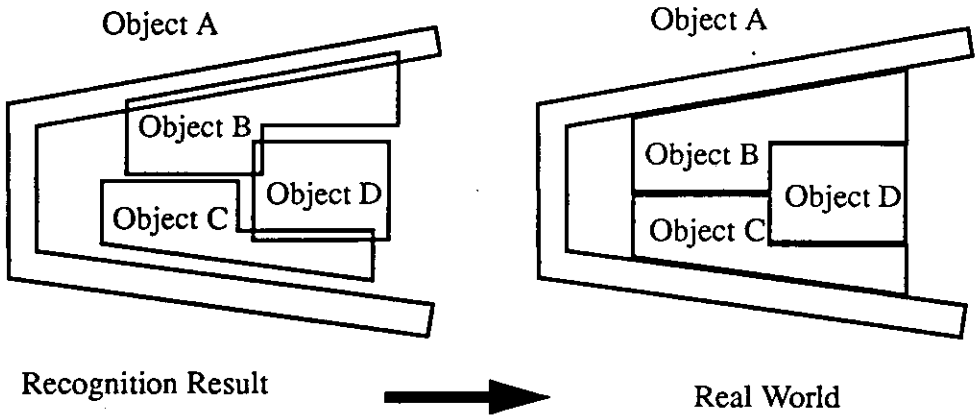


Figure 2: Example of simultaneous constraints

2.3 Simultaneous non-linear equations

Generally speaking, simultaneous non-linear equations can be solved iteratively by various methods. Usually, however, due to local minimum solutions, obtaining real solutions is very difficult. In our system, observation provides approximate solutions. Thus, from these solutions, we will solve the equations by the Newton-Raphson method. Namely, we will linearize the non-linear equations around the approximate solutions and then solve simultaneous linearized face contact equations.

The homogeneous transformation matrix T in Eq. 1 is divided into the constant part, T, corresponding to the approximate configuration derived by observation, and the unknown variable part, ΔT . The variable matrix, ΔT is expressed by six parameters, that is, translation x, y, z and rotation about x, y, z-axis, α(yaw), β(pitch), γ(roll). Then Eq. 1 becomes

$$e = \mathbf{n}\Delta T_a^{-1} T_b^{-1} T_b \Delta T_b \mathbf{v}, \tag{3}$$

where

$$\Delta T_k = P(x_k, y_y, z_k) R_x(\alpha_k) R_y(\beta_k) R_z(\gamma_k), \tag{4}$$

$$P(x,y,z) = \begin{pmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{5}$$

$$R_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha & -sin\alpha & 0 \\ 0 & sin\alpha & cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{6}$$

$$R_y(\beta) = \begin{pmatrix} cos\beta & 0 & sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -sin\beta & 0 & cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{7}$$

$$R_z(\gamma) = \begin{pmatrix} cos\gamma & -sin\gamma & 0 & 0 \\ sin\gamma & cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{8}$$

When $x = y = z = \alpha = \beta = \gamma = 0$, ΔT is a unit matrix.

Simultaneous equations combining Eq. 3 for all related vertices are

$$\mathbf{f(q)} = \mathbf{e}, \tag{9}$$

where $\mathbf{e}$ is the error at all vertices; $\mathbf{e} = (e_1, \ldots, e_i, \ldots, e_m)$, m is the number of vertices, $\mathbf{q}$ is the generalized coordinate system of necessary adjustments; $\mathbf{q} = (q_1, \ldots, q_j, \ldots, q_n)^t = (\ldots, x_k, y_k, z_k, \alpha_k, \beta_k, \gamma_k, \ldots)^t$ and n denotes the total degrees of freedom of all movable objects (six times the number of movable objects).

Considering the Taylor expansion of the equation, we obtain

$$\mathbf{f(q + \Delta q)} \cong \mathbf{f(q)} + (\frac{\partial \mathbf{f}}{\partial \mathbf{q}})\Delta \mathbf{q}, \tag{10}$$

where $(\frac{\partial \mathbf{f}}{\partial \mathbf{q}})$ is a n by m Jacobian matrix. At $\mathbf{q} + \Delta \mathbf{q}$, the face contact constraint should be satisfied:

$$\mathbf{f(q + \Delta q)} = \mathbf{0}. \tag{11}$$

Then the correction value $\Delta \mathbf{q}$ is obtained by solving the linear equation

$$(\frac{\partial \mathbf{f}}{\partial \mathbf{q}})\Delta \mathbf{q} = -\mathbf{e} \tag{12}$$

Since $\Delta\mathbf{q}$ has redundant degrees of freedom, the rank of the matrix $\mathbf{q}$ is equal or less than n. We can obtain $\Delta\mathbf{q}$ by using the singular value decomposition method which we will discuss in the next section.

Once $\Delta\mathbf{q}$ is obtained by using the singular value decomposition method, we can correct $\mathbf{q}$ as

$$\mathbf{q}_{new} = \mathbf{q}_{old} + \Delta\mathbf{q}, \tag{13}$$

$\mathbf{q}_{initial}$ is given by observation. We will repeat this iterative process until $\mathbf{q}$ converges.

2.4 Redundant degrees of freedom

The face contact equations have redundant degrees of freedom. For the parameters corresponding to the redundant degrees of freedom, we will use the values given by observation. The other parameters will be updated to satisfy the face contact constraints.

Simultaneous linear equations with redundant degrees of freedom can be solved using the singular value decomposition method.

In the case of $m \geq n$[2] ,singular value decomposition of m by n matrix, $\frac{\partial \mathbf{f}}{\partial \mathbf{q}}$ is

$$\frac{\partial \mathbf{f}}{\partial \mathbf{q}} = UWV^t, \tag{14}$$

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

$$W = diag(w_j), \tag{15}$$

where $w_1 \geq w_2 \geq \ldots \geq w_l = w_{l+1} = \ldots = w_n = 0$.

Let's consider the physical meaning of this W.

Eq. 12 is written using the decomposition terms:

$$UWV^t\Delta\mathbf{q} = -\mathbf{e} \tag{16}$$

[2]In the case of $m < n$, we can satisfy $m \geq n$ by adding rows which contain only zero elements .

$$WV^t\Delta\mathbf{q} = -U^t\mathbf{e}, \tag{17}$$

$$W\mathbf{v} = \mathbf{u}, \tag{18}$$

where

$$\mathbf{v} = V^t\Delta\mathbf{q} \tag{19}$$

and

$$\mathbf{u} = -U^t\mathbf{e}. \tag{20}$$

This equation shows that even if the lower $n-l$ elements of $\mathbf{v}$, which correspond to zero singular values, change their values, they do not affect the error vector $\mathbf{u}$. In other words, these values cannot be determined by the given face contact constraints. These lower elements correspond to the redundant degrees of freedom.

For these redundant degrees of freedom, we will use the initial values given from observation; we will not correct configurations of the objects with respect to the redundant degrees of freedom. We will update parameters corresponding to non-zero singular values. Thus, we will update the configurations of objects using

$$\mathbf{v} = W^*\mathbf{u}, \tag{21}$$

where

$$W^* = diag(w_1^{-1}, w_2^{-1}, \ldots, w_l^{-1}, 0, \ldots, 0). \tag{22}$$

Finally, we obtain the correction term[3] as

$$\Delta\mathbf{q} = -VW^*U^t\mathbf{e}. \tag{23}$$

[3]This correction term gives the least square solution of Eq. 12. Thus, the converged value given by the Newton-Raphson method is the minimum of square of f (q) in the neighborhood.

3 Adjustment of Motion Parameters

This section will discuss the method for adjusting motion parameters by using the technique for solving simultaneous face contact equations which was developed in the previous section. First, we will explain the method for adjusting the current configurations of objects so that all the face contact equations have zero residues. Then, we will explain the difference between adjusting the current configurations and adjusting the motion parameters. Finally, we will connect these two adjustments and complete the method which adjusts the motion parameters.

3.1 Adjustment of configurations

We will introduce a face contact relation graph to express face contact relations among objects. Each arrow in the graph represents a face contact relation recovered from observation. The root arrowed node is the *manipulated* object, which is moved by the assembly operation to generate the face contact relation, while the end arrowed node is the *environment* object, which is stationary during the assembly operation [7]. The goal configuration of each assembly operation is attached to the node corresponding to the manipulated object of the operation.

Let us consider the example in Figure 3. Work Table is fixed in the world coordinate system [4]. Castle 1 is put on Work Table by the assembly operation, T_1, and achieves a face contact relation, R_1. Since Cube 1 is the manipulated object and Work Table is the environment object in this operation, the arc points to Work Table from Castle1. Castle2 is then put on Work Table, T_2, by achieving R_2 face contact relation. Finally, Stick is put on both castles by T_3, by achieving R_3 and R_4 relations.

For each arc in the graph, we have set up face contact equations. In each equation, $\mathbf{n}$ is given by a face of the manipulated object. For example, Castle 1 in Figure 3 with respect to the assembly operation which generates the contact relation R_1 and $\mathbf{v}$ is given by a vertex of the environment object (for example Work Table in Figure 3).

By examining the residues of the all face contact equations, we can identify configurations needing to be adjusted. Then, we solve face contact equations with variables of such configurations using the method described in the previous

[4] T_0 is the configuration of Work Table. You may consider that Work Table is put in the world by an imaginary assembly operation whose goal configuration is T_0.

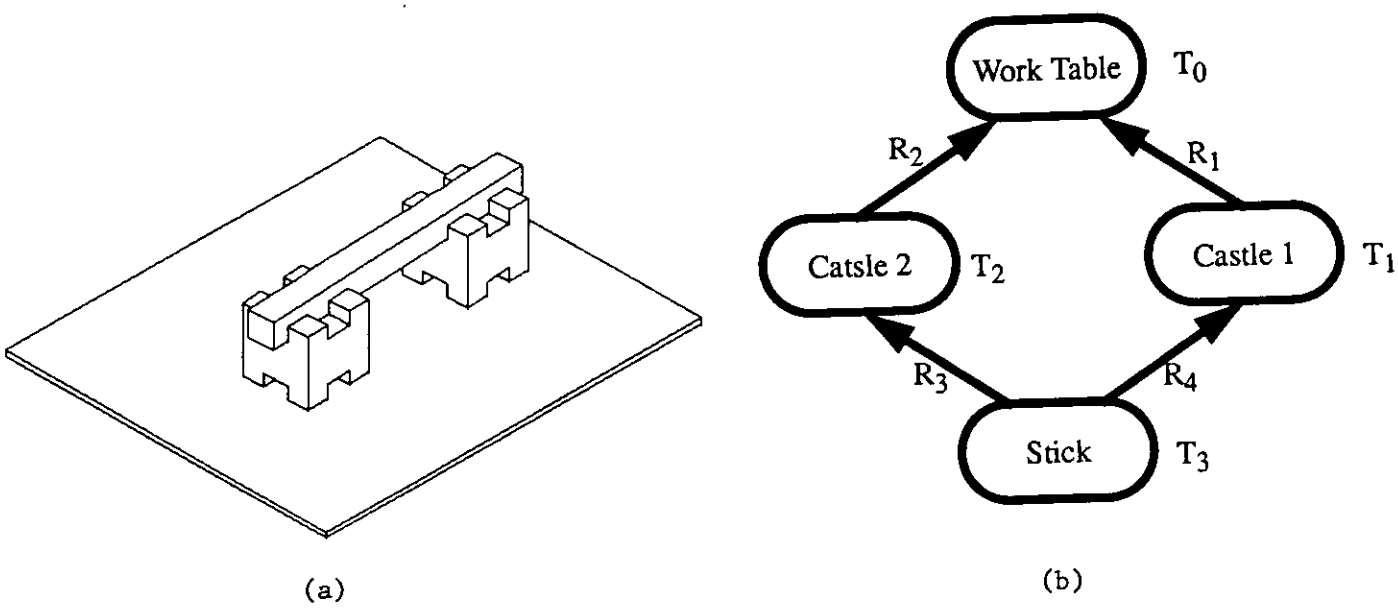


Figure 3: Stick between Two Castles; (a) Internal geometric representation, (b) relation graph.

section. The solutions provide the adjusted object configurations.

More precisely, we can set up an adjustment procedure for object configurations using face contact relations as follows:

- Identifying variable matrices

    1. Remove nodes of stable objects from the relation graph. (We cannot modify the configuration of the stable object.)

       -> Work Table is a stable object; Work Table node is removed from the relation graph. Note that $R_{castle1-table}$ and $R_{castle2-table}$ arcs are not removed.

    2. Select the arcs corresponding to the face contact relation which should be adjusted by examining residues of each face contact equation. Any face contact relations having residues larger than a certain threshold are selected.

       -> Let us suppose that face contact equations between stick and two castles have large residues; error terms, $e_{stick-castle1}$ and $e_{stick-castle2}$ are larger than a threshold value. Thus, Arc $R_{stick-castle1}$ and $R_{stick-castle2}$ are selected.

3. Choose objects which are connected in the relation graph containing the selected arcs.
   -> Castle 1, Castle 2 and Stick are chosen because these three objects are related directly in the relation graph.
4. Assign variable matrices to the configuration of the chosen objects.
   -> $T_{castle1}$, $T_{castle2}$ and T_{stick} are the homogeneous coordinate matrices of Castle 1, Castle2, and Stick, respectively. The variable matrices, $\Delta T_{castle1}$, $\Delta T_{castle2}$ and ΔT_{stick} are assigned.

- Solving simultaneous face contact equations.

1. Set up simultaneous face contact equations, $f(\mathbf{q}) = \mathbf{e}$ for all related contact equations in the modified relation graph.
   ->

$$
\begin{aligned}
f_{castle1-table} : e_{castle1-table} &= \mathbf{n}_{castle1}\Delta T_{castle1}^{-1} T_{castle1}^{-1} T_{table}\mathbf{v}_{table} \\
f_{castle2-table} : e_{castle2-table} &= \mathbf{n}_{castle2}\Delta T_{castle2}^{-1} T_{castle2}^{-1} T_{table}\mathbf{v}_{table} \\
f_{stick-castle1} : e_{stick-castle1} &= \mathbf{n}_{stick}\Delta T_{stick}^{-1} T_{stick}^{-1} T_{castle1}\Delta T_{castle1}\mathbf{v}_{castle1} \\
f_{stick-castle2} : e_{stick-castle2} &= \mathbf{n}_{stick}\Delta T_{stick}^{-1} T_{stick}^{-1} T_{castle2}\Delta T_{castle2}\mathbf{v}_{castle2}
\end{aligned}
\tag{24}
$$

   Note that since $R_{castle1-table}$ and $R_{castle2-table}$ are connected from Castle 1 and Castle 2, the corresponding contact equations are also included in the simultaneous equation. However, no variable matrix is assigned to Table.
2. Calculate the Jacobian of $\mathbf{f}$, $\frac{\partial \mathbf{f}}{\partial \mathbf{q}}$.
   -> Calculate the partial derivatives of $f_{castle1-table}$, $f_{castle2-table}$, $f_{stick-castle1}$, $f_{stick-castle2}$ with respect to $(\cdots, x_{castle1}, y_{castle1}, z_{castle1}, \alpha_{castle1}, \beta_{castle1}, \gamma_{castle1}, \cdots)$ using the formula in Appendix.
3. Obtain $\Delta\mathbf{q}$ using the singular value decomposition method as Eq. 23.
   -> Obtain ΔT_{stick}, $\Delta T_{castle1}$, and $\Delta T_{castle2}$.
4. Update $\mathbf{q}$.
   -> Correct T_{stick}, $T_{castle1}$ and $T_{castle2}$ using the correction term, ΔT_{stick}, $\Delta T_{castle1}$, and $\Delta T_{castle2}$.
5. Iterate from 1 to 4 until $\mathbf{q}$ converges.
   -> Iterate from 1 to 4 until T_{stick}, $T_{castle1}$, and $T_{castle2}$ do not move.

3.2 Modification of face contact relation graph

Our goal is to generate an error free assembly plan; to obtain error free motion parameters of assembly operations in the plan. Usually, the correspondence between one motion parameter of one assembly operation and a current object configuration in the relation graph is one to one. Namely, we can use the current configuration of the manipulated object as the motion parameter of the assembly operation. (See Figure 4(a).) In this case, we can simply modify the motion parameters according to the adjustment of the object configuration describes in the previous section. However, when several objects move by one single assembly operation, then the one to one relation between the configurations and motion parameters does not hold. This subsection will consider how to modify the contact relation graph to accommodate such cases.

3.2.1 Operation dependency list

When several sub-assembled objects move, the motion parameter of the operation is represented by the current configuration of the manipulated object. However, the configuration of a sub-assembled object attached to the manipulated object is determined not only by the current motion parameter but also by the previous motion parameters which determine the relationship between the sub-assembled and the manipulated object.

In Figure 4(b), at step 1, the configuration of object A is given by the motion parameter D_m. Then at Step 2, we place object B using the motion parameter D_s. At step 3, we move object A using the motion parameter T. Then due to this operation, the object B also moves.

In order to represent such relations, we will introduce an operation dependency list (ODL), a set of operations which cause an object configuration change. We will attach an ODL to each object node instead of object configurations in a relation graph.

An ODL is a symbolic list of the homogeneous transformations of motion parameters. The product of the homogeneous transformations in a ODL represents the current configuration of the object. Thus, an ODL provides the relation between the current configuration and the motion parameter which locates the object through the previous assembly operations.

At each step, an ODL of an object is updated accordingly. In the case that an operation is applied to a single object, its ODL contains the motion parameter of

the operation. In the case that an operation is applied to sub-assembled objects, the ODL of the manipulated object contains the motion parameter of the operation, as is the case for a single object. On the other hand, an ODL of a remaining object in the sub-assembled object group contains the current motion parameter, the inverse of the previous ODL of the main object, and its previous ODL. Thus, the ODL provides the relationship between the current configuration, which is given as the product of the ODL, and the motion parameters of the previous operations.

For example, a sub-assembled relation is formed between main and sub object in Fig 4(c). At step 1 and 2, object A and B have ODLs D_m and D_s, respectively. They form a sub-assembled object group. At step 3, we apply an operation represented as T to the main object. The main object is the manipulated object of this operation. Then, the ODL of the main object is updated to T. Since the sub-object is sub-assembled to the main object, it also changes its configuration by this operation. Thus, the ODL of the sub object is updated to $TD_m^{-1}D_s$ which corresponds to the list of the current operation, T, the inverse of the previous ODL of the main object, D_m, and the previous ODL of the sub object, D_s. The product of the contents of the ODL provides the current configuration of the sub object.

A face contact equation is described by using a ODL as:

$$e = \mathbf{n}D_a^{-1}D_b\mathbf{v} \tag{25}$$

where D_a and D_b are ODLs of contacting objects.

Using face contact equations given by ODLs instead of those given by transformations of object configurations, we can obtain adjusted motion parameters, embedded in the ODL, in a way similar to the adjustment of configurations described in section 3.1.

3.2.2 Disappeared relation

Another difficult case occurs when the current object configuration is given by another object which has already disappeared from the scene and leaves the second object not contained in the current relation graph. For example, in Figure 5, at Step 1, we place object B using T_1. You may consider object B as a fixture. Then, at Step 2, we align object A using the fixture and parameter T_2. Since we do not need to have the fixture, at Step 3, we remove object B using T_3. The current configuration of object A at Step 3 is given by the previous relationship between object A and B.

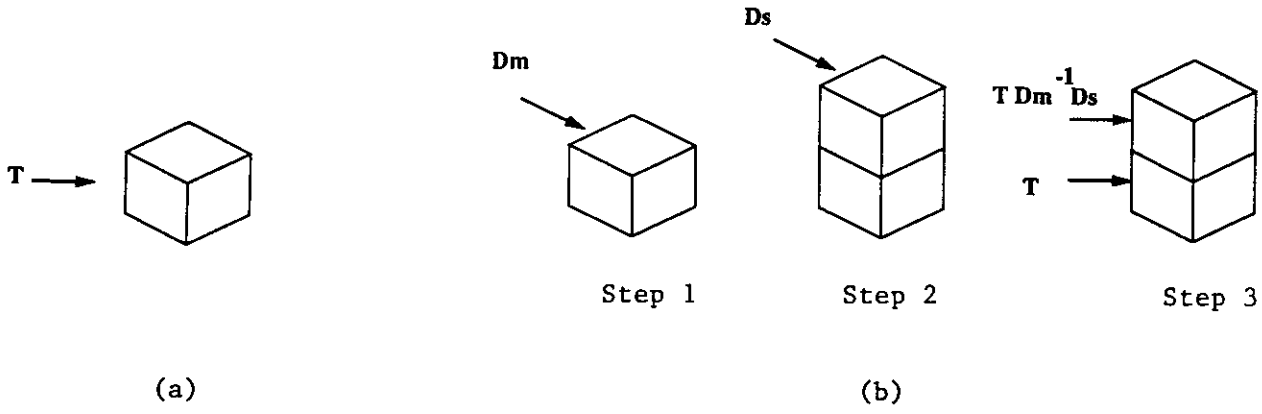


Figure 4: Operation Dependency List; (a) single object, (b) sub-assembled objects, ODLs of the sub-assembled objects.

In order to represent such disappeared relations, we will record all relations which occur during operations so that we can retrieve necessary disappeared relations. Due to this modification of the graph, when several assembly operations have been applied to the same object, the object appears as multiple nodes in the graph.

3.3 Adjustment procedure for motion parameters

By attaching ODLs and maintaining disappeared relations the face contact relation graph maintains the motion parameters of all assembly operations applied previously. Thus, by applying the method described in the previous section to a modified graph, we can determine the adjusted motion parameters.

More precisely, we will examine all residues of face contact equations corresponding to all arcs in the relation graph. If any of the residues are greater than a threshold value, the motion parameter corresponding to the arc should be adjusted. By solving the face contact equations by setting these parameters as variables, we can determine the adjusted motion parameters.

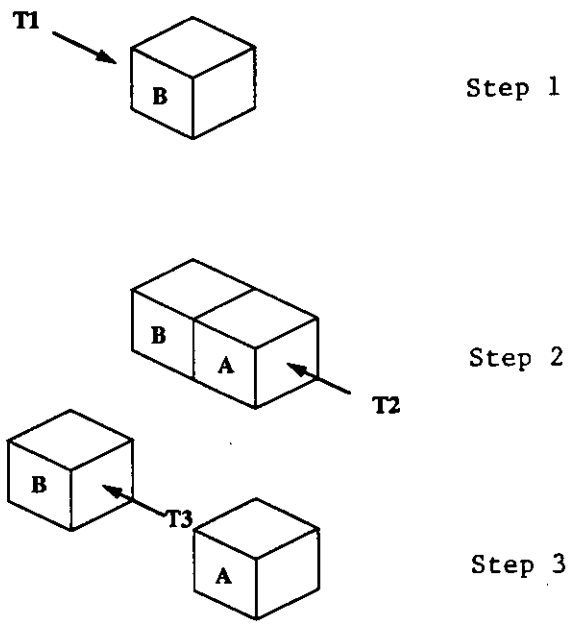


Figure 5: disappeared relation

3.3.1 Example assembly task

Let us consider a sequence of assembly operations shown in Figure 6 for an illustration of motion parameters adjustment. Here, each motion parameter for one assembly operation is represented as homogeneous transformation T_i of the goal configuration of a manipulated object with respect to the world coordinate system.

1. Cube 1 is put on Work Table using T_1 assembly operation.
2. Cube 2 is put on Work Table by T_2.
3. Cube 3 is put on Work Table making a face contact to Cube 1 by T_3.
4. Cube 1 is moved away from Cube 3 by T_4. Cube 1 loses face contacts with Cube 3 and Work Table.
5. Stick is inserted into Cube 2 and Cube 3 by T_5. Stick makes face contacts with Cube 2 and Cube 3. Stick, Cube 2, and Cube 3 form a subassembled object group.
6. The subassembled group, Stick, Cube 2 and Cube 3, is moved simultaneously and put into holes of Work Table together by T_6. Old face contacts of these objects with Work Table disappear and new face contacts with Work Table are made. Stick is the manipulated object and the operation applied to whole subassembled objects is represented as T_6, which expresses the goal configuration of Stick.

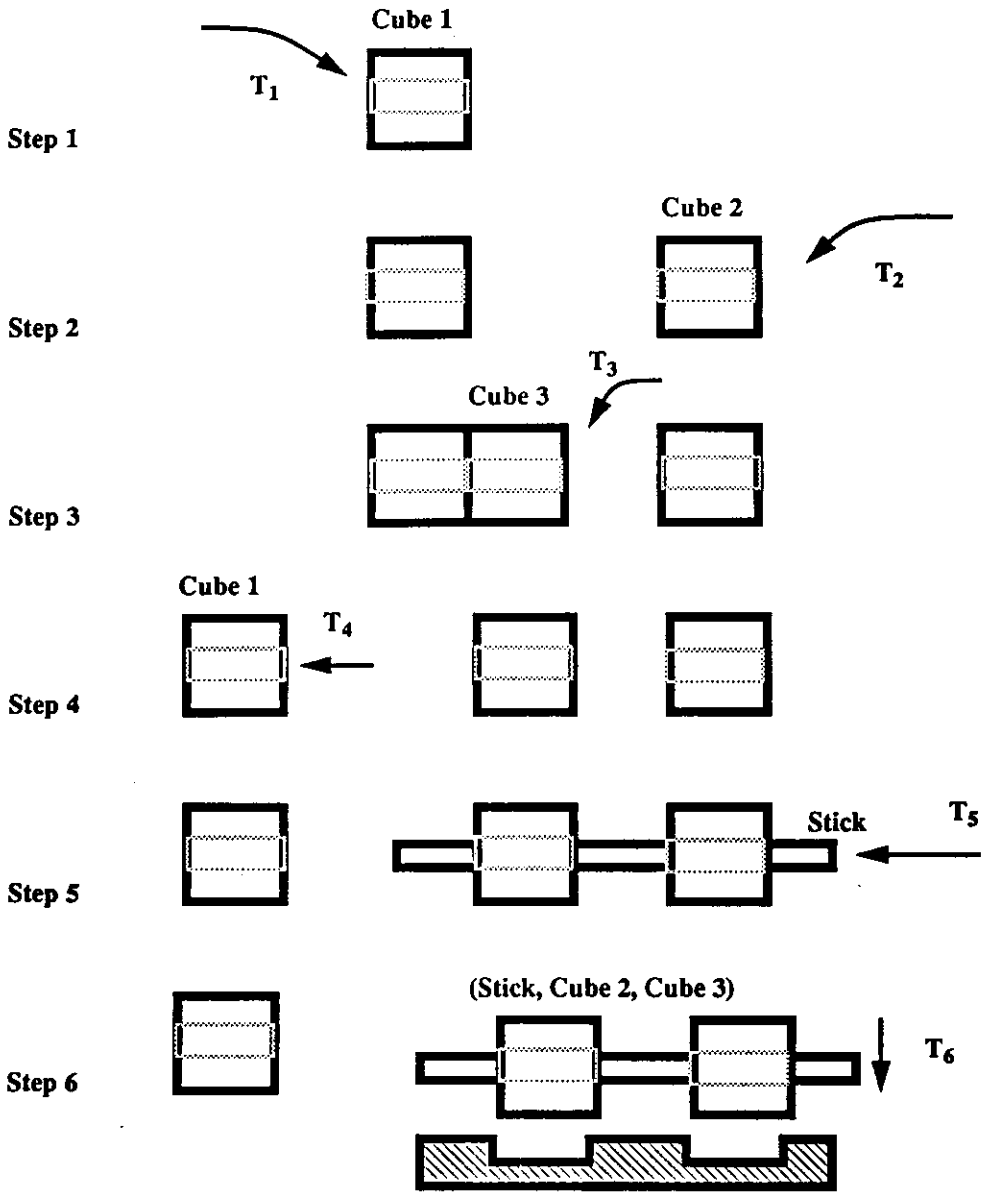


Figure 6: Example Task

At Step 6, in order to put them into the two holes correctly, we have to adjust the configuration relation between the two cubes; this adjustment cannot be achieved by the adjustment of current configurations. We have to adjust motion parameters in Step 2 and Step 3 so that they create the proper relations among subassembled objects. For this purpose, we have introduced ODLs.

Let us consider how to adjust the configurations of two Cubes and Stick at Step 5. For the adjustment of Cube 3, we have to consider the contact of Cube 1 and Cube 3 at Step 2. However, such face contact has already disappeared at Step 5. In order to remedy this situation, we have maintained disappeared relations in the graph.

Relation graphs at each step of the task shown in Fig 6 are shown in Fig 7. At Step 4, Cube 1 is moved away from Cube 3 and Cube 1 loses face contacts with Cube 3 and Work Table. But these face contact relations are maintained in the relation graph, in order to maintain disappeared relations. At Step 6, ODLs of sub-assembled objects Stick, Cube 2 and Cube 3 are updated as described in the previous section.

3.3.2 Adjustment procedure

Let us consider Step 6. In order to put Stick, Cube 2 and Cube 3 into holes of Work Table correctly, the adjustment procedure is as follows:

- Identifying variable matrices for motion parameters.

  1. Remove stable objects from the relation graph.
     -> Work Table is a stable object.
  2. Determine the arc corresponding to the face contact relation which should be adjusted.
     -> Let us suppose that face contacts between two cubes and work table should be adjusted; error terms $e_{Cube2-Worktable}$ and $e_{cube3-Worktable}$ are larger than a threshold value. Arcs R8 and R9 are determined to be adjusted.
  3. Select objects in the relation graph which are connected directly to the arc(s).
     -> Cube2 and Cube3 are connected to arc R8 and R9. Thus, Cube 2 and Cube 3 are selected.

4. Retrieve ODLs of the selected objects and extract motion parameters in the ODL(s).

   -> Cube2 has a ODL containing motion parameters, T_2, $T5$, T_6. Cube3 has a ODL containing T_3, T_5, T_6.

5. Choose motion parameters which are connected in the relation graph to the extracted motion parameters.

   -> Since T_1 is connected from T_3 through R_4, T_1 is also chosen. Thus, T_1, T_2, T_3, T_5 and T_6 are chosen for adjustments. Note that T_4 is not related to those operations; it is not chosen for adjustments. Note that T_1 is retrieved through a disappeared relation R_4.

- Solving simultaneous face contact equations.

1. Set up contact equations for each contact relation generated by chosen motion parameters. Here the contact equation is described by using ODLs as:

$$e = \mathbf{n} D_a^{-1} D_b \mathbf{v} \tag{26}$$

->

$$\begin{aligned}
R1 : e_{cube1-table} &= \mathbf{n}_{cube1} \Delta T_1^{-1} T_1^{-1} T_0 \mathbf{v}_{table} \\
R2 : e_{cube2-table} &= \mathbf{n}_{cube2} \Delta T_2^{-1} T_2^{-1} T_0 \mathbf{v}_{table} \\
R3 : e_{cube3-table} &= \mathbf{n}_{cube3} \Delta T_3^{-1} T_3^{-1} T_0 \mathbf{v}_{table} \\
R4 : e_{cube3-cube1} &= \mathbf{n}_{cube3} \Delta T_3^{-1} T_3^{-1} T_1 \Delta T_1 \mathbf{v}_{cube1} \\
R6 : e_{stick-cube2} &= \mathbf{n}_{stick} \Delta T_5^{-1} T_5^{-1} T_2 \Delta T_2 \mathbf{v}_{cube2} \\
R7 : e_{stick-cube3} &= \mathbf{n}_{stick} \Delta T_5^{-1} T_5^{-1} T_3 \Delta T_3 \mathbf{v}_{cube3} \\
R8 : e_{cube2-table} &= \mathbf{n}_{cube2} ODL_{cube2}^{-1} T_0 \mathbf{v}_{table} \\
e_{cube2-table} &= \mathbf{n}_{cube2} \Delta T_2^{-1} T_2^{-1} T_5 \Delta T_5 \Delta T_6^{-1} T_6 T_0 \mathbf{v}_{table} \\
R9 : e_{cube3-table} &= \mathbf{n}_{cube3} ODL_{cube3}^{-1} T_0 \mathbf{v}_{table} \\
e_{cube3-table} &= \mathbf{n}_{cube3} \Delta T_3^{-1} T_3^{-1} T_5 \Delta T_5 \Delta T_6^{-1} T_6 T_0 \mathbf{v}_{table}
\end{aligned}$$

2. Solve these equations, iteratively in the same way as Step 2 to Step 5 in the previous adjustment procedure.

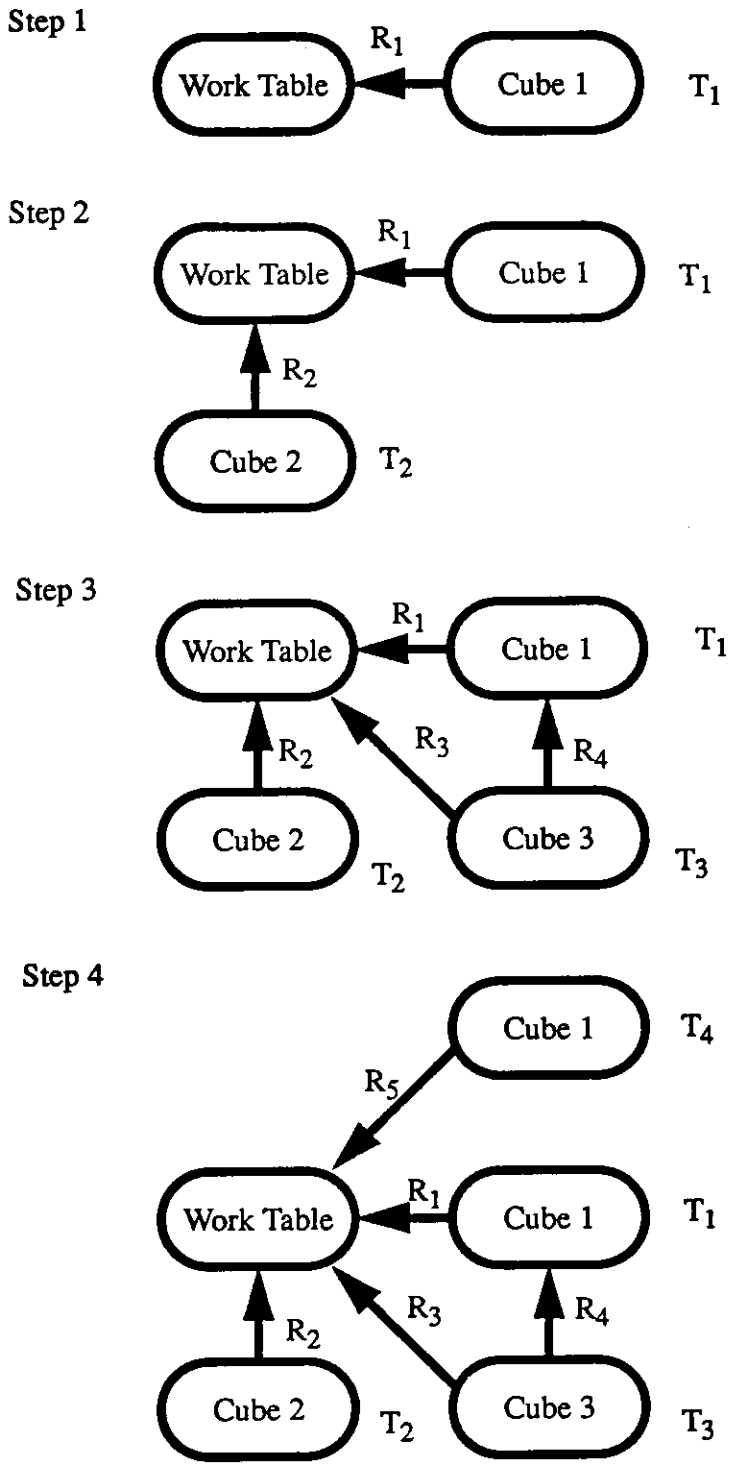
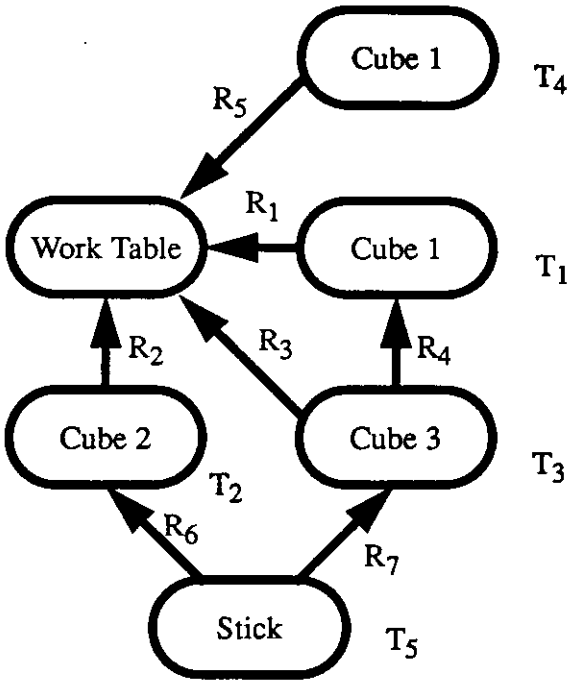
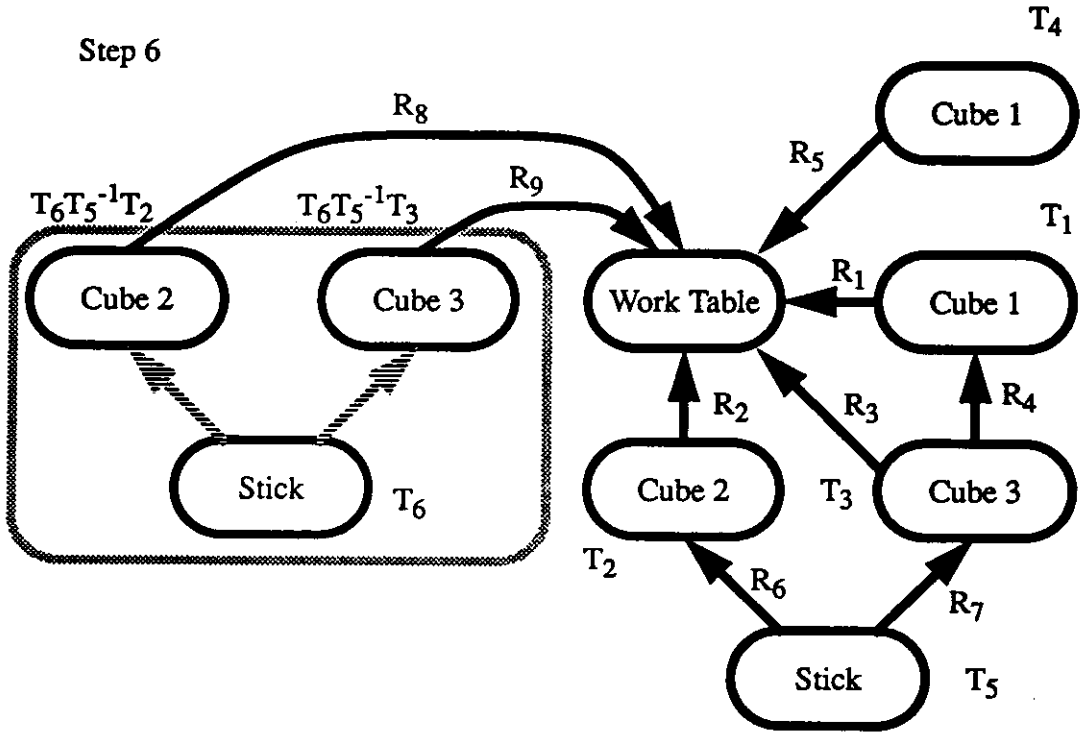


Figure 7: Relation Graph

Step 5



Step 6



Relation Graph (cont)

4 Implementation in Assembly Plan from Observation

4.1 System overview

The human operator performs an assembly operation in front of a TV camera and a range finder. Here, the TV camera continuously observes the scene. When a human operator performs an operation, the brightness values in the images vary. From the difference in brightness values, the system detects when a change occurs in the scene.

After a certain period from the detection in brightness values, the APO system invokes a range finder to generate more reliable depth information of the scene. Comparing the current depth map with the previous depth map obtained at the previous assembly operation, it extracts the difference in range data. The difference in the two range maps corresponds to a manipulated object. By analyzing such difference, the APO system recognizes the manipulated object and determines its configuration.

The APO system generates a geometric representation of the manipulated as well as environmental objects using the Vantage geometric modeler.

A robot assembly operation consists of two parts: a motion macro such as move-to-contact or insert-into, and motion parameters such as from $(0,0,0,0,0,0)$ to $(1,0,0,0,0,0)$. By examining geometric representations of the manipulated and environmental objects, the APO system determines face contact relations between those objects. Consulting the procedure tree, which relates face contact relations with necessary assembly motion macros, the APO system determines the motion macro which causes the current face contact relation. Examining object configurations in geometric representation, the APO system determines motion parameters to complete the assembly operation.

Finally, the APO system commands the assembly operation to the robot to do the same operation.

4.2 Iterative error correction

The current APO system is implemented for a robot to follow the human operation each time. Thus, the error correction procedure runs and corrects object configurations after each assembly observation (range finder observation). Under this

tracking mode of the APO system, we have implemented the following iterative correcting method so that a robot needs fewer error recovering operations. [5]

Let us suppose that the human performs the nth operation and the APO system finding the residue **e** of the face contact equation which has been achieved by the current human operation is greater than a certain threshold value; it finds some inconsistency in configuration parameters. It is possible to eliminate this residue by adjusting all the motion parameters as described in the previous section.

However, the current implementation of the APO system tries to find the k most recent ODLs necessary to correct the residue, where k is between 1 and n. At the first trial, by using only the nth ODL, the APO system executes the adjustment procedure. If the procedure finds $D_{n'}$, the nth adjusted ODL to eliminate **e**, the procedure succeeds, and the APO system performs only the nth assembly operation using the corrected $D_{n'}$. If the correction of the nth operation fails, the APO system tries to correct the nth and the $n-1$th ODLs simultaneously. If it fails, the APO system examines the earlier ODLs until it finds a set of the ODLs $\{D_n, \ldots, D_i\}$ to eliminate **e**. Since the human operator succeeds in achieving the same assembly relations, it is guaranteed that if the APO system uses all the ODLs, it can eliminate the residue. Thus, there should be a set of ODLs among $\{D_n\}$ through $\{D_n, \ldots, D_1\}$ to eliminate **e**.

Let us suppose the system finds a set of the operations $\{D_n, \ldots, D_i\}$ that should be corrected. Our current system does not add any extra assembly operations for error correction besides a reverse order of disassemble operations. The APO system performs the disassemble operations from the $n-1$th operation through the ith operation. Then, it repeats the ith through nth operation using the newly corrected motion parameters.

[5] Depending on the implementation, use either this track mode or all-at-once mode. If all-at-once mode is chosen, the system simply observes the entire sequence of human assembly operations, and builds up the whole face contact relation graph. Then, it solves the simultaneous equations all at once using the method described in the previous section. Thus, it is not necessary to use the iterative error correction method described in this section.

5 Examples

5.1 Simple case

Fig. 8 shows the three-step human operations:

- *Step 1* – put a castle (castle1) on a table,
- *Step 2* – put another castle (castle2) on the castle,
- *Step 3* – insert a stick between the two castles.

At each step, the APO system observes human operations and tries to repeat the same operations.

In this example, we will focus on the 3rd step. This step aims to insert a stick into the hole between two castles. In this example, due to careless human operations at step 2, the hole between the two castles does not have enough clearance to insert the stick. At step 3, a human adjusts this hole and inserts the stick with a simultaneous operation using both hands.

On the other hand, the system only detects an appeared stick and identifies the configuration of the stick; the movement of Castle 2 is too small to be detected. So the internal geometric representation has an illegal face contact relation between the stick and the hole as shown in Figure 9(a). This example aims to correct the motion parameters so that it generates the relation as shown in Fig 9(b). Without this adjustment, the APO system fails to perform the operation as shown in Fig. 10.

At the 3rd step, the APO system generates the relation graph as shown in Step 3 of Fig. 8. From the amount of residues in R_3 and R_4 face contact equations, the system detects inconsistency in the graph. By using the procedure to identify variable matrices in page 19 with R_3 and R_4, the APO system identifies T_3,T_2,T_1 as the ODLs necessary to adjust.

Since the system is in iterative correction mode instead of all-at-once mode, the APO system tries to correct this inconsistency by only adjusting T_3. By assigning a variable to T_3, the system sets up face contact equations. By iteratively solving this equation using the SVD method described in page 20, the system obtains a converged T_3'. Then, the system recalculates the residue, e of the face contact equations using the converged T_3' and examines whether the newly obtained e is less than the threshold value or not. In this example, the residue is still larger than the threshold value, and the trial fails.

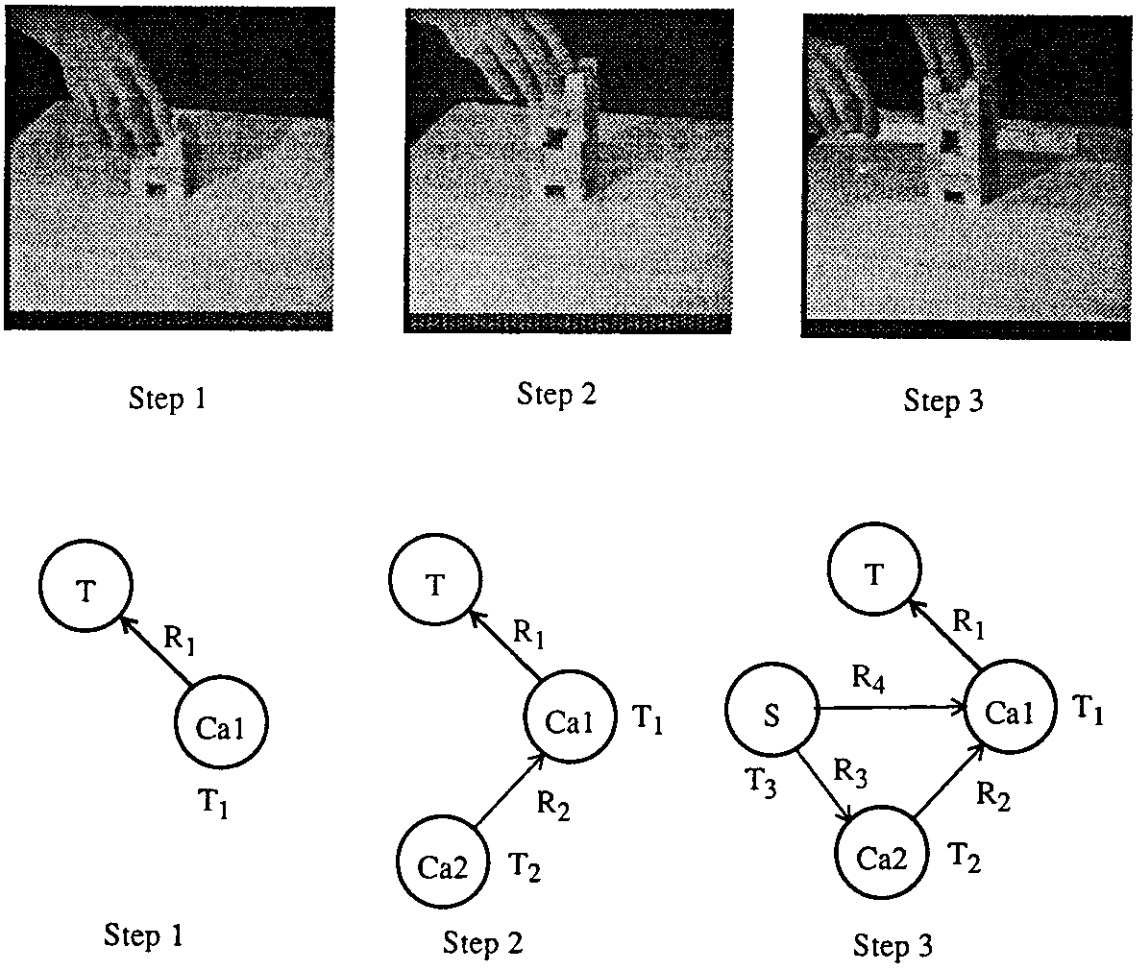


Figure 8: Human operation and face contact relation graph

Then, the APO system tries to repeat the same procedure using T_3 and T_2 as variable matrices. In this case, it succeeds in finding T_2' and T_3' that satisfy the face contact equations (with residues smaller than the threshold value).

The system disassembles Castle 2 using T_2 (Fig. 11(a)) and place Castle 2 using the new T_2' (Fig. 11(b)). Then, the system inserts Stick between Castle1 and Castle 2 using the new T_3' (Fig. 11(c)).

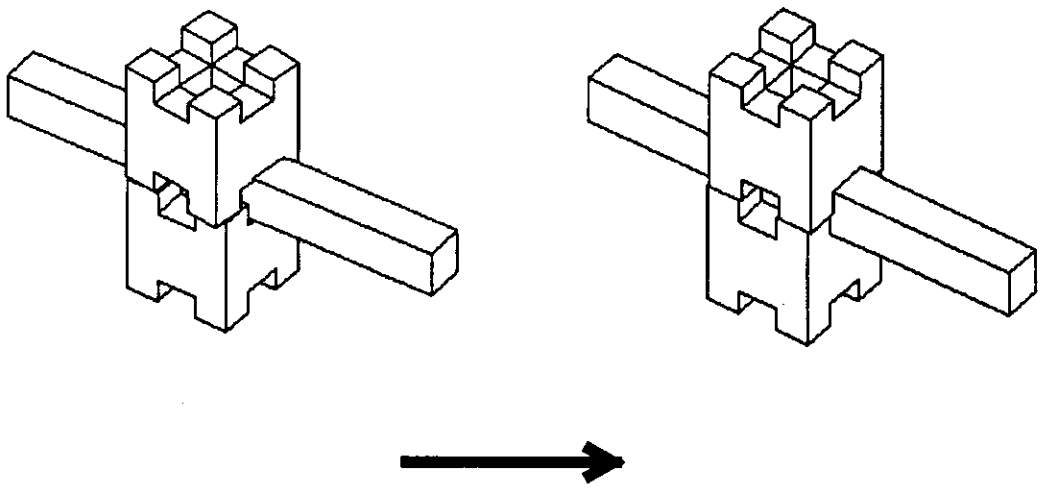


Figure 9: Stick between Castle (drawings)

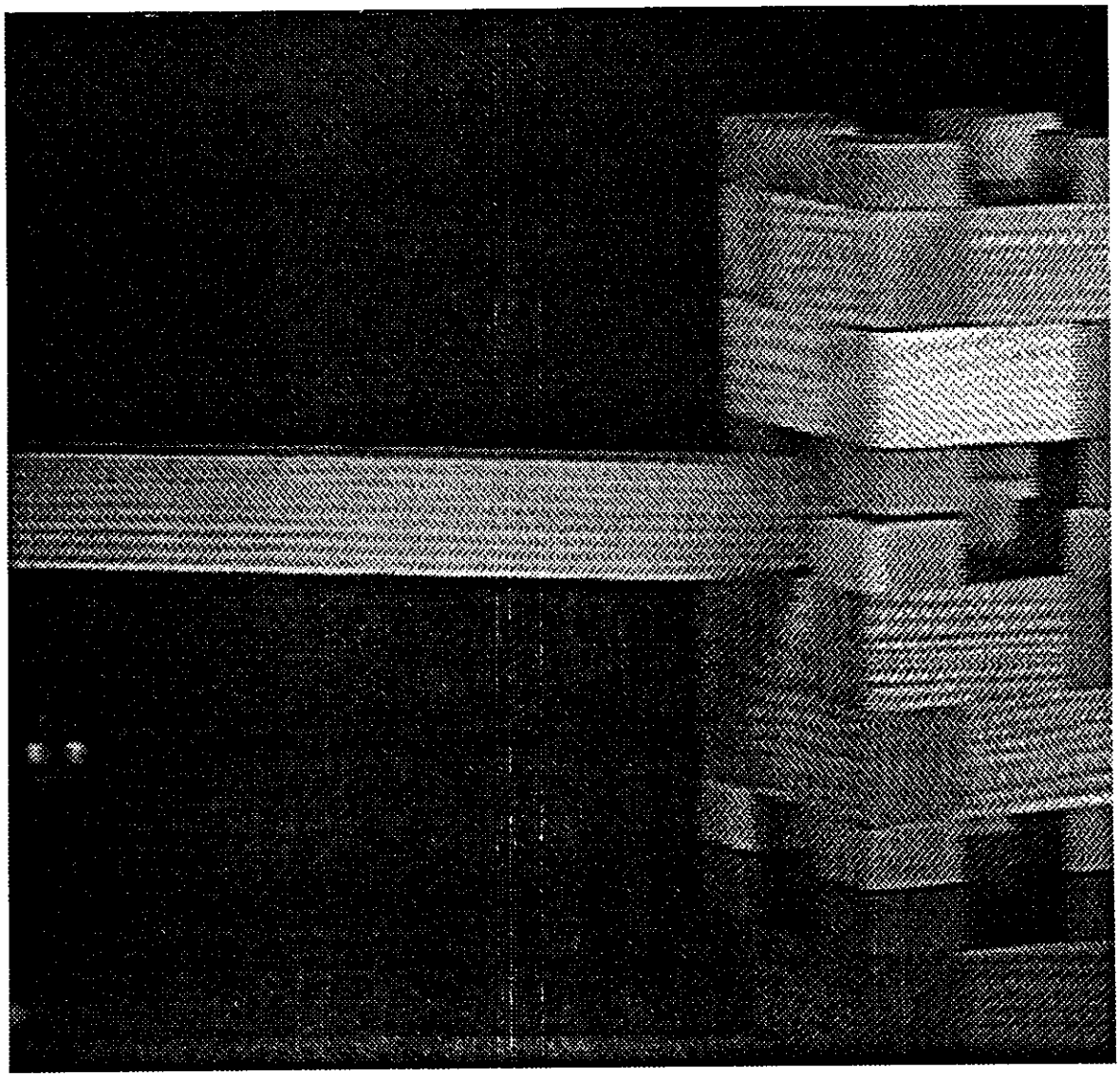
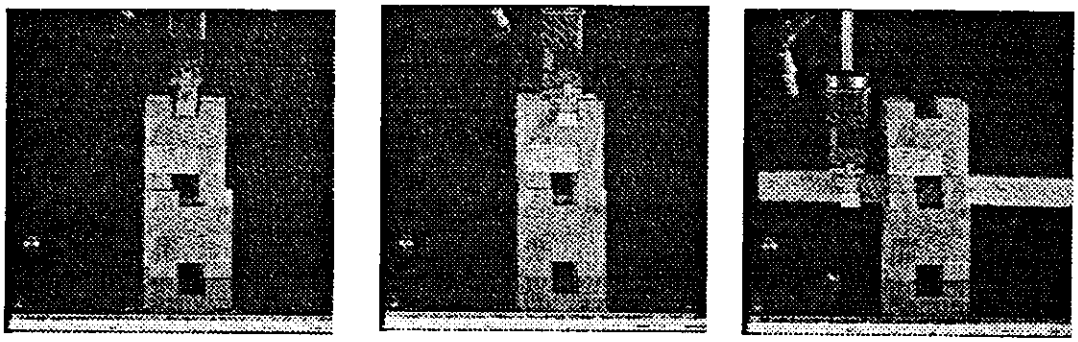


Figure 10: Failure of Example 1



(a) (b) (c)

Figure 11: Robot operation

5.2 Operations containing sub-assembled objects

A human performs the following operations:

- *Step 1* – put Castle 1 on Work table,
- *Step 2* – put Castle 2 on Work table,
- *Step 3* – put Cube on Castle 1,
- *Step 4* – insert Stick into Cube (Stick and Cube are sub-assembled.),
- *Step 5* – put Stick and Cube on Castle1 and Castle2.

These steps and their face contact relation graphs are shown in Fig. 12.

Due to the human operational error at Step 2, the system predicts the failure as shown in Fig. 13.

At Step 5, the face contact equations given by R_5 and R_6 have larger residues. From the face relation graph, Stick, Castle 1 and Castle 2 are related nodes. Stick has the ODL, D_{5s}, containing T_5, T_4, T_3. Castle 1 has the ODL, D_1, containing T_1, and Cast 2 has the ODL,D_2 containing T_2. Thus, the candidate variable matrix set is formed by T_5, T_4, T_3, T_2, T_1. Note that the motion parameters T_3 and T_4 are retrieved through the D_{5s} ODL.

Following the iterative error correcting procedure, the APO examines, $\{D_5m, D_5s\}$, $\{D_5m, D_5s, D_4\}, \ldots,$, iteratively. It finds that the motion parameter set $\{T5, T4, T3, T2\}$ has a solution matrix set that satisfies the face contact equations.

The system disassembles the group of objects following the inverses of Step 4 and Step 3 (Figure 14(a) and (b)). The system modifies the configuration of Castle 2 using D_2' (Figure 14(c) and (d)). It assembles the objects following Step 3, Step 4, and Step 5 using D_3', D_4' and D_{5m}' (Figure 14(e),(f) and (g)).

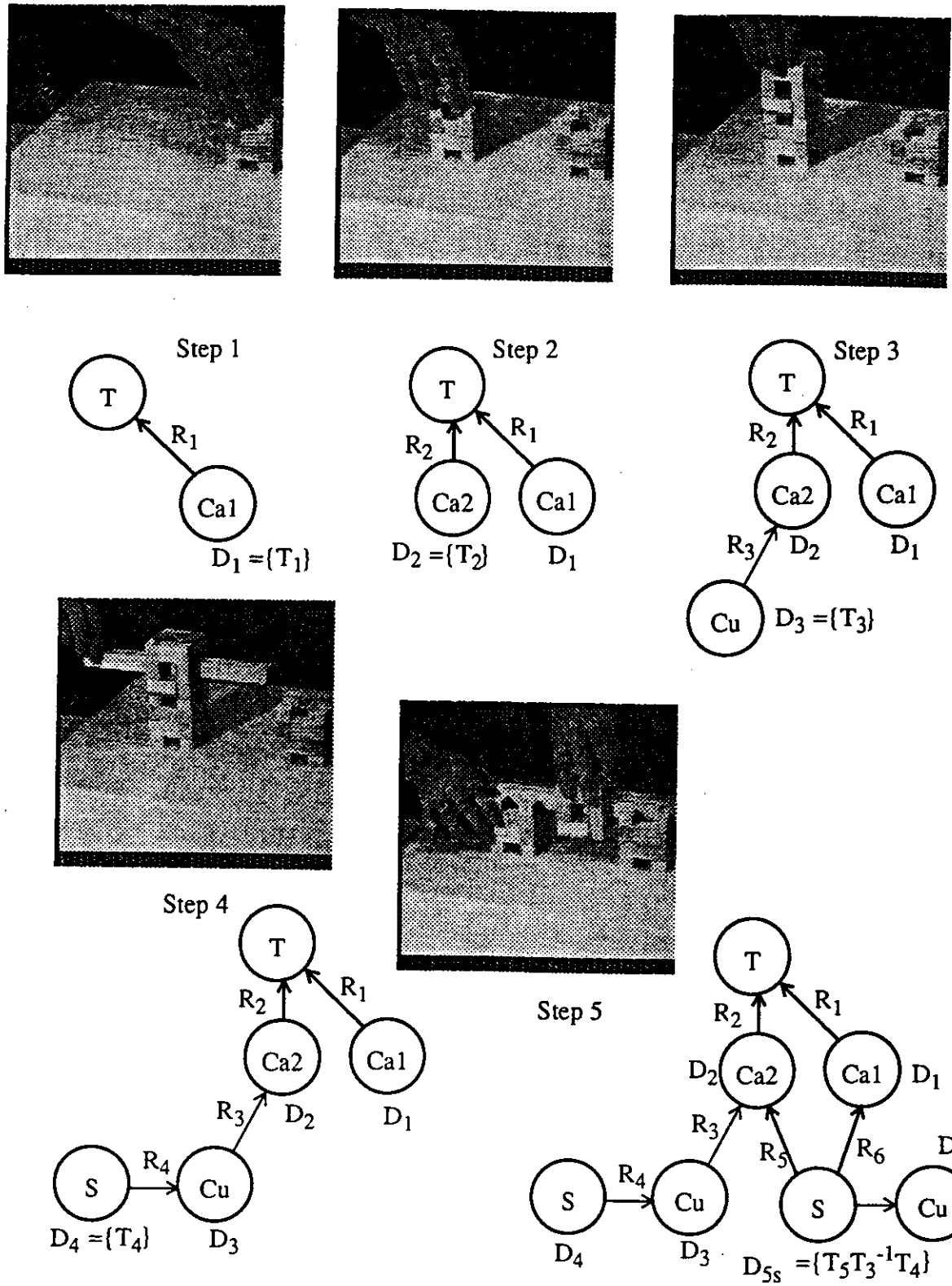


Figure 12: Human Operation 2 (Sub-assembly operations)

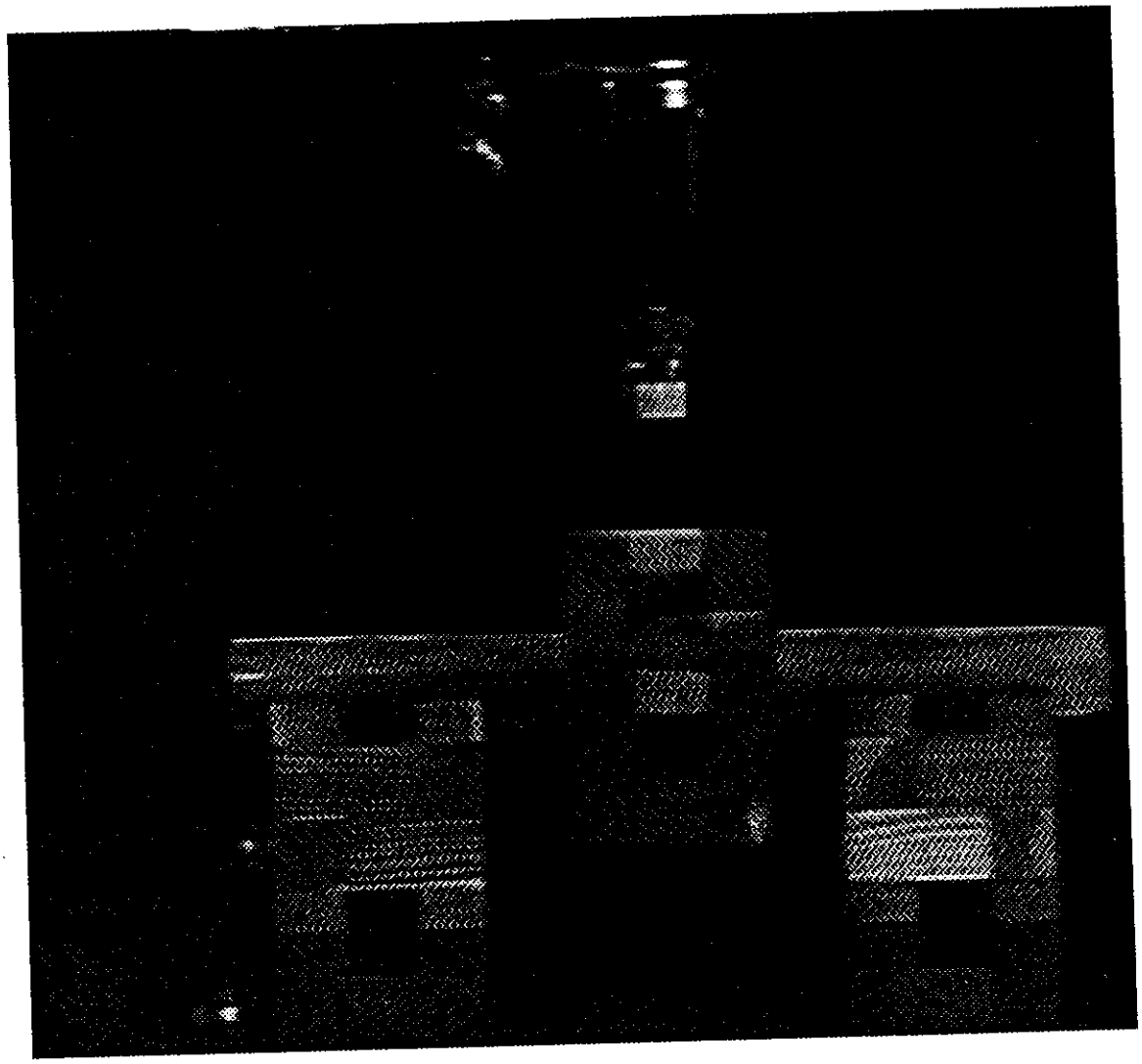


Figure 13: Failure example 2

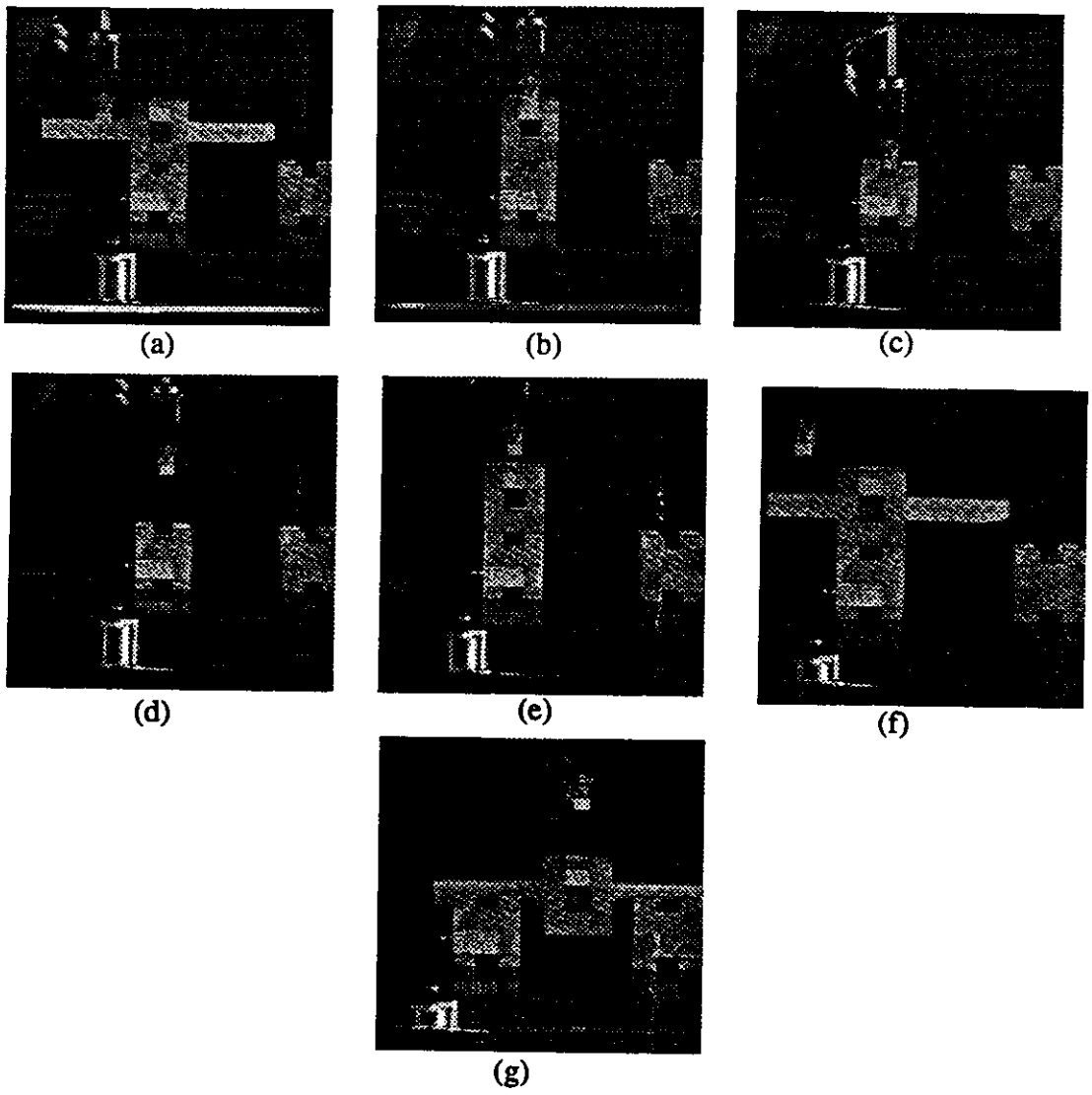


Figure 14: Robot Operation 2

5.3 Operations containing a disappeared relation

The final example contains a disappeared relation which needs to be corrected.

The human operator performs the following operations:

- *Step 1* – put Castle 1 on Work Table,
- *Step 2* – put Cube on Work Table,
- *Step 3* – put Castle2 so that it has face contact with Cube (you may consider the Cube as a fixture to align the configuration of Castle2).
- *Step 4* – removes Cube, (because we have finished alignment of Castle2 with Cube)
- *Step 5* – put Stick on Castle 1 and Castle 2.

See Fig. 15.

At Step 5, the face contact equations given by R_6 and R_7 are not satisfied. The system predicts the situation shown in Fig. 16 if it uses T_5.

Since R_6 and R_7 contain a sub graph formed by Stick, Castle 1, Castle2, and Cube, a motion parameter set, $\{T_5, T_3, T_2, T_1\}$ is formed as a variable matrix set. Note that since the disappeared face contact relation R_4 is maintained as an arc in the graph, the motion parameter T_2 is retrieved though the arc.

The system examines $\{T5\}, \{T5, T3\}, \ldots$, iteratively and obtains T_5', T_3', T_2' to satisfy the face contact equations. The system disassembles using the inverses of Step 4 and Step 3 (Figure 17(a) and (b)), modifies the cube configuration using T_2' (Figure 17(d)), and then assembles using T_3', T_4', T_5' (Figure 17(e),(f) and (g)).

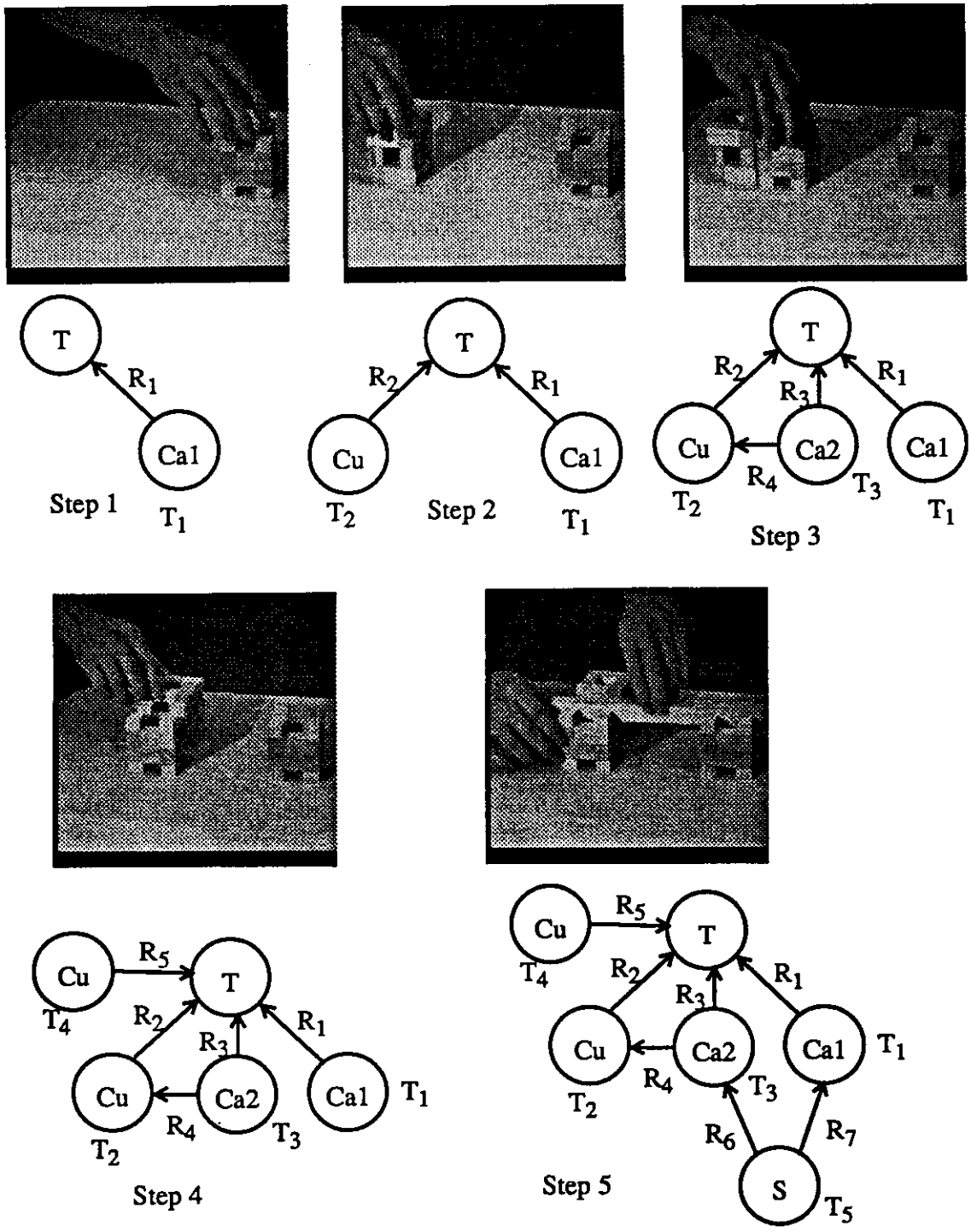


Figure 15: Human operation 3

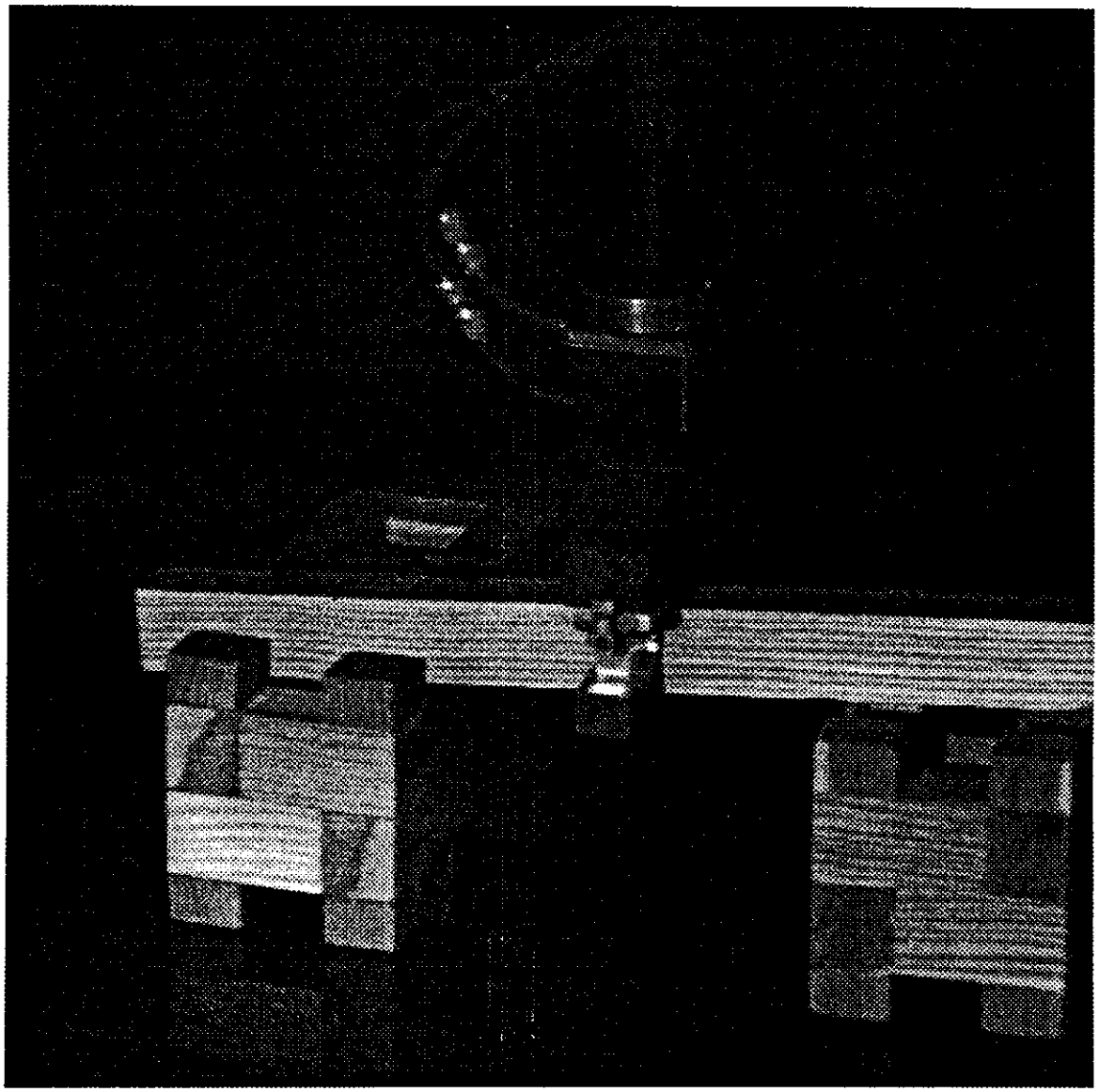


Figure 16: Failure example 3

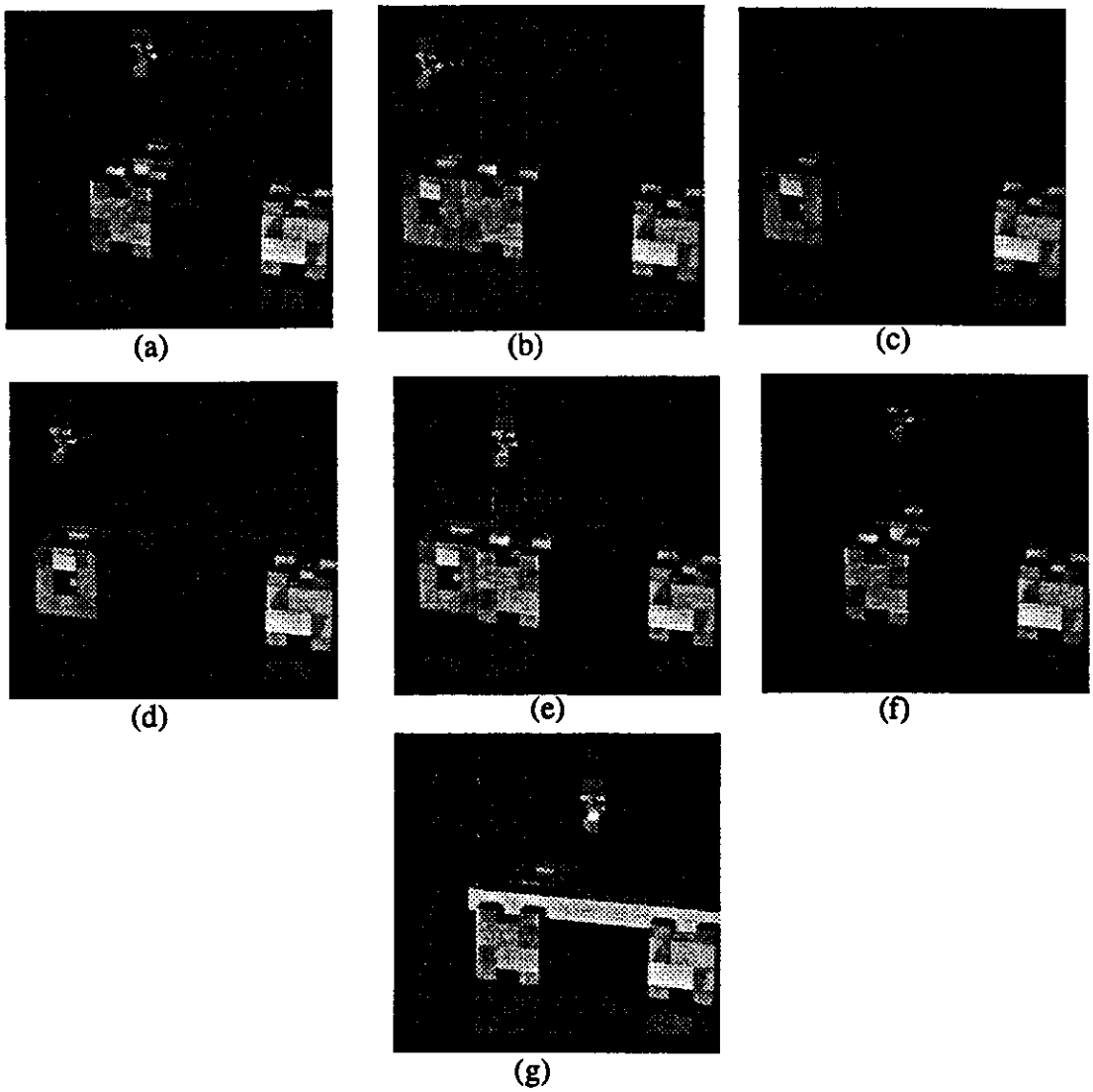


Figure 17: Robot operation 3

6 Conclusion

This paper describes a method to correct motion parameters based on face contact relations. A face contact equation is defined so that a vertex of one face is on the plane including the other face. Motion parameters are corrected by simultaneously solving face contact constraint equations using the singular value decomposition method on the initial values given from observation.

We have represented relations among motion parameters using face contact relation graphs, whose nodes denote objects with motion parameters applied to the objects and whose arcs denote face contact relations. In order to correct motion parameters of subassembled objects, operation dependency lists (ODL)s, symbolic lists of homogeneous transformations to represent motion parameters are introduced.

We have implemented this method in the APO system, applied the method to several assembly examples, and verified the effectiveness of the method.

Future directions include: how to handle curved surfaces, and how to handle objects which have wide clearances.

7 Appendix

Each component of the Jacobian matrix $\frac{\partial f_i}{\partial q_j}$ is calculated by substituting ΔT_k or ΔT_k^{-1} which correspond to q_j in Eq.3 as following values:

$$\frac{\partial \Delta T_k}{\partial x_k} = -\frac{\partial \Delta T_k^{-1}}{\partial x_k} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{27}$$

$$\frac{\partial \Delta T_k}{\partial y_k} = -\frac{\partial \Delta T_k^{-1}}{\partial y_k} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{28}$$

$$\frac{\partial \Delta T_k}{\partial z_k} = -\frac{\partial \Delta T_k^{-1}}{\partial z_k} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{29}$$

$$\frac{\partial \Delta T_k}{\partial \alpha_k} = -\frac{\partial \Delta T_k^{-1}}{\partial \alpha_k} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{30}$$

$$\frac{\partial \Delta T_k}{\partial \beta_k} = -\frac{\partial \Delta T_k^{-1}}{\partial \beta_k} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{31}$$

$$\frac{\partial \Delta T_k}{\partial \gamma_k} = -\frac{\partial \Delta T_k^{-1}}{\partial \gamma_k} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{32}$$

8 Acknowledgement

Raj Reddy and Takeo Kanade provided many useful comments and encouragements. Discussions with Matt Mason and Mike Erdmann are quite helpful. Kathryn Porsche proofread drafts of this paper and provided many useful comments which have improved the readability of this paper. The authors also thank the members of the Task-oriented Vision Laboratory, the Robotics Institute, Carnegie Mellon University, for their valuable comments and suggestions.

References

[1] H.G. Barrow and R.J. Popplestone. *Relational description in picture processing*, pages 377–396. Edinburgh Univ Press, 1970.

[2] H.F. Durrant-Whyte. Consistent integration and propagation of disparate sensor information. *Inten. Journal of Robotics Research*, 6(3):3–24, Fall 1987.

[3] Ayache N. Faverjon B. Faugeras, O.D. and F. Lustman. Building visual maps by combining noisy stereo measurement. In *Proc. of Intern. Conf. on Robotics and Automation*, pages 1433–1438, San Francisco, April 1986. IEEE computer society.

[4] O.D. Faugeras and M. Hebert. The representation, recognition, and locating of 3-d objects. *The International Journal of Robotics Research*, 5(3):27–52, 1986.

[5] R. Finkel, R. Taylor, R. Bolles, R. Paul, and J. Feldman. Al, a programming system for automation. Technical Report AIM-177, Stanford University, Artificial Intelligence Laboratory, Stanford, CA, 1974.

[6] S. Hirai and Sato. T. Motion understanding for world model management of telerobot. In *IEEE Intern. Conf. on Intelligent Robots and Systems*, pages 124–131, 1989.

[7] K. Ikeuchi and T Suehiro. Towards assembly plan from observation: Task recognition with polyhedral objects. Technical Report CMU-CS-91-167, School of Computer Science, Carnegie Mellon University, August 1991.

[8] L.I. Lieberman and M.A. Wesley. Autopass: an automatic programming system for computer controlled mechanical assembly. *IBM Journal of Res. Develop.*, 21(4):321–333, 1977.

[9] Huang T.S. and Faugeras O.D. Liu, Y. Determination of camera location from 2D to 3D line and point correspondences. In *CVPR*, pages 82–88. IEEE Computer Society, June 1988.

[10] D.G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31:355–395, 1987.

[11] T. Lozano-Perez. Automatic planning of manipulator transfer movements. *IEEE Trans. System Man and Cybernetics*, SMC-11(10):681–689, 1981.

[12] T. Lozano-Perez and P.H. Winston. Lama: a language for automatic mechanical assembly. In *Proc. of 5th Intern. Joint Conf. on Artificial Intelligence*, pages 710–716, 1977.

[13] N.O. Sliwa and R.W. Will. A flexible telerobotic system for space operations. In *Proc. Space Telerobotics Workshop*, pages 285–292, Pasadena, 1987.

[14] R.C. Smith and P. Cheeseman. On the representation and estimation of spatial uncertainty. *Intern. Journal of Robotics Research*, 5(4):56–67, 1986.

[15] D.E. Whitney. State space models of remote manipulation tasks. In *Proc. of 1st Intern. Conf. Artificial Intelligence*, pages 495–507, 1969.