Hand Action Perception and Robot Instruction

Yunde Jiar, Mark Wheeler and Katsushi Ikeuchi

March, 1996
CMU-CS-96-116

School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

This research was sponsored in part by the Advanced Research Project Agency under the Department of the Army, Army Research Office under grant number DAAH04-94-G-0006, and in part by National Science Foundation under grant CDA-9121797. Views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of NSF, ARPA, US-ARMY, the United States Government.

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. goverment.

Keywords: Hand perception, Assembly task, Model-based tracking, Depth image sequence, Human action interpretation, Robot instruction

Abstract

It is extremely popular and much more efficient way for teachers or instructors at school, work, or play, to teach their students a task just by demonstrating the task clearly in front of the students. Then the students replicate the demonstrated task to verify if they are able to do it successfully.

Based on this observation, we presents a general and robust approach to hand action perception for automatic robot instruction using depth image sequences. We aim to build such a robot system which should learn not only the initial poses and final destinations of the objects and the orders of assembly, but also should perceive how objects move and how they are manipulated.

In our system the human instructor must simply demonstrate an assembly task in front of a vision system in the human world, no dataglove or special markings are necessary. The recorded image sequences are used to recover a depth image sequence for model-based human hand and object tracking to form the perceptual data stream. The data stream is then segmented and interpreted for generating a task sequence which describes the human hand action and the relationship between the manipulated object and the hand. The task sequence might be composed of a series of subtasks and each subtask involves four phases: approaching, pre-manipulating, manipulating and departing.

In this paper, we also discuss a robot system that replicates the observed task and automatically validates the replication results in the robot world.

Contents

1 Introduction 1

2 Related works 3

3 System overview 4

4 The acquisition of perceptual data stream 7

4.1 Perceptual data stream for interpretation 8
4.2 Model-based localization 9
4.3 Multiple rigid object tracking 11
4.4 Hand tracking 13

5 Spatial relationships in workspace 18

5.1 Relationships between hand and all other objects 19
5.2 Occlusion relations 20
5.3 Spatial relationships among objects 21

6 Hand action interpretation 25

6.1 Assembly task description 25
6.2 Task segmentation 26
6.3 Action interpretation 28
6.4 Robot hand action evaluation 32

7 Implementation 34

7.1 Human demonstration 35

7.1 Robot replication 47

8 Conclusion 51

9 Acknowledgments 51

Reference

1 Introduction

Robots used for automation tasks must be designed for flexibility and fast reprogrammability to meet the possible demands of rapidly changing consumer needs. Unfortunately, programming a robotic system is difficult both for novices and experts. There are many programming methods which have been discussed in order to make robot programming easier, such as teaching by guiding [1], textual programming [5], and task-level languages [8][19]. These methods suffer from problems concerning flexibility and efficiency. There is much work trying to mitigate such problems by using different approaches to robot programming. The typical approach is robot programming by demonstration [10][12][13], or teaching by showing [18][17]. The basic idea behind this method is to instruct a robot by using a user-friendly interfaces and sensors to perceive and interpret the tasks either in the real world or in the virtual environment for generating the control command sequences. The existing work based on this idea still has many substantial problems, such as requiring invasive sensing, or complicated user interfaces, or making assumptions which limit applicability.

As the power of computer is growing more and more and the efficient and practical algorithm of computer vision is available, it is possible to realize the instruction to a robot simply by showing various tasks in front of the robot vision system. This is just based on the observation from the human world. In the human world, it is extremely popular and much more efficient way for teachers or instructors at school, work, or play, to teach their students a task just by demonstrating the task clearly in front of the students. Then the students replicate the demonstrated task to verify if they are able to do it successfully. It is also natural and understandable to expect future robots be novices with fundamental knowledge like human novices, who can learn and replicate the same tasks as the instructor demonstrated. In this view, we can learn a lots from the human activities, and map as much as human world into robot world.

In the human case, students would always pay much attention to not only the motions and states of objects but also the hand action or body action of the instructor when they are being taught, especially for novices. This is because the hand action or body action could be providing rich information for a task about:

1. How manipulating
    . how approaching an object.
    . how touching or grasping an object.
    . how or in which way moving an object.

2. What going on
    . what kind of motion an object is doing

. motion without distinct feature, for example, turing a bolt with a cylinder head.
. occluded object motion.

3. How acting
. wiping, polishing or deburring object.
. pressing object.

Not all of the indications mentioned above are very useful for students, it is absolutely extreme useful for a robot who has not as intelligent as human does. For instance, a child can easily select an approaching path to reach an object without being taught much, but a robot has to do a lot amount of computation for planning approaching path. It is also much difficult for a robot to plan a grasping action while a child easily does. If we simply show a robot how to do these and let him following the same path, the burden of computation and system reliability can be effectively solved. Beyond learning task from the instructor, the robot could be able to perceive itself for reporting and evaluating its action based on the instructor's action, and providing feedback to correct robot action.

In this paper, we describe a more general framework of hand action perception based on depth image sequences for robot instruction. We aim to build a robot system which should learn not only the initial poses and final destinations of the objects and the order of assembly, but also should perceive how objects move and how they are manipulated. Thus. tracking and understanding both object motion and human action will be crucial for a robot to learn a general assembly task. Our system has two multibaseline stereo vision systems with the same configuration located respectively in the human world and the robot world. The human instructor must simply demonstrate the task in front of the vision system in the human world, no dataglove or special markings are necessary. The recorded image sequences are used to recover a depth image sequence for model-based human hand and object tracking to create the perceptual data stream. The data stream is segmented and precisely interpreted to form a task sequence of the description of the human hand action and object motion for generating the robot control sequence or reporting what is going on in the workspace. This work follows the assembly plan from observation (APO) paradigm proposed by Ikeuchi and Suehiro [10].

We begin the report with a brief review of related work in robot programming (Section 2), followed by an overview of system in Section 3. In Section 4, we describe model-based tracking multiple objects as well as human hand with a articulated structure model. Then, the relationships of the hand and objects in workspace is discussed in Section 5. Section 6 gives a paradigm to interpret the hand action based on perceptual data streams. In this section, we also describe how to use such a method to evaluate the robot hand action to see whether it succeeds or fails to replicate

the task the instructor performed. Section 7 demonstrates the implementation of our method in our robot system. We close with a discussion of the approach and topics for future research.

2 Related Work

Ogata and Takahashi [22] use a virtual reality environment as a robot teaching interface. The operator's movement in the virtual reality space via a dataglove is interpreted as a series of robot task-level operations using a finite automaton model. Hashimota and Buss [7] concentrate on static grasp analysis of the human hand while interacting with objects in a virtual reality environment. They use a sensor glove to measure finger joint position and provide force feedback. In contrast with virtual reality, Tung and Kak [33] present a method in which a robot can learn new assembly tasks by monitoring the motion of a human hand in real world using a dataglove. All of these works rely on invasive sensing and could not be used easily to get accurate and complete data about assembly tasks.

Kang and Ikeuchi [12][13] present a more accurate method to track human hand motion using a dataglove and stereo vision. They use recorded CyberGlove and Polhemus data to approximate the motion manipulated object, and use stereo vision and the 3D template matching (3DTM) algorithm [35] to refine the object poses for interpreting the human grasp in the task. For interpreting the human grasp, they have developed a method that uses a representation called the contact web in conjunction with a grasp taxonomy to identify a grasp. The grasp is represented by a grasp abstraction hierarchy which comprises low-level (location and joint angles), intermediate-level (e.g., virtual fingers), and high-level (type of grasp) information. But this work still be a invasive sensing and can not get fine motion of the fingers either.

A non-invasive teaching method is vision-based programming. Inaba and Inoue [11] describe vision-based robot programming system via a computer vision interface. Task specification involves initializing object models using a laser spot scanner. Task programming is subsequently accomplished by moving the established object model to the desired position in the vision interface. Similar to this work, Hamada et al. [6] propose a system in which the operator selects an object from a menu displayed on a computer screen and attaches it to the image of a real scene, where it is superimposed onto the screen image using a mouse. During instruction, the operator can simulate the plan using the screen image and let a manipulator carry out the operations with the aid of a computer vision system. Shepherd [26] describes a visual robot programming environment which allows interaction with the constructed scenes using a mouse and actual scenes captured using a camera, by which the programmer demonstrate the actions that the robot must

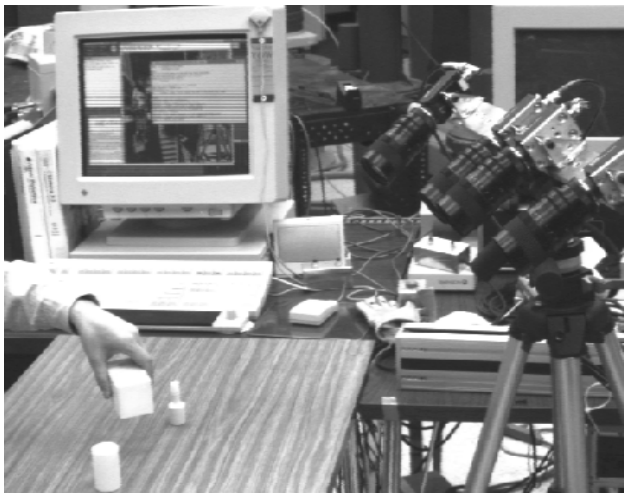
perform. All of these methods still require the special skills of the operators and a great deal of interaction with system.

Kuniyoshi et al. [18] developed a system which can be taught reusable task plans by watching a human performing assembly tasks via a real-time stereo vision system. The human instructor only needs to perform a task in front of the system while a robot extracts task description automatically without disturbing it. But this system is restricted to pick-and-place operations. Although they emphasize that hand tracking is very important in programming by demonstrating, they use simple vision features (mass of center, binary image) to roughly track hand state and then give qualitative descriptions of hand states, such as pick, or place, using heuristic constraints. Such system could not work in the general case, such as inserting, turning, and pushing, neither determine the critical information such as grasping poses, contact points.

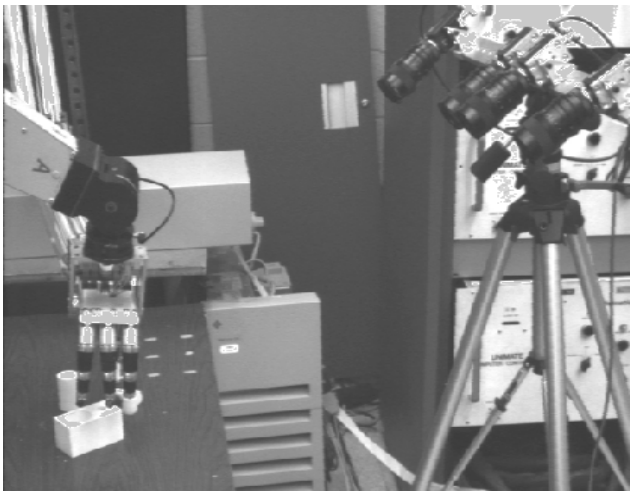
There is some work on the action perception linking with natural language. Thibadea [30] presented a action perception scheme based on logic-linguistic analysis for developing a machine that interprets visually observed events as discrete actions. Siskind [27] presents a method for representing the truth conditions which are expressed as compound expressions in a particular event logic. The events are described by ordinary simple spatial-motion verbs in a way that can be grounded in simulated visual input. But these works only focus on the event representation and inference from the input image stream in the symbolic level. But in the perception, both symbolic and parameters are needed to describe the world model quantitatively and qualitatively.

3 System Overview

We divide our system into two parts called the human world (Figure 1a) and the robot world (Figure 1b). Figure 2 shows the block diagram of the system. In the human world, there are only a four-camera multibaseline stereo vision system [20] [14] and a workstation for computation. The human world might be neighboring to or far away from the robot world for safety or convenience reasons. The vision systems are connected to the iWARP computing system (a high-speed array architecture) via two four-port video interfaces. The iWARP computing system is capable of reading the digitized video signal from the four cameras at video rate, e.g. 30Hz. Currently, we set the sampling speed at 10 frame/second. This makes it possible for the system to completely record the hand and object trajectory during performing tasks. An alternative to the iWARP system is a VCR system containing four video cassette recorders, four time code generators, a time controller and a digitizer [21]. The digitizing image sequences are transmitted to a workstation for the depth image recovery [14], object recognition and action interpretation.



(a) Human world



(b) Robot World

Figure 1 System Overview

In the human world, an instructor is able to demonstrate an assembly task, for instance picking up a block, inserting a cylinder into a hole and/or turning a bolt, in front of the vision system. The instructor does not need to use any other additional devices such as datagloves, complex human-computer interfaces, nor are there any special requirements such as special markings on objects and hands. The recorded image sequences are used to recover a depth image sequence for model-based human hand and object tracking to form the perceptual data stream. The data stream is then segmented and interpreted for generating the task sequence description table (TSDT) which describes the human hand action, the relationship of the hand and the manipulated object, and object motion. The task sequence might be composed of a series of subtasks and each subtask involves four phases, approaching, pre-manipulating, manipulating and departing.

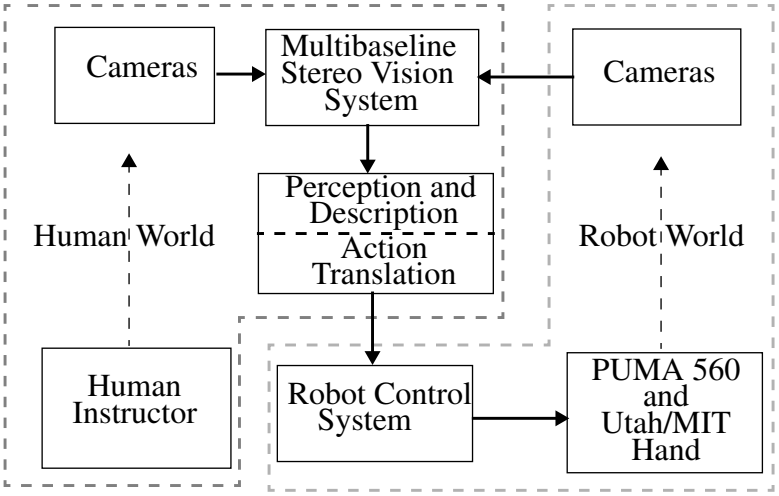


Figure 2 Block diagram of the system

The following is the system working procedure:

1. Calibrate vision System.
2. Model objects.
3. Model hand.

4. Capture image sequences while an instructor or a robot is performing a task.

5. Recover the depth image sequence.

6. Determine initial configuration of work space based on the *zero-th* depth image.

7. Track hand and objects forming the data stream of perception.

8. Segment the data stream.

9. Interpret the human action.

10. Compute the relationships among objects in the workspace.

11. Create the task sequence description table (TSDT).

12. Translate the TSDT from the human world into robot control command sequence.

The first four steps are inputs, 11 and 12 are outputs for the human world. In this paper, we will focus on the 7-11 steps. The interesting readers can refer to the relevant reference for the rest steps, where [14][31] for system calibration, [9] for object modelling, [34] for hand modelling, [14][20] for the acquisition of the image sequence and depth recovery, Kang and Ikeuchi [12] for translating the human action into robot dextrous hand and Tung and Kak [32] into the robot gripper.

In the robot world, the robot system replicates the instructor's action for performing the same task and the vision system supervises robot actions to report what is going on and evaluated the robot action is successful or in failure by applying the same approach of interpretation in the human world. The robot system comprises a PUMA 560 arm and a dextrous hand Utah/MIT hand. Both the arm and hand are controlled via Chimera, a real-time operation system for sensor-based control developed at CMU [28]. There is the same working procedure in the robot world as in the human world except that the last step is to evaluate the robot action based on the TSDT.

4 The acquisition of perceptual data streams

The robot system should learn not only the initial poses and destinations of the objects and the orders of assembly, but also should perceive how objects move for replicating the task. For this purpose, the system has to perceive the trajectories of all objects with orientations at each point in the sequence for forming perceptual data streams. This is sufficient for simple tasks and structured environment. In general, the system has to track both objects and the human hand in able to robustly and completely understand a given task for the purpose of efficiently replicating the task.

Otherwise, system will encounter difficulties, especially with partial occlusion case and motions like screw-turning a bolt with a cylindrical head. After tracking, the perceptual data streams of each object and each part of the hand are created for interpretation. In this section, we first briefly introduce multiple objects as well as hand tracking based on the 3-D template matching algorithm [35], then describe the representation of perceptual data streams.

4.1 Perceptual data stream for interpretation

Assume in a workspace, as shown in Figure 3, there are V_X rigid objects to be assembled and U_H rigid parts of the hand to be used to manipulated. The *v-th* manipulated object denoted as X^v with a coordinate system attached to its moment centroid $\mathbf{X^v}$. The *u-th* rigid part of the hand is denoted as H^u with a coordinate system **H**. Thus, the workspace can be expressed as:

$$Workspace = \left\{ H^u, X^v \middle| \begin{matrix} u = 0, 1, 2, \ldots, U_H, \\ v = 0, 1, 2, \ldots, V_X \end{matrix} \right\} \tag{1}$$

X^0 is defined to be a static object on which the world frame **W** is attached. From now on, we define a table as this static object.

Suppose the transformations of the hand and objects between the current frame and its previous frame with respect to **W** are known, denoted as $T_H(i, i-1)$ and $T_X(i, i-1)$ respectively. Then, at the *k-th* frame the transformations of the hand and the objects with respect to **W** can be computed:

$$T_H(k) = T_H(0,0) \prod_{i=1}^{k} T_H(i, i-1) \tag{2}$$

$$T_X(k) = T_X(0,0) \prod_{i=1}^{k} T_X(i, i-1) \tag{3}$$

$T_H(0,0)$ and $T_X(0,0)$ are defined as the initial transformations of the hand and the object with respect to **W**.

Thus, the set of their transformation sequence with respect to the world coordinate system **W** is expressed by:

$$T = \left\{ T_{H^u}(k),\ T_{X^v}(k) \quad \middle| \begin{matrix} k = 0, 1, 2, \ldots, k_e \\ u = 0, 1, 2, \ldots, U_H \\ v = 0, 1, 2, \ldots, V_X \end{matrix} \right\} \tag{4}$$

In the following, we will discuss how to compute the transformations $T_H(i, i-1)$ and $T_X(i, i-1)$.

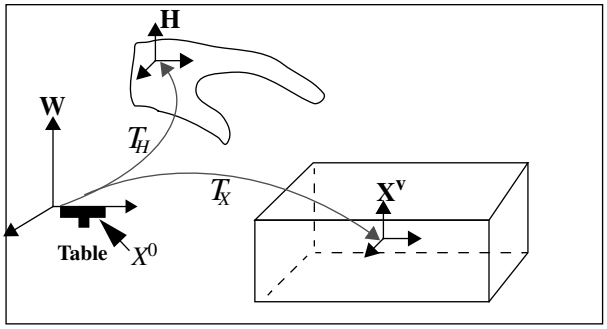


Figure 3 The transformations between the world and objects' frames

4.2 Model-based localization algorithm

The 3-D template matching algorithm (3DTM) is able to localize a rigid object accurately under a given initial pose. The object models are created using Vantage, a geometric modeler developed at Carnegie Mellon University [9]. 3DTM uses 3D templates consisting of sets of points sampled from surfaces of models for object localization. The constraint on the templates is that visible points on the model surface match range data points in the image. Suppose $q = (q_t, q_\theta)^T$ to be the vector of model parameter where $q_t = (q_x, q_y, q_z)^T$ is the vector of translation parameters and $q_\theta = (q_s, s_u, q_v, q_w)^T$ is the vector of rotation parameters using the quaternion representation [4]. The error, $z_i(q)$, to be the distance between the model point and the data point nearest the model point, is given by:

$$z_i(q) = \min_{\vec{a} \in D} \left\| \vec{x}_i(q) - \vec{a} \right\|$$

where *D* is the set of three-dimentional data points in the image, and $\vec{x}_i(q)$ is the world coordinate of the *i-th* model point transformed using the model parameters *q*. A MAP estimate of *q* is found out by minimizing the energy function:

$$E(q) = \sum_{i \in V(q)} \rho\left(\frac{z_i(q)}{\sigma}\right) \tag{5}$$

where *V(q)* is the set of visible model points for the model parameters *q*, $\rho(\cdot)$ a Lorentzian error distribution:

$$\rho\left(\frac{z}{\sigma}\right) = \log\left(1 + \frac{1}{2}\left(\frac{z}{\sigma}\right)^2\right)$$

The Lorentzian distribution is similar in shape to the Gaussian except that large errors are assumed to occur with a higher relative probability than in the Gaussian error model. This makes the larger error (due to the occlusion or noise) effect little on the estimate result.

The 3DTM algorithm can be given symbolically as following:

. Input depth data D, 3DTM model, initial pose q.

. Create k-d tree from *D*.

. Repeat:

(Minimizing Energy Computation)

Get visible model points: V(q)

Current energy: $E_0 = E(q)$

Translation gradient: $\vec{dq} = -\frac{\partial E}{\partial q_t}$

Step size: $\Delta q = (\Delta q_t, \Delta q_\theta) = \left(\frac{\vec{dq}}{|\vec{dq}|} \cdot 1mm, 0\right)$

While $E(q + \Delta q) < E(q)$ $q = q + \Delta q$

Rotational gradient: $\vec{dq} = -\frac{\partial E}{\partial q_\theta}$

Step size: $\Delta q = (\Delta q_t, \Delta q_\theta) = \left(0, \frac{\vec{dq}}{|\vec{dq}|} \cdot 1^\circ\right)$

While $E(q + \Delta q) < E(q)$ $q = q + \Delta q$

Until $E(q) = E_0$,

Until no improvement is made.

. Output final pose q.

This method has a reasonable convergence time and produces accurate results. The tracking method based on this algorithm is just using the object pose at previous frame as the initial pose of the current frame for localizing the final pose of the object at this frame which will be the initial pose of the next frame.

4.3 Tracking multiple objects

Using 3DTM to tracking one object is easier than tracking multiple objects because multiple objects bring out some serious occlusion problem. The 3DTM program is robust to some degree of the object occlusion, significant occlusion may occur in the course of the task performance, and the program does not work well or fails. The same problems occur in hand tracking. So we take the same steps as [12] to mask out range pixels corresponding to the objects which are not being manipulated. Before doing that, the system has to detect the occlusion (see Section 5) while it is tracking objects. In practice, we have found that the algorithm will typically converge to the global minimum with up to 50% of the object occluded. If a object is occluded more than 50% of its size, its pose will be determined by the constraints or prior knowledge.

The description of using 3DTM to tracking all these objects is presented below:

(1) Input depth image sequence, 3DTM models.

(2) Determine the poses of the objects in the *zero-th* depth image as the initial poses of object tracking. $k = 0$, $k = k+1$, $v = 0$.

(3) Load k^{th} depth image D and create k-d tree from D.

(4) Load the *v-th* model and its pose in the *(k-1)-th* depth image as the initial pose of 3DTM. and $v = v+1$.

(5) Repeat minimizing energy computation

(6) Output the model pose $T(k, k-1)$ in this frame, which is the transformation from *(k-1)-th* frame to the *k-th* frame

(7) Go to step 4 until the poses of all models are estimated in the depth image.

(8) Go to step 2 until the poses of all models in all depth image sequence are estimated.

The system can not track objects in real-time due to the large amount of image data and the computational requirements of our vision algorithms. Currently, the time to recover a depth image with 8-bit accuracy from four intensity images of size 512x486 to tracking objects in the frame is less than 50 seconds in workstation.

Figures 4, 5 and 6 show the tracking results, where are 5 rigid objects in the workspace, including two rigid fingers. Because the robot thumb and index fingers are black and invisible for the stereo system, so we put on white caps for stereo sensing.

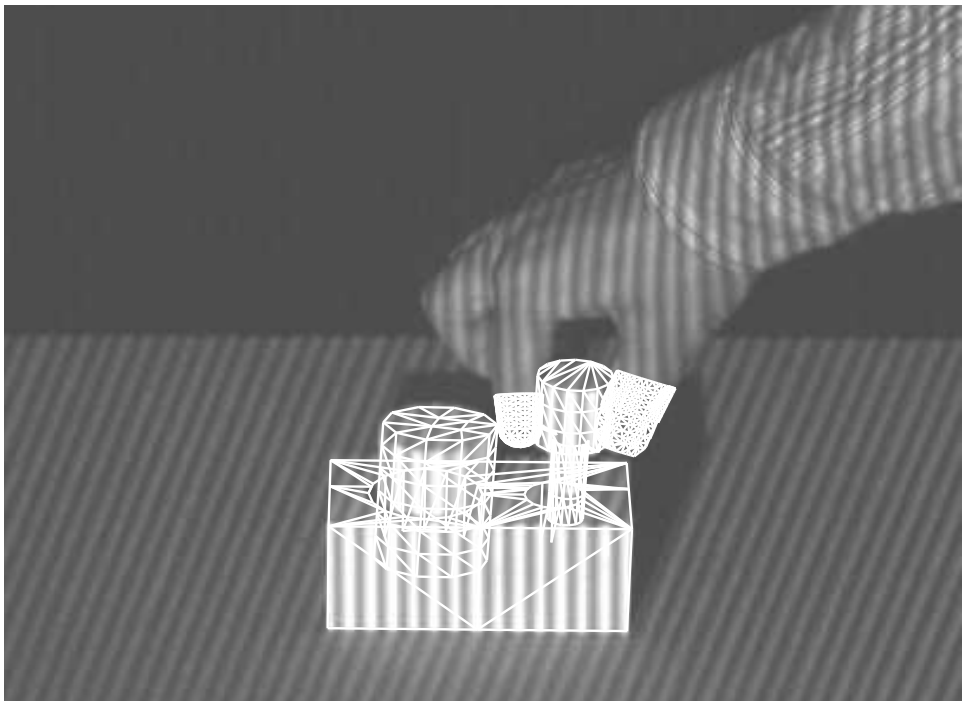


Figure 4 Human hand turing bolt

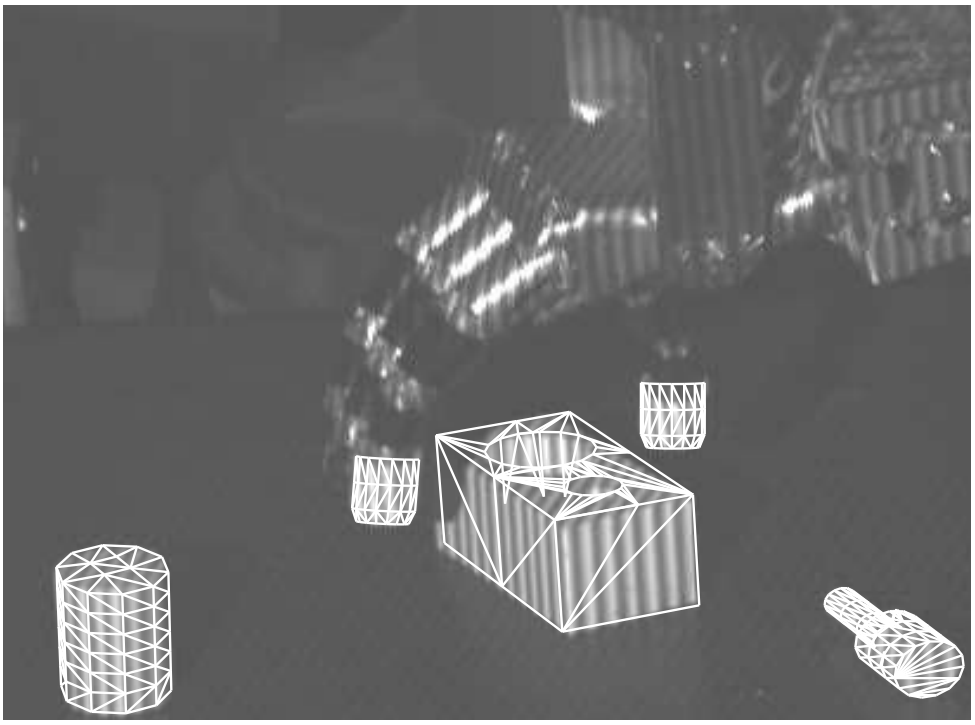


Figure 5 Tracking robot fingers and objects

4.4 Hand tracking

There is a lot of work [15] [24] on tracking human hand for hand gesture recognition, few work for hand action interpretation. Tracking human hand actions, especially hand fine motion, is not easy because the hand is an articulated structure with about 30 degrees of the freedom and with frequent self-occlusion. Moreover, the joint movements result in the hand shape changing in various ways and the surface around the joints significantly deformed. These make the tracking whole hand much complicated. In the section 4.3, we use 3DTM to track the fingers of the hand successfully only under the small displacement, e.g. less than 50% of the finger size. This means finger speed should be limited in 6mm/frame. In addition, fingers can not be occluded more than 50% of its size. To solve such problems we segment the hand into sixteen parts, palm and three parts for each finger connected by joints. Each one can be approximately by a rigid body, as shown in Figure 7. The similar models see [24][16][25].

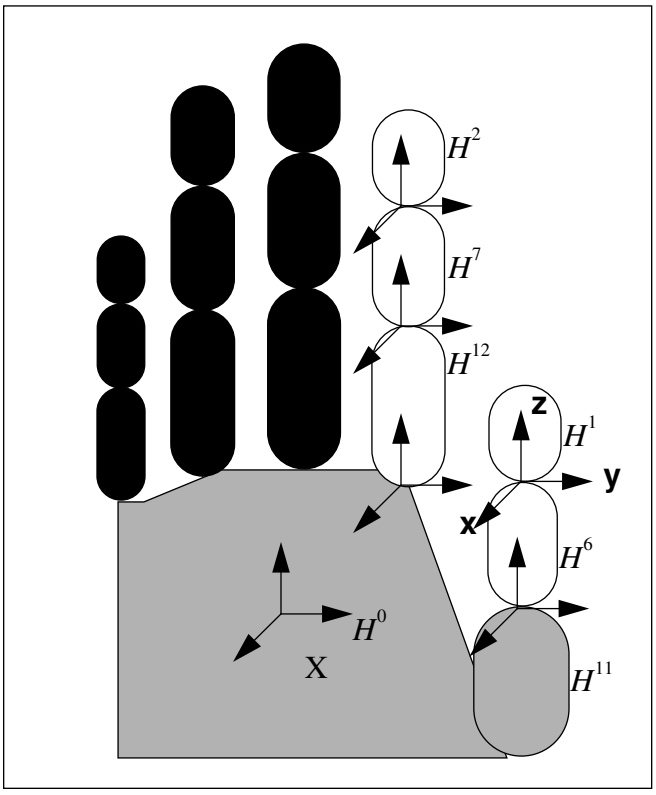


Figure 6 The human hand model

In practice, we are able to use index and thumb to perform much complicated task, so we suppose only to use the thumb and index to manipulate objects for simplicity. Thus, it is sufficient that the system just track each segment of these two fingers for hand action interpretation. In fact, because each segment of the fingers is too short and cylinder-like shape, it is difficult to find the accurate position and orientation, especially in significant displacement or occlusion occurring. Thus we select the palm H^0 and the thumb metacarpal H^{11} together as an anchor (shadow part in Figure 7) for roughly estimating the pose of fingers because the back of the palm can be approximately regarded as a rigid object with bigger size. Moreover, H^0 and H^{11} together can form a shape with a stable geometric feature in the thoroughly task performing for robust tracking. Therefore, for tracking the thumb and index, the system has to track 6 rigid parts which constitute a simplified hand model involving the palm and thumb metacarpal H^0 and H^{11}, the proximal phalanxes of the thumb and index H^6 and H^{12}, the middle phalanx and distal phalanx of the index H^7, H^2 and H^1, as shown in Figure 7 except black parts.

We use vision-based modelling method [34] to the human hand. The modelling procedures are following:

1 . Take image sequence from all views of the model by the four-camera multibaseline stereo vision system.

2 . Recover depth images and estimate poses to form the surface patches of the model.

3 . Alignment of surface patches

4 . Merging of Surface patches by shrinking the over-lapping region of 2-patches until they meet and add triangles to link them together

5 . Segment the hand model into sixteen rigid bodies.

6 . Convert the models with mesh representation into 3DTM model.

The following is tracking steps

1 . Tracking palm

2 . Tracking the proximal of the thumb which has two freedoms, rotations around x and y axis while other four freedoms are given by the pose of the palm.

3 . Tracking the distal of the thumb which only has one freedom, rotation around x, while other five freedoms are given by the pose of the proximal

4 . Tracking the proximal of the index which has two freedoms, rotations around x and y while other four freedoms are fixed by the pose of the palm.

5 . Tracking the middle of the index which only one freedom, rotation around x while other five freedoms are fixed by the pose of the proximal.

6 . Tracking the distal of the index which has one freedom, rotation around x while other five freedoms are given by the pose of the middle

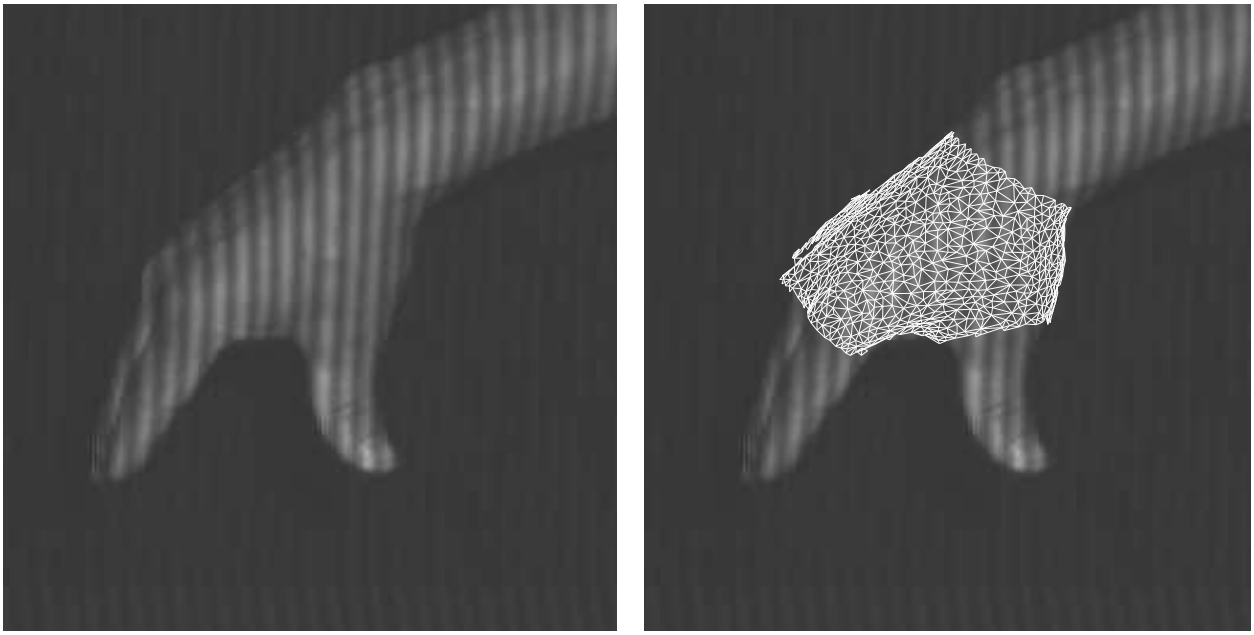


Figure 7 The palm model as tracking anchor

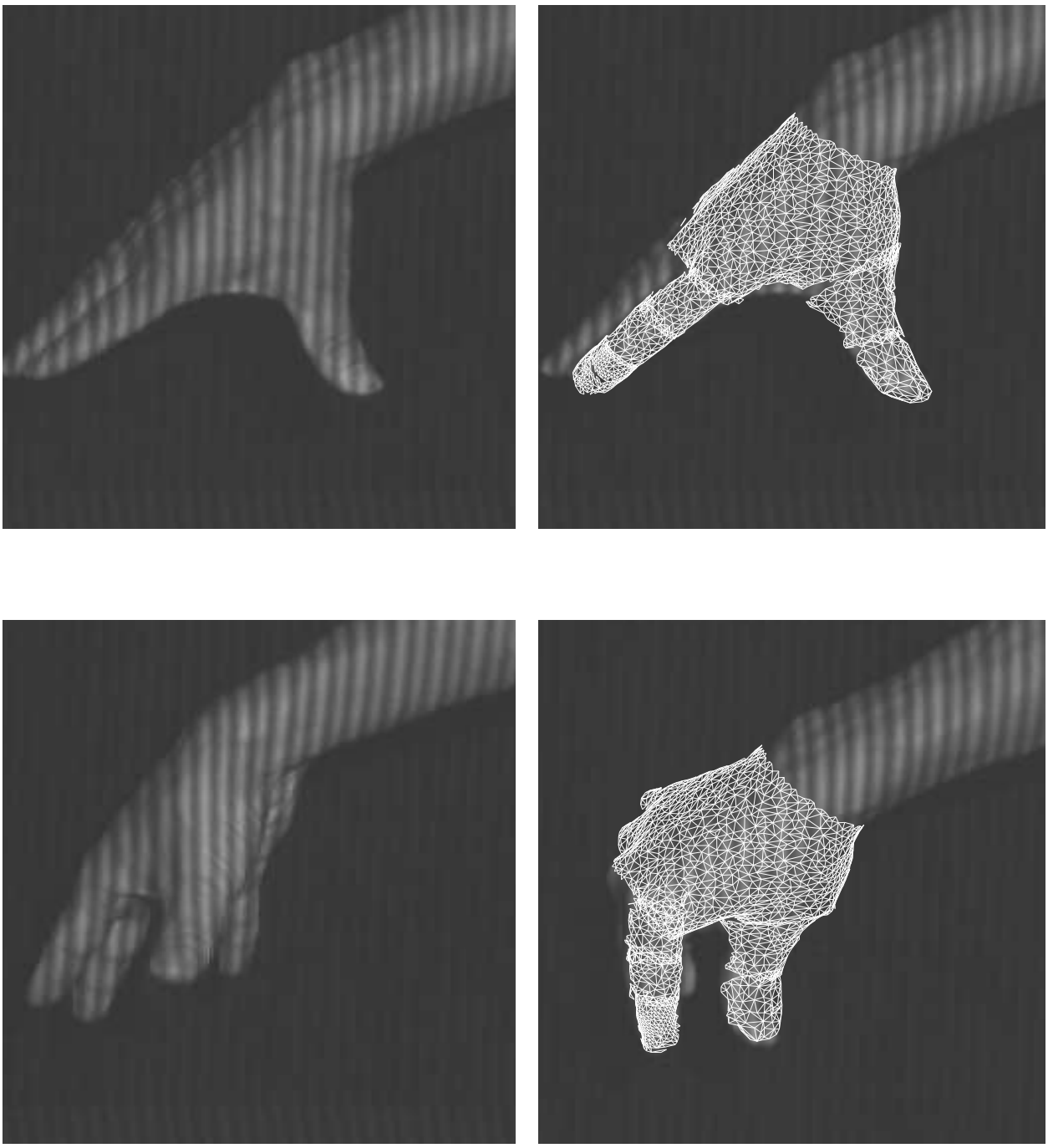


Figure 8 Tracking hand grasping action

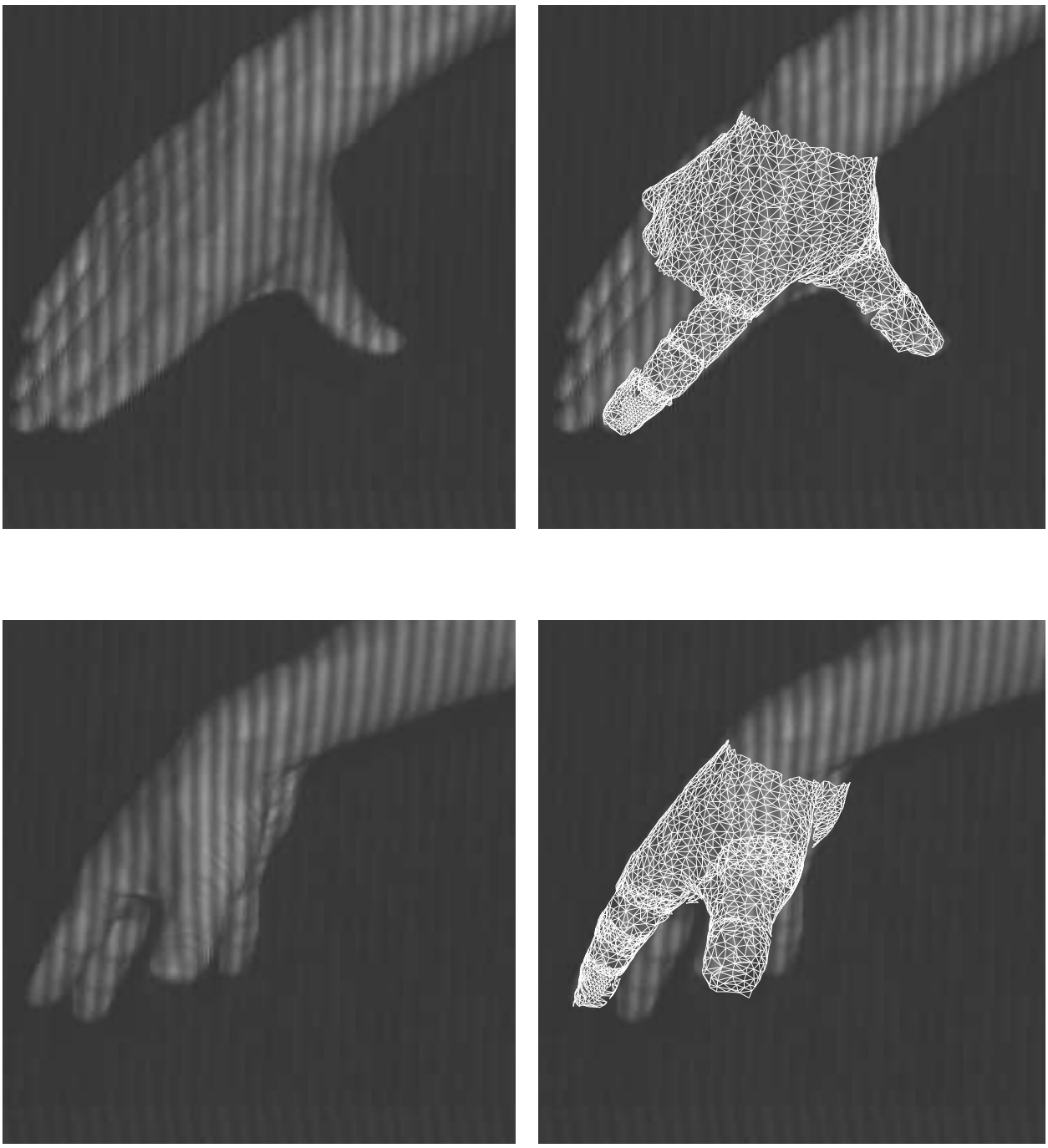


Figure 9 Tracking hand rotation action

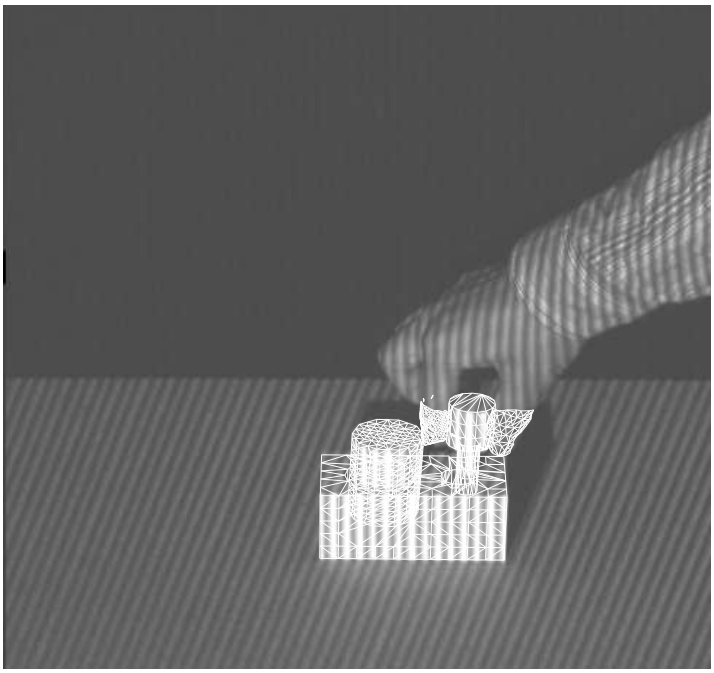


Figure 10 Tracking human finger with the real finger models

5 Spatial relationships in workspace

A system for task understanding must include three-dimensional spatial reasoning and time sequence analysis as a primary component, because relationships among objects in space and time are a rich source of knowledge and constraints for task interpretation. In this section, we describe the representation of the motion of the human hand and objects and their spatial relationships.

The shape of a rigid object can be represented by primitives, such as points, lines, curves, planes and curved patches. Such primitives can be used for object recognition and localization, shape recovery and classification. In our system for assembly work, the models of the objects in the workspace are known as well as the model of the instructor hand. For finding out the relations among the hand and objects, we use the set of planar and curved patches as a high level representation to describe the shape of an object for representing object relations, such as contact and occlusion. We use the points on each patch as a low level representation to express the relations among objects quantitatively. For example, we can efficiently test whether individual points are close enough to be considered contacting, and whether two points occluded one another. Thus the *u-th* rigid H^u of the hand and the *v-th* object X^v (Figure 11) can be expressed as sets of patches

(denoted by symbols *i* and *s*) and points (denoted by symbols *j* and *t*) on those patches in their own coordinate systems:

$$H^u = \{H_i^u\} = \left\{ H_{ij}^u \;\middle|\; \begin{matrix} i = 0, 1, 2, \ldots, I_u - 1 \\ j = 0, 1, 2, \ldots, J_u - 1 \end{matrix} \right\} \tag{6}$$

$$X^u = \{X_s^v\} = \left\{ X_{st}^v \;\middle|\; \begin{matrix} s = 0, 1, 2, \ldots, S_v - 1 \\ t = 0, 1, 2, \ldots, T_v - 1 \end{matrix} \right\} \tag{7}$$

From now on, we use H to denote the whole hand, H^u the *u-th* part of the hand, and H_i^u the *i-th* point of the *u-th* part of the hand.

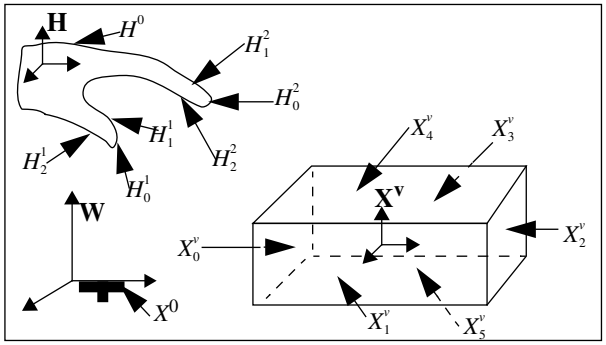


Figure 11 The representation of a hand and objects

5.1 Relations between hand and all other objects

Suppose the initial poses of all objects and the hand with respect to **W,** $T_H(0)$ and $T_X(0)$ (see Eq. (2) and (3)). It is easy to calculate the surface points of all objects and the hand at the *k-th* frame with respect to **W** as follows:

$$\mathcal{H}_{st}^u(k) = \{ T_{H^u}(k) \; H_{st}^u \} \tag{8}$$

$$\mathcal{X}_{ij}^v(k) = \{ T_{X^v}(k) \; X_{ij}^v \} \tag{9}$$

Suppose $(x_{ij}^{u}, y_{ij}^{u}, z_{ij}^{u})$ and $(p_{st}^{v}, q_{st}^{v}, r_{st}^{v})$ are the point coordinates of the hand and the *v-th* object. We define the relationship between the hand and an object as follows:

$$\begin{aligned} R_{H^u X^v}(k) &= \left\{ \mathcal{H}_{ij}^{u}(k) - \mathcal{X}_{st}^{v}(k) \right\} \\ &= \left\{ \left(x_{ij}^{u} - p_{st}^{v}, y_{ij}^{u} - q_{st}^{v}, z_{ij}^{u} - r_{st}^{v} \right) \right\} \end{aligned} \tag{10}$$

From the Eq. (10), we could define the nearest Euclidean distance between the hand and the *v-th* object in the *k-th* frame

$$R_{HX^v}^{distance}(k) = min \left\| \mathcal{H}_{ij}^{u}(k) - \mathcal{X}_{st}^{v}(k) \right\| \tag{11}$$

If $R_{HX}^{distance}(k)$ is equal to zero, the hand and the object contact with each other. We can say (the *j-th* point of) the *i-th* surface of the hand contacts (the *t-th* point of) the *s-th* surface of the object.

The contact area is:

$$R_{H^u X^v}^{contact-area} = c \times \left| C_{H^u X^v} \right| \tag{12}$$

where c is the area factor and $\left| C_{H^u X^v} \right|$ is the number of the set of contact points

$$C_{H^u X^v} = \left\{ \left(\mathcal{H}_{ij}^{u}(k), \mathcal{X}_{st}^{v}(k) \right) \middle| R_{H^u X^v}^{distance}(k) = 0 \right\} \tag{13}$$

From Eq. (10), (11), and (12), we can tell how fingers contact the manipulated object and which parts of the object is contacted. These are very important for mapping the observations into the robot world.

5.2 Occlusion Relations

As we know the hand partially occlude an object while it is grasping the object. The tracking algorithm can work well even the object is occluded 50%. Still, detecting occlusion as well as calculating the occlusion area are necessary for making the tracking algorithm robust. Suppose the view direction is parallel to the Z axis of the world coordinate system and the view scene size is $M \times N$ and pixel point is denoted by *(I, J)* with the zero initial values of the pixels.

Map the 3-D points of an object in the world coordinates to 2-D view points in the view scene by orthographically projecting. Then visible point sets of surfaces of objects and the hand without being occluded by other object are given by:

$$\begin{aligned} R_{H^u}^{view} &= \{z_{ijIJ}^u\} \\ &= \{\max_{i,j}\{\ z_{ij}^u,\ \mid x_{ij}^u = I, y_{ij}^u = J\}\} \end{aligned} \tag{14}$$

$$\begin{aligned} R_{X^v}^{view} &= \{\{r_{ijIJ}^v\}\} \\ &= \{\max_{i,j}\{\ r_{ij}^u,\ \mid p_{ij}^u = I, q_{ij}^u = J\}\} \end{aligned} \tag{15}$$

The occluded point set of the *u-th* part of the hand by the *v-th* object is given by:

$$R_{X^vH^u}^{occluded} = \{\{z_{ijIJ}^u\} \mid\ r_{ijIJ}^v > z_{stIJ}^u\} \tag{16}$$

If z_{ijIJ}^u exists, we can say, in the view scene, the *j-th* point of the *i-th* patch on the *u-th* object occludes the *t-th* point of the *s-th* patch on the object.

The occluded area is computed by:

$$R_{X^vH^u}^{occluded-area} = c \times \left| R_{X^vH^u}^{occluded} \right| \tag{17}$$

where $\left| R_{X^vH^u}^{occluded} \right|$ is the number of occluded points

We can also compute $R_{HX}^{occluded}$ and $R_{HX}^{occluded-area}$ in the same way mentioned above. This occlusion analysis is general case, it does not need to assume the object shapes are convex, nor how they occlude each other.

5.3 Spatial relationships among objects

The assembly goal is to locate and fixture workpieces or objects together at a desired configuration. As the operation proceeds, the spatial relationships among the objects are changing until the goal configuration. Thus detecting the configuration at any time or any state is a prominent problem in the automatic assembly domain. Asada and Hirai [2] present a kind of symbolic descrip-

tion of the geometric constraints among workpieces for guiding the moving part to the desired destination. They set up the list of all the contacting pairs determined by the position and orientation of one workpiece relative to the other. Suehiro and Ikeuchi [29] present a face contact relation graph to express face contact relation among objects. The root arrow node in the graph is the manipulated object, which is moved by the assembly operation to generate the face contact relation, while the end arrow node is the environment object, which is stationary during the assembly operation. the goal configuration of each assembly operation is attached to the node corresponding to the manipulated object of the operation. Here we present a hierarchical structure to describe the spatial relationships among objects based on the object representation mentioned in the previous part. The Figure 12 shows the hierarchical structure of objects contact relation

Suppose there are n objects in the workspace. Each object has m_i patches. The contact relations in the workspace are not more than $\prod_{i=0}^{n-1} \frac{m_i n\,(n-1)}{2}$. The following is the general structure used in our system

.

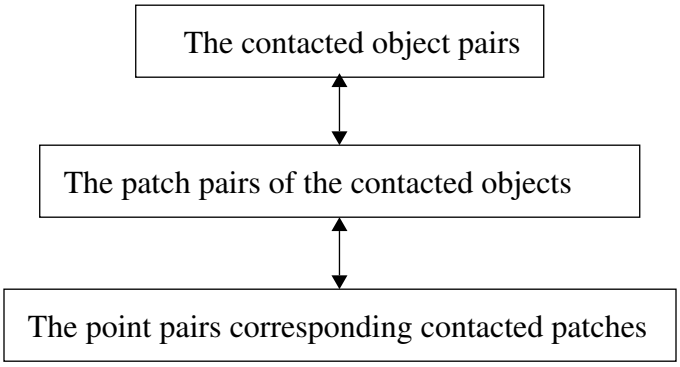


Figure 12 The hierarchical structure of objects contact relation

For instance, there are two objects placed on the table in the workspace as shown in Figure 13.

Figure 13 (a):

First level: $Assembly = \{\left(X^0, X^1\right), \left(X^0, X^2\right)\}$

Second level: $\left(\left(X^0, X^1\right) = \{\left(X^0, X_0^1\right)\}\right), \qquad \left(\left(X^0, X^2\right) = \{\left(X^0, X_0^2\right)\}\right)$

Third level $\left(\left(X^0, X_0^1\right) = \{\left(X_i^0, X_{0,j}^1\right)\}\right), \qquad \left(\left(X^0, X_0^2\right) = \{\left(X_i^0, X_{0,j}^2\right)\}\right)$

where i and j is the coordinate of the points on the zero-th patches on the first and second object respectively.

Figure 13 (b):

First level: $Assembly = \{\left(X^0, X^1\right), \left(X^0, X^2\right), \left(X^1, X^2\right)\}$

Second level: $\left(X^0, X^1\right) = \{\left(X^0, X_0^1\right)\}, \qquad \left(X^0, X^2\right) = \{\left(X^0, X_0^2\right)\}, \qquad \left(X^1, X^2\right) = \{\left(X_2^1, X_1^2\right)\}$

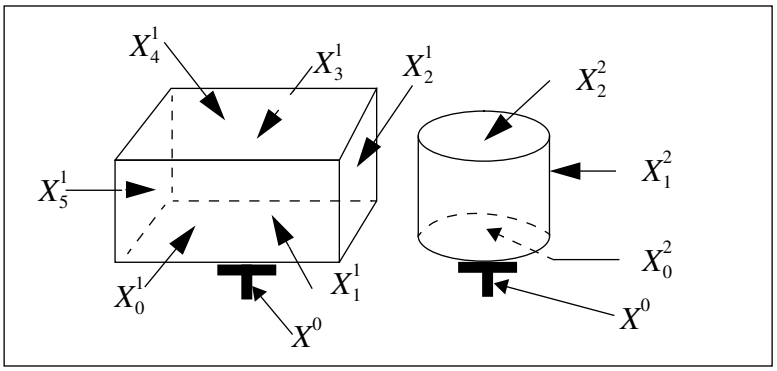
Third level: $\left(X^0, X_0^1\right) = \{\left(X_i^0, X_{0,j}^1\right)\}, \qquad \left(X^0, X_0^2\right) = \{\left(X_i^0, X_{0,j}^2\right)\}, \qquad \left(X_2^1, X_1^2\right) = \{\left(X_{2,s}^1, X_{1,t}^2\right)\}$

Figure 13 (c)

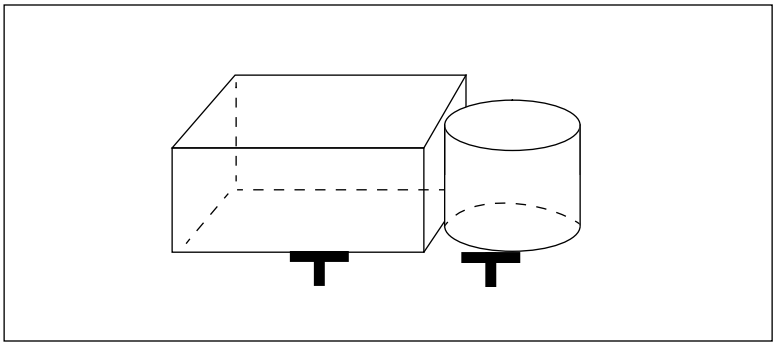
First level: $Assembly = \{\left(X^0, X^1\right), \left(X^1, X^2\right)\}$

Second level: $\left(X^0, X^1\right) = \{\left(X^0, X_0^1\right)\}, \qquad \left(X^1, X^2\right) = \{\left(X_2^1, X_1^2\right)\}$

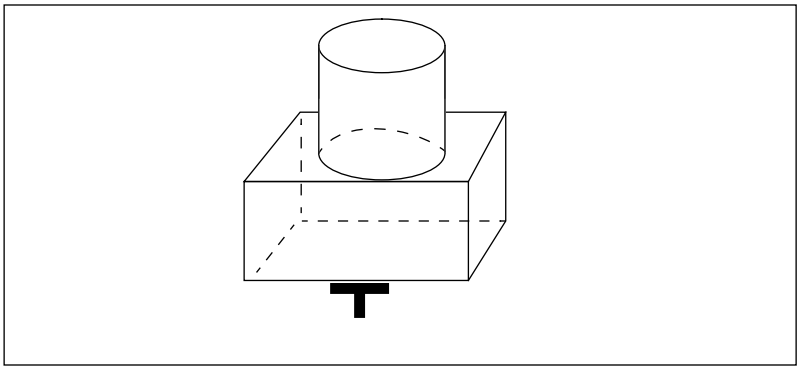
Third level: $\left(X^0, X_0^1\right) = \{\left(X_i^0, X_{0,j}^1\right)\}, \qquad \left(X_2^1, X_1^2\right) = \{\left(X_{2,s}^1, X_{1,t}^2\right)\}$



(a)



(b)



(c)

Figure 13 The contact relation representation

6 Hand action interpretation

6.1 Assembly task description

We use the model-based data stream to describe the workspace. A simple way to map the human world into robot world is that the robot replicates the demonstrated task just by using the raw data stream collected by perception system. This is true if the perception system and execution system are working much more precisely, such as teleoperation and industrial robot in the structure environment. But in general, the human world is not exactly the same as the robot world or they are working in completely different environments. In order to make the system more flexible and robust, the system has to interpret the data stream as the robot control commands for performing the demonstrated task. Before interpreting, the segmentation of the task should be done first. Zak and Das [36] segment the task into three parts in their training teleoperation system. First, there is only motion in free space, then the robot end effector and the attached tool are moving in contact with environment and finally there is motion in free space again. Kang and Ikeuchi [13] developed a task segmentation based on the hand finger motion wearing a dataglove. They divided a task involving grasping action into pregrasp, grasp and manipulation phase. The feature used are the fingertip polygon area (the finger polygon being the polygon whose vertices are the fingertips), hand speed, and the volume sweep rate, which is the product of the first two.

For manipulating an object in an assembly task, there are four clearly distinguishable phases, approaching the object, contacting, manipulating and departing from this object. We refer to these four consecutive phases as a subtask of an assembly task. The subtask can be easily recognized by finding three segmentation points, named contact, start-manipulate, and end-manipulate, using the relationships between hands and objects at each frame (see Section 5). This is based on the observation from a teacher who makes his each action very clear to students and always resolves the complex action into simple action for being understanding easily. For example, he could not turn a bolt into a threaded hole directly. Instead, he might pick up a bolt, insert it into a threaded hole, then grasp the bolt again and turn it into the hole step by step.

From the initial frame (Start) to the end frame (End) in an assembly task, there might be many objects corresponding many suggests, each subtlest involves many hand actions. For instance, from the starting point to contacting point, there is the approaching phase which may have a different trajectory with different orientations for different purposes. Figure 14 shows the state diagram of an assembly task; circles are used to denote the segmentation points and blocks represent

the phases. After task segmentation and action interpretation, the system generates a task sequence as following:

$$\begin{aligned} Task &= \{\langle Start, Subtask_w, End\rangle \mid (w = 1, 2, \ldots, n)\} \\ Subtask_w &= (Approach_w, Pre-Manipulate_w, Manipulate_w, Depart_w) \end{aligned} \tag{18}$$

The task sequence contains n subtasks, each subtask consisting of four actions, approach, pre-manipulate, manipulate, and depart.

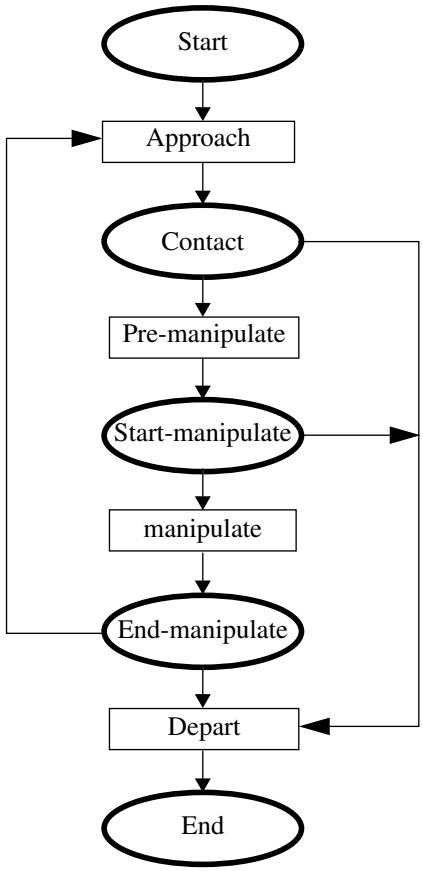


Figure 14 State diagram of an assembly task

6.2 Task Segmentation

As mentioned above there are four manipulating phases divided by three distinct states: contact, start-manipulate, and end-manipulate. In the following, we detail the segmentation and interpreta-

tion of the perceptual data stream from a work space (Eq. (8) and (9)). For simplicity, we have assumption as follows:

1. All data is accurate and correct.

2. All of the models of objects and the hand with their initial poses are known.

3. All of objects are not occluding with each other at the starting frame.

4. Only one object is being manipulated at a time.

. Contact

The instant k_c is the frame at which at least one finger of the hand makes contact with an object while before k_c, the contacting point set ${}_{C_{H^u X^v}}$ is empty (denoted by symbol Φ). The hand-object relation at this instant will be defined by Eq. (13).

$$\begin{aligned} Contact_w = \{ & T_{H^u}\left(k_c^w\right), T_{X^v}\left(k_c^w\right) \Big| \\ & \left({}_{C_{H^u X^v}}\left(k_c^w\right) \neq \Phi \right) \wedge \left({}_{C_{H\ X^v}}(k) = \Phi \right), \quad k = k_c^w - 1,\ k_c^w - 2, \ldots,\ k_{em}^{w-1} + 1 \ \} \end{aligned} \tag{19}$$

At this instant, the contacted object X^v is referred to as the manipulated object.

. Start-Manipulate

The frame k_{sm}^w, at which manipulation starts, occurs when the manipulated object is about to be moved along with the hand from its previous stable state after being contacted.

$$\begin{aligned} Start-Manipulate_w = & \left\{ \left(T_H\left(k_{sm}^w\right), T_{X^v}\left(k_{sm}^w\right) \right) \Big| \right. \\ & \left(\dot{\mathcal{H}}^u\left(k_{sm}^w\right) = \dot{X}^v\left(k_{sm}^w\right) \neq 0 \right) \wedge \left(\begin{array}{c} \dot{X}^v(k) = 0 \\ k = k_{sm}^w - 1, k_{sm}^w - 2, \ldots, k_c^w \end{array} \right) \wedge \left({}_{C_{H^u X^v}}(k) \neq \Phi \right) \left. \right\} \end{aligned} \tag{20}$$

Where $\dot{X}^v\left(k_{sm}^w\right), \dot{\mathcal{H}}^u\left(k_{sm}^w\right)$ are the velocity of the *v-th* object and the hand at the frame k_{sm}^w.

. End-Manipulate

The frame k_{em}^{w} at which the hand stops manipulating the object occurs when the hand is releasing the manipulated object and departing from it.

$$End\text{–}Manipulate_{w} = \{\left(T_H\left(k_{em}^{w}\right), T_{X^v}\left(k_{em}^{w}\right)\right) \Big| \left(C_{H\ X^v}\left(k_{em}^{w}\right) = \Phi\right) \wedge \left(C_{H^u X^v}(k) \neq \Phi\right),\ k = k_{em}^{w} - 1, k_{em}^{w} - 2, \ldots, k_{sm}^{w} \ \} \tag{21}$$

6.3 Action interpretation

Generally speaking, the hand action is very complex when it manipulates an object. In some cases, we could not describe it and neither transfer it to other persons, for example, to shake an old key gently for opening the locker, to scratch a statutes. This is a problem which is concerning about training and learning domain. In our work, we only discuss explicit actions such as moving, pushing, dropping, etc. We have mentioned in the last part there are three dominant points for segmentation during manipulation, that is, contact, start-manipulate and end-manipulate. Between each state, there might be one or one more possible actions. After segmentation, we will extract meaningful action with compact parameters for robot to replicate the same action for performing the demonstrated tasks.

. Approach

Between the starting frame (or the departing frame of the last frame) and the contacting frame is the approaching phase. Before contacting the manipulated object, we have more than one approach trajectory because the system can not recognize which object will be manipulated when there are more than one object in front of the hand. In our system, we calculate relationships between the hand and the objects in the workspace in order to detect which object is manipulated and which one is touched or occluded. At the k_c-*th* frame, the *v-th* object is contacted, that is:

$$Approach_{w} = \{\ T_{H^u}(k),\ T_{X^v}(k) \mid \quad k = k_{em}^{w-1},\ k_{em}^{w-1} + 1 \ldots, k_c^{w} - 1\ \} \tag{22}$$

Where $w=1$ indicates starting frame and $w>1$ means beginning with the departing frame of the last manipulated object.

. Pre-manipulate

After contacting the manipulated object at the end of the approach phase, we differentiate two types of relations between the hand and the manipulated object called touch (Figure 15a and b) and grasp (Figure 15c). The actions during the manipulate phase can then be analyzed using this differentiation.

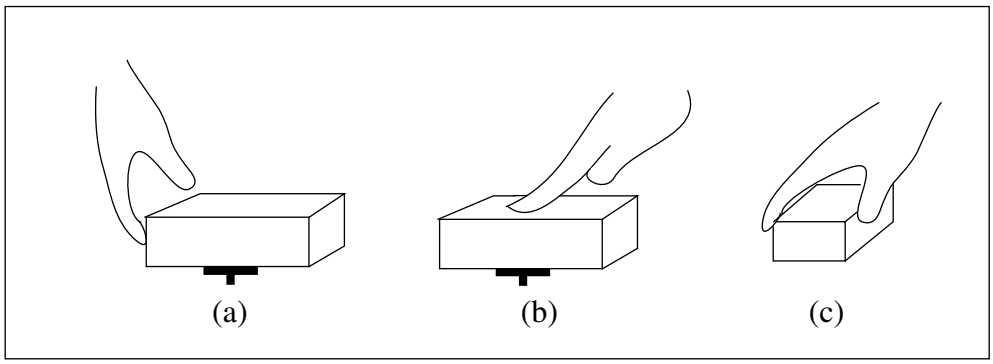


Figure 15 Touch and Grasp

(1) Touch

If only one finger or more fingers with aligned together touch an object, we classify it as a touch manipulation. In general, assume that there are n parts of the hand contacting the object with a contact unit normal $\overrightarrow{n_{H^v}}$ for each part. Thus, the touching action can be defined as

$$\begin{aligned} Touch_w = \{ & \left(T_{H^u}(k),\, T_{X^v}(k) \right) \Big| \\ & \forall A^i \geq 0, \sum_i A^i > 0 \Rightarrow \left(\sum_i A^i \cdot \overrightarrow{n_{H^i}} \neq 0 \right), \\ & k = k_c^w, k_c^w + 1, \ldots, k_c^w \quad i = v_1, v_2, \ldots, v_n \} \end{aligned} \tag{23}$$

where A^i is a coefficient of the contact normal.

(2) Grasp

On the other hand, when there are multiple contacts between the fingers and the object and the forces along the contact normal are at equilibrium, we define the action to be a grasp.

$$
\begin{aligned}
Touch_w = \{ & \left(T_{H^u}(k),\ T_{X^v}(k) \right) \Big| \\
& \exists A^i \geq 0,\ \sum A^i > 0 \wedge \left(\sum_i A^i \cdot \overrightarrow{n_{H^i}} = 0 \right), \\
& k = k_c^w, k_c^w + 1, \ldots, k_c^w \quad i = v_1, v_2, \ldots, v_n \ \}
\end{aligned}
\tag{24}
$$

. Manipulate

We classify the actions in the manipulate phase (from frame k_{sm} to frame k_{em}) into several distinct assembly actions: push, push-pull, pickup, screw and drop. These are defined based on the contact relations.

(1) Push

When the fingers of the hand touch the manipulated object while it maintains contact with the work-table, we define the manipulate action as a push.

$$
\begin{aligned}
Push_w = \{ & \left(T_{H^u}(k),\ T_{X^v}(k) \right) \Big| \\
& \left(C_{X^0 X^v}(k) \neq \Phi \right) \wedge Touch\left(k_{sm}^w \right), k = k_{sm}^w, k_{sm}^w + 1, \ldots, k_{em}^w \ \}
\end{aligned}
\tag{25}
$$

(2) Push-Pull

When the fingers grasp the manipulated object while it maintains contact with the work-table, we define the action to be a push-pull.

$$
\begin{aligned}
Push - Pull_w = \{ & \left(T_{H^u}(k),\ T_{X^v}(k) \right) \Big| \\
& \left(C_{X^0 X^v}(k) \neq \Phi \right) \wedge Grasp\left(k_{sm}^w \right), k = k_{sm}^w, k_{sm}^w + 1, \ldots, k_{em}^w \}
\end{aligned}
\tag{26}
$$

(3) Pick-Up

When the hand grasps the manipulated object and moves it such that it makes contact with other assembled objects, we define the action to be pick-up.

$$\begin{aligned} Pick-up \;=\; & \{\left(T_{H^u}(k),\, T_{X^v}(k) \right) \Big| \\ & \left(C_{X^0 X^v}(k) \;=\Phi \right) \wedge Grasp\left(k_{sm}^{w} \right),\, k = k_{sm}^{w}, k_{sm}^{w}+1, \ldots, k_{em}^{w} \} \end{aligned} \tag{27}$$

(4) Screw

The screw motion is a repetitive motion. Thus it is natural to use repetitive feature to detect such manipulation. We use the sum of the coupled velocity moments (SCVM) of the index and thumb of the hand to represent the repetitive feature, which is expressed by:

$$SCVM \;=\; \sum_{i=0}^{1} \sum_{k=0}^{N} \left(\left(\vec{r}_i(k) - \vec{c}(k) \right) \times \left(\vec{r}_i(k+1) - \vec{r}_i(k) \right) \right) \tag{28}$$

where $\vec{c}(k) \;=\; \dfrac{\left(\vec{r}_0(k) - \vec{r}_1(k) \right)}{2}$,

$\vec{r}_i$ are the vectors of centroids of the index ($i = 0$) or the thumb ($i = 1$) of the hand.

If the index and thumb are used to turn a bolt into a threaded hole (as shown in figure 12), the SCVM is a repetitive function with the repetitive value R. Thus Screwing phase can be represented as:

$$\begin{aligned} Screw_w = \; & \{\left(\left(T_H(k) \right),\, T_{X^v}(k) \right) \Big| \\ & (R \neq 0) \;\wedge\; \left(C_{H^u X^v}(k) \neq \Phi \right) \wedge Grasp\,(k),\, k = k_{sm}^{w}, k_{sm}^{w}+1, \ldots, k_{em}^{w} \} \end{aligned} \tag{29}$$

(5) Drop

When the object falls freely in the k_d frame, it has a speed larger than M,

$$\begin{aligned} Drop_w \;=\; & \{\left(T_H\left(k_d^w \right),\, T_{X^v}\left(k_d^w \right) \right) \Big| \\ & \left(\left| \dot{T}_{X^v}(k_d) - \dot{T}_{X^v}(k_d - 1) \right| > M \right) \;\wedge\; \left(C_{HX^v}(k_d) \;=\Phi \right),\; k_{sm}^{w} \le k_d \le k_e w \} \end{aligned} \tag{30}$$

6.4 Robot hand action evaluation

After being instructed an assembly task, the system automatically generate a task sequence description table (TSDT). Base on the TSDT, the robot command sequence can be easily created by the module of the task translation for the robot to replicate the task the instructor has done. Interesting readers can refer to Paul and Ikeuchi [23] for the robot with grippers, Kang and Ikeuchi [12] for the robot with dexterous hand.

Human can use their eyes to learn how to perform an assembly task as well as to provide feedback signal for correcting its action during working. Similarly, we expect the robot vision system has the same features:

. Providing feedback in real-time for correcting the action
. Evaluating its activities successful or failed
. Reporting what going on in the robot world

We use the four camera multibaseline stereo system in the robot world. This system can not be working in real time because of expensive computation. Based on this system, our approach can only be used for the last two functions. In this section we just focus on the robot action evaluation.

Suppose what the instructor did is absolutely correct. The evaluation robot action is simply to compare them with the instructor's. The task sequence can be represented in a hierarchical level as show in Figure 16. In the symbolic level, the task is the top layer containing a series of subtasks. Each subtask involves two description sets, action set and object relation set. The low level is the models and the model-based data stream calculated from the depth image sequence. From the low level, the primitives forming actions and the object relation are derived. The primitives are basic and compact representation about original data in symbolic and data format. For example, we use line segments or arc segments to represent the moving path in each phase. In the contact (symbolic format) frame, we give the contact points (data format) in both the hand and object as well as the contact area (data format).

The robot action can be evaluated in any layer of the symbolic level or the data level. Obviously, the more lower layer the robot will replicate at, the more difficult the robot does successfully. In our system, evaluation proceeds from the top level to the low layer while the interpretation is starting by the low level. We must point out that successfully replicating task at a lower layer also indicates successfully doing it at all higher layer, not be sure that it is not true in the opposition. It

depends on the application. Because there is such a task which does not need the accurate pose, for instance, contacting the object surface, dropping an object, both without position requirements.

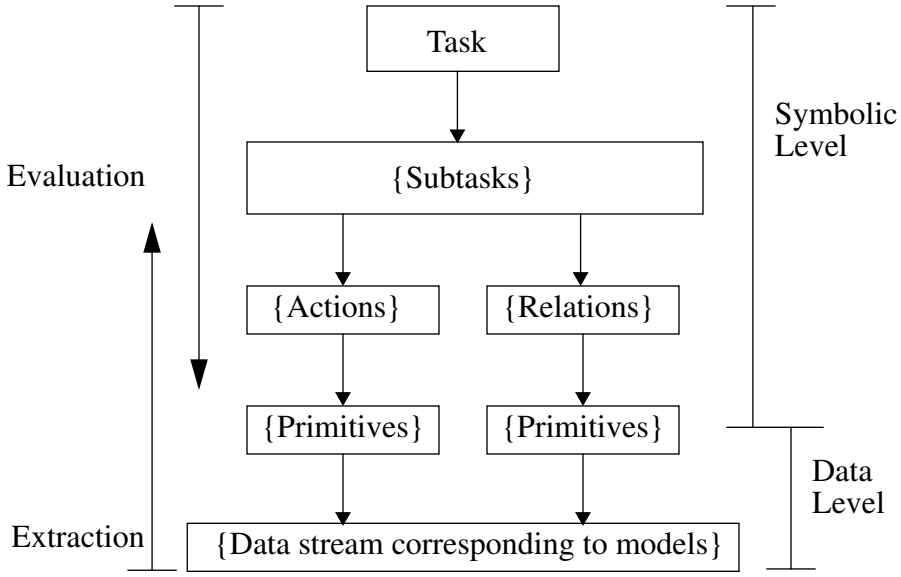


Figure 16 The diagram of task layer

. In the action layer, there are seven possible failed modes shown as follows:

$$Failure1 = \overline{Start}$$

$$Failure2 = \overline{Approach}$$

$$Failure3 = \overline{Contact}$$

$$Failure4 = \overline{Pre-manipulate}$$

$$Failure5 = \overline{Manipulate}$$

$$Failure6 = \overline{Depart}$$

$$Failure7 = \overline{End}$$

7 Implementation

In this section, we will overview the implementation of our approach using some typical assembly examples involving grasping, picking-placing, pushing and turning actions. There are several error sources existing in the different parts of the system, such as calibration error, depth recovery error, model error, tracking error and other computation approximation. In addition, we use a tessellation value of 2 mm for our models. When we use the formulas described above, we have to determine thresholds to mask such errors. We select 6 *mm for* the contact threshold of relative distance (Eq. (11)) between the hand and an object or among objects, 20 *mm* as departing threshold of the distance of the hand from the manipulated object. If the relative distance between two objects is less than 6mm, they contact each other. Otherwise they have no contact relationship. For robustness, we also take the average of the following three relative distances from the contact frame or the departing frame to verify the contact or departure. After contact is established, if the object moves along with hand with a speed larger than 10mm/frame, the manipulating phase begins. If the object and the hand are moving in the different velocities under contacting, it is likely a polishing or wiping action which we did not discuss here.

We already mentioned that we only use two fingers, index and thumb, to manipulate objects. So if only one patch of the object being contacted, it is the touching action. If there are two patches of the object being contacted, and if the weights A^0 *and* A^1 from Eq. (23) or (24) are not equal to zero, this is a grasping pre-manipulation. Otherwise, it is a pushing pre-manipulation. Based on Eq. (25), (26), and (27), the push-pull action or pick-place action can be recognized. For screwing action, we look for repetitive motions lasting from 5 frames to 50 frames. Once all interpretation is completed the system forms the task sequence description table (TSDT) for task translation into a robot control command sequence. The TSDT includes the frame number, task segmentation, subtasks, actions, dominant parameters such as contact position and grasping states which are extremely important for manipulation plan. At last, the system forms the task sequence description table (TSDT) for task translation into the robot control command sequence. The TSDT includes the frame number, task segmentation, subtasks, actions, objects' relations, and with corresponding parameters.

7.1 Human demonstration

7.1.1 Example 1

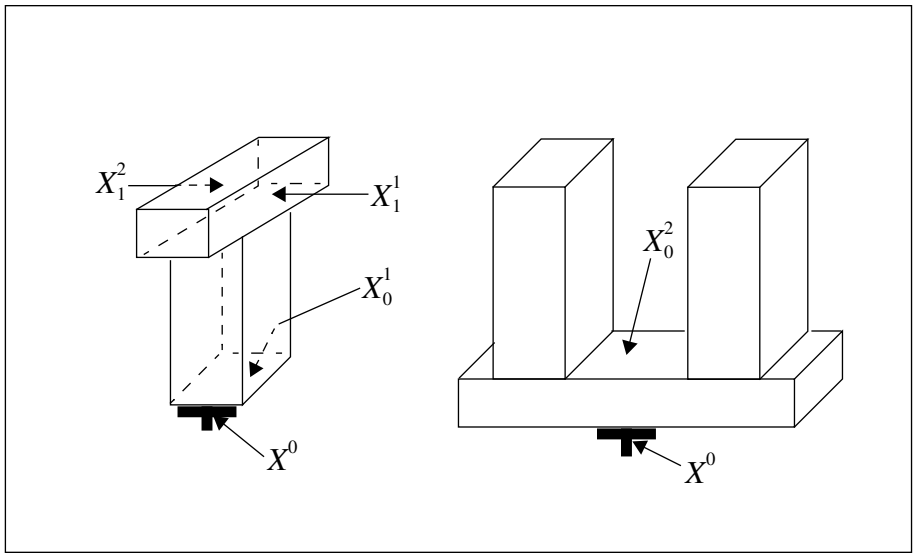


Figure 17 Example 1: Peg-hole assembly task

As an example, peg-hole assembly task, as shown in Figure 17, is widely used in the assembly researching work. Basically, the manipulation is that the human hand or robot hand approaches and grasps the peg firstly, and then moves it to the place much close to the hole and insert it into the hole. This task only involves one subtask. That is, only one object is manipulated only one time. The segmentation and interpretation results are showed in Figure 18 and Table 1 respectively. At the first five frame, the configuration workspace is not changed and hand is still in the stationary. From *6th* frame to *13th* frame, the hand is approaching the peg while the peg is still in the stationary. At the *14th* frame, the distance between the hand and the peg is equal to 5.3 mm, and the average distance is 2.1 mm, less than 6 mm, so the contact frame k_c = 14. Under keeping contacting, it is first time at the *24th* frame for the object speed to be larger than 10 mm/frame moving together with the hand, so starting manipulating frame is k_s = 24. At the departing frame k_d = 67, the R-distance is equal to 65.1 mm, and the average distance is 96.2 mm. At the starting manipulating frame, there are two fingers contacting two patches of the peg, the normal of contacting patches are (0, -1, 0), (0, 1, 0) with respect to its own coordinate system, see Figures 17 and 19, thus $A^0 = -A^1 \neq 0$.

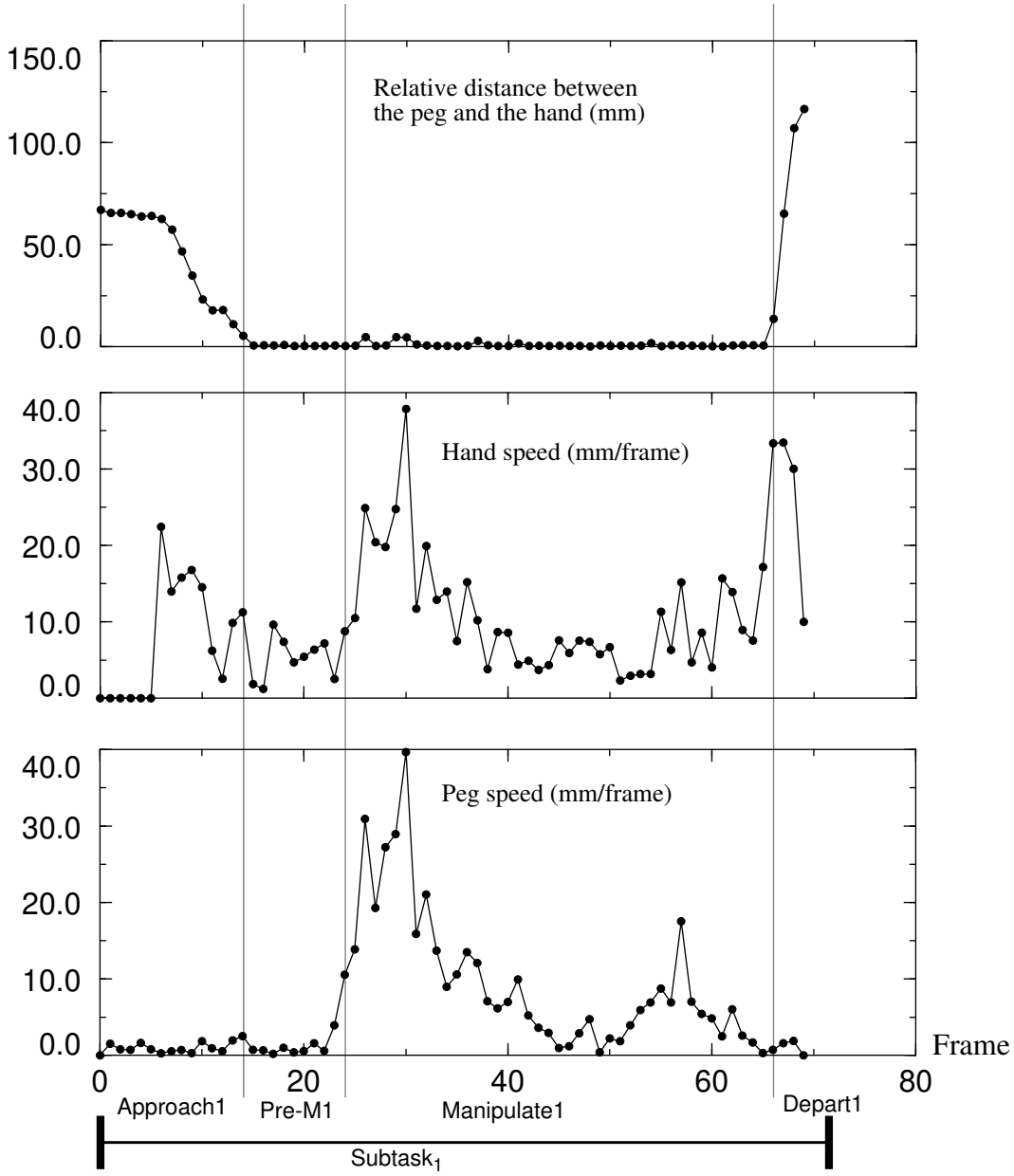
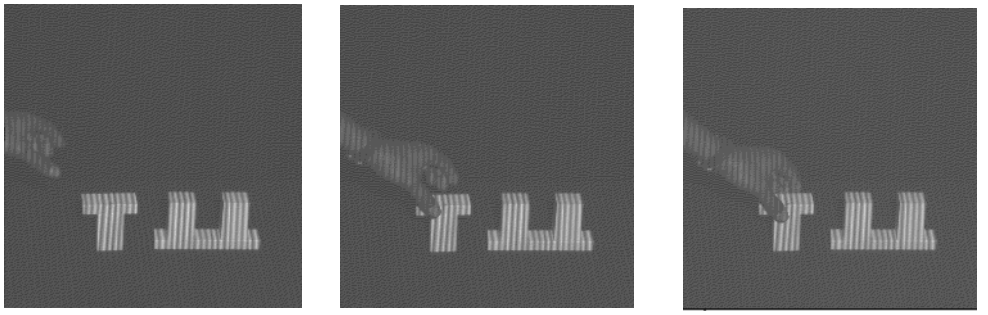
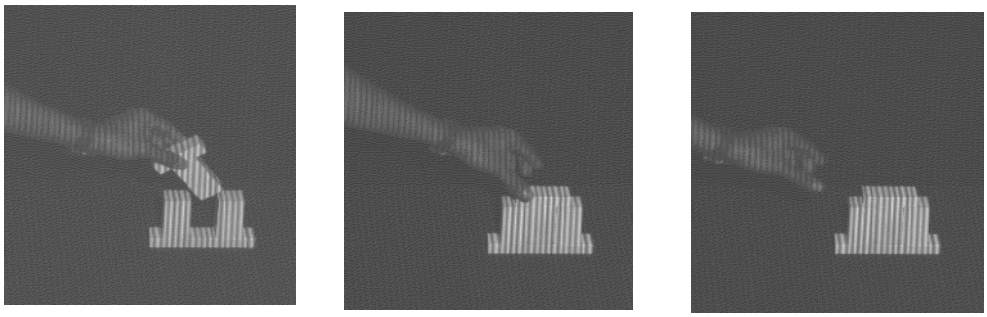


Figure 18 The segmentation of inserting assembly task



Start: f=0 f=14:Contact f=24:Pick-up



f=44:Place-down f=67:Depart f= 69:End

Figure 19 Image sequence of the example 1

Table 1: The TSDT of example 1

Subtask (w) Object (v)	Frame	Action	Parameter
	0	Start	
w = 1 v = 1	1-13 14 15-21 22 23-66 67-68	Approach grasp Pick-Up Depart	 Contact ( ) Grasp ( )
	69	-End	

7.1.2 Example 2

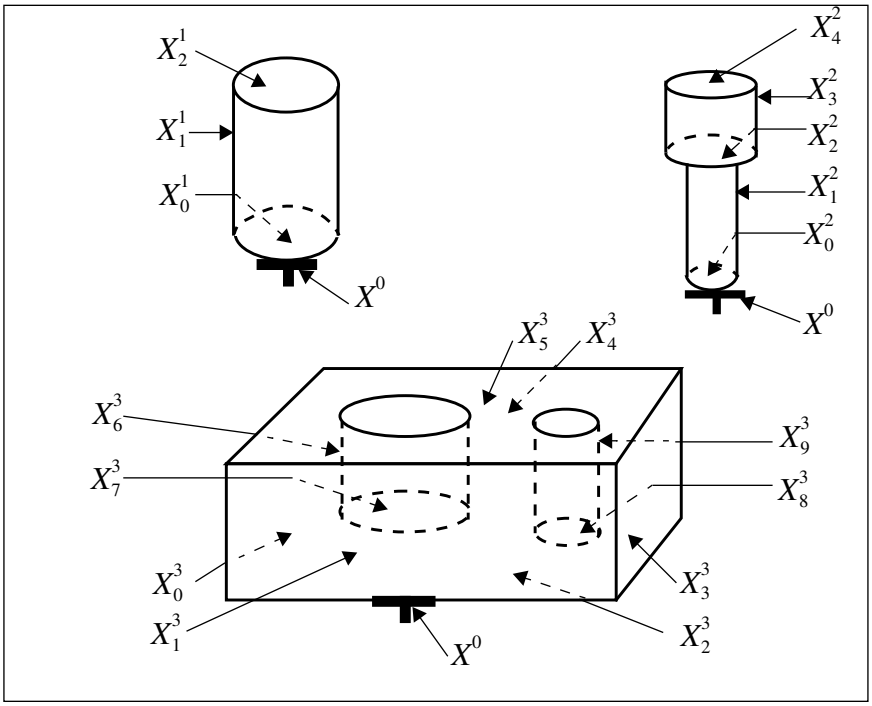


Figure 20 Example 2: Block-cylinder-bolt assembly task

The comprehensive assembly task model showed in Figure 20 involves four subtasks, picking up the block from the source place and placing it to the destination, picking up the cylinder from the source place then inserting it into the bigger hole of the block, picking up the bolt from the source place and inserting it into the threaded hole of the block, and turning it until the plane patches X_2^1 and X_4^2 are on the same plane. Here there are two manipulation phases for the bolt, transmitting phase and turning phase. Figure 21 shows the segmentation results of manipulating the block and cylinder. The first graphic in this figure is the relative distance between hand and the block, while the second is the one between the hand and the cylinder. From those two graphics, the subtask1 and subtask2 can be recognized according the definition in section 6. Where subtask1 is ranging from frame 0 to frame 61 and manipulated object is the block. The subtask 2 is ranging from frame 62 and frame 118 and manipulated object is the cylinder. In each subtask, the contact frame, starting manipulation frame and departing frame can be determined by using the speed curves of the cylinder, the block and hand speed.

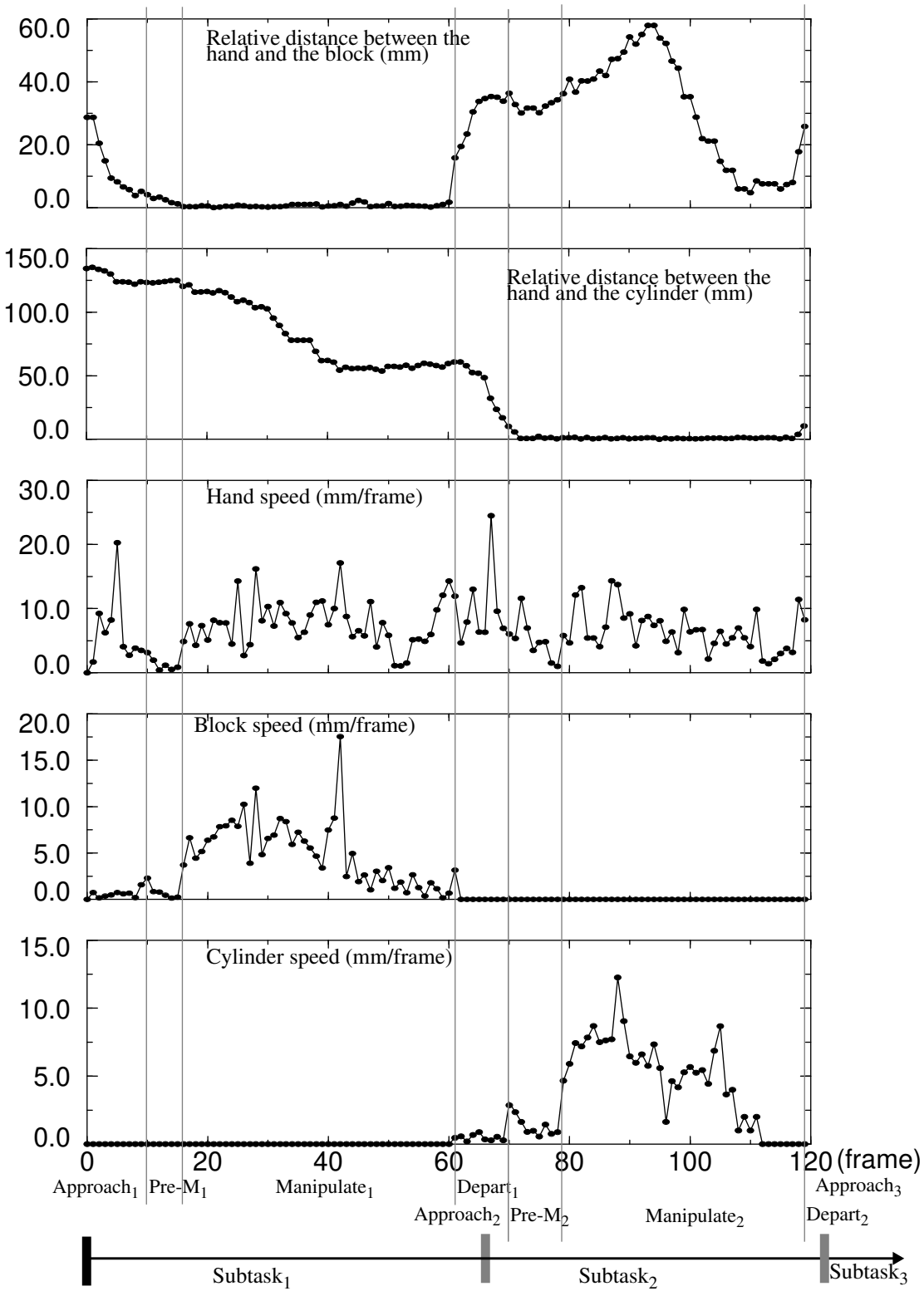


Figure 21 The segmentation of manipulating the block and cylinder

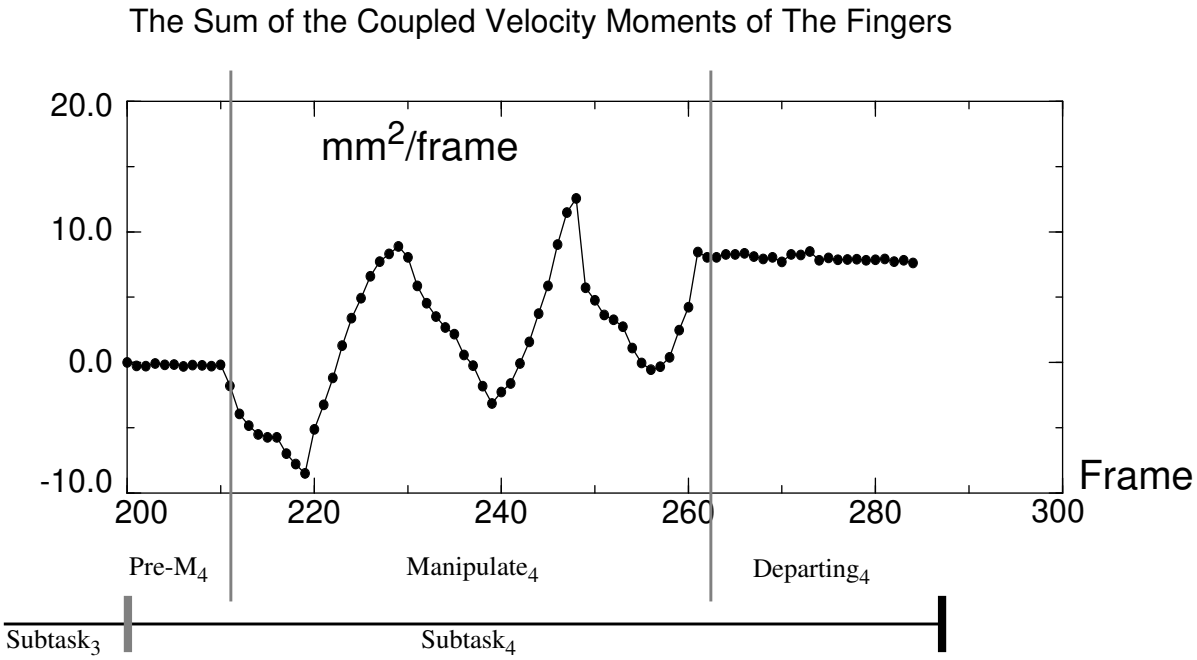


Figure 22 Human hand turning bolt

From Figure 21, the starting frame, starting manipulating frame, and departing frame of the block are frame 10, 16 and 61, while the cylinder are frame 70, 79 and 118. At the starting manipulation frames, the pre-manipulation actions are computed, involving the types of the pre-manipulation and contacting coordinates (see Table 2) which are very useful for robot to do grasping plan. In this assembly example, the subtask 3 is picking up the bolt and inserting it into the threaded hole. The following is the subtask 4, turning bolt into the threaded hole. Figure 22. shows the curves of the sum of the coupled velocity moments of the fingers which tells that this is a repetitive phase.

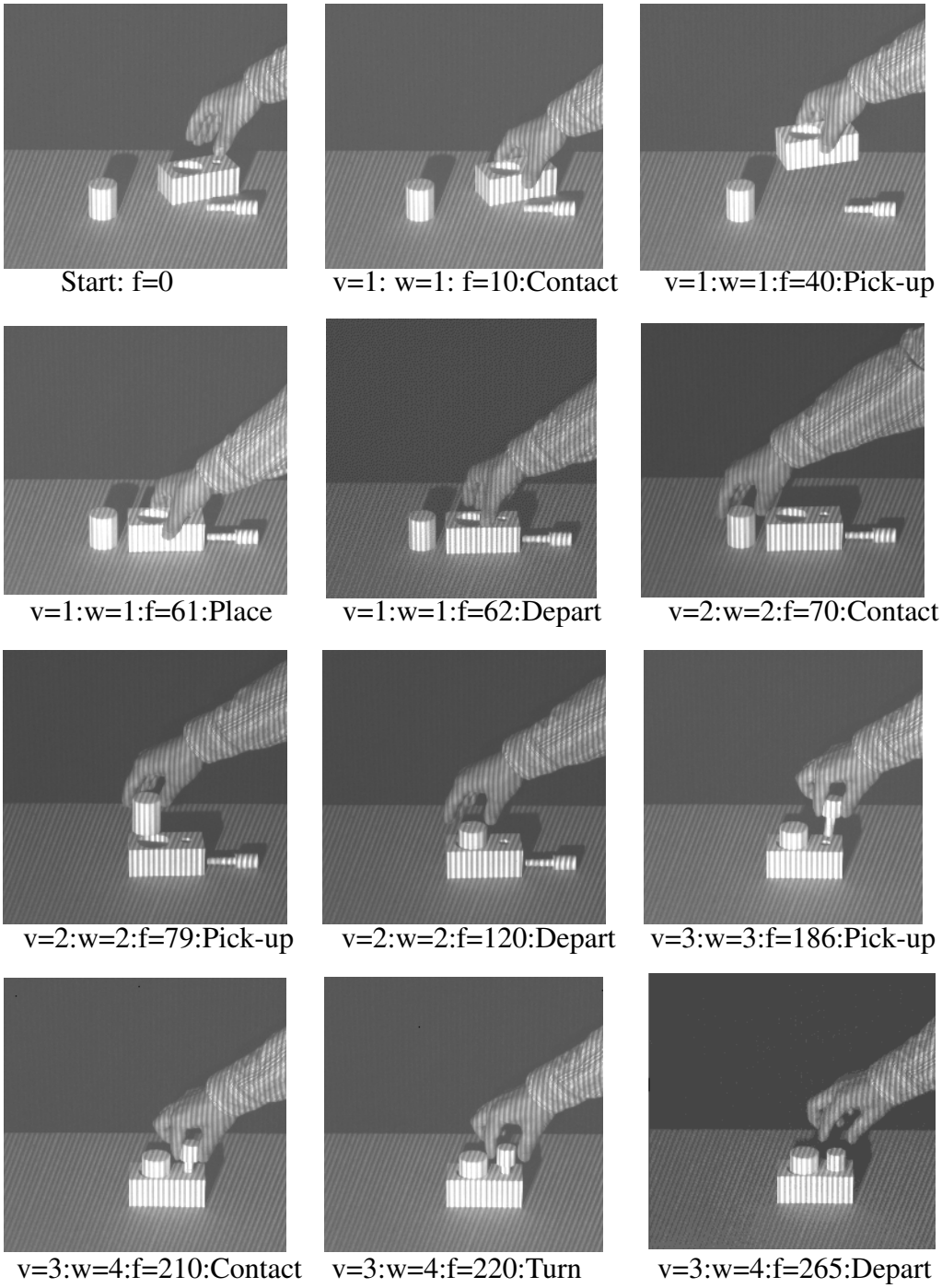


Figure 23 Image sequence of the example 2

Table 2: The TSDT of the Example 2

Subtask (w) Object (v)	Frame	Action	Parameter
	0	Start	
w = 1 v = 1	1-9 10 11-15 16 17-60 61-	Approach grasp Pick-Up Depart-	 Contact ( ) Grasp ( )
w = 2 v = 2	-69 70 71-78 79 80-119 120-	Approach Grasp Pick-Up Depart-	 Contact ( ) Grasp ( )
w = 3 v = 3	-121 196-	-Approach ..... Depart-	
w = 4 v = 3	-200 201 202-210 211 212-265 266-	-Approach Grasp Turning Depart-	 Contact ( ) Grasp ( )
	-280	-End	

7.1.3 Example 3

Figure 24 shows the models of four objects in this example, where the slot is made from a plate X^2 and two small sticks X^3 and X^4. Because the width of the slot is 48 mm that is only a little bit bigger than 44 mm, the width of the stick X^2, it is not easy to put the stick into the slot just from the top unless dropping it which is probably prohibited in many assembly task. So we take a push action to perform this task, as shown in Figure 26. The task involves five subtasks. First, position the plate X^1. Second, put the two small sticks X^3 and X^4 on the two sides of the plate respectively. Third, pick up the bigger stick X^2 and put half of it down into the slot, then depart. And last, push the end of the stick forward until it fits the slot. Here the bigger stick is manipulated twice corresponding to the subtask 4 and subtask 5. Figure 25 shows the segmentation of the subtasks. From the relative distance between the hand and the stick comes two contact areas, from frame 108 to frame 261, and from frame 274 to frame 289, which are corresponding to the subtask 4 and the subtask 5.

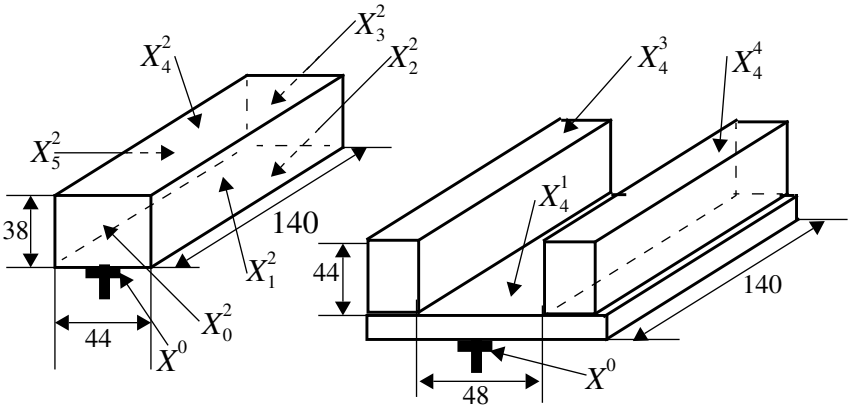


Figure 24 Example 3: Stick-slot assembly task

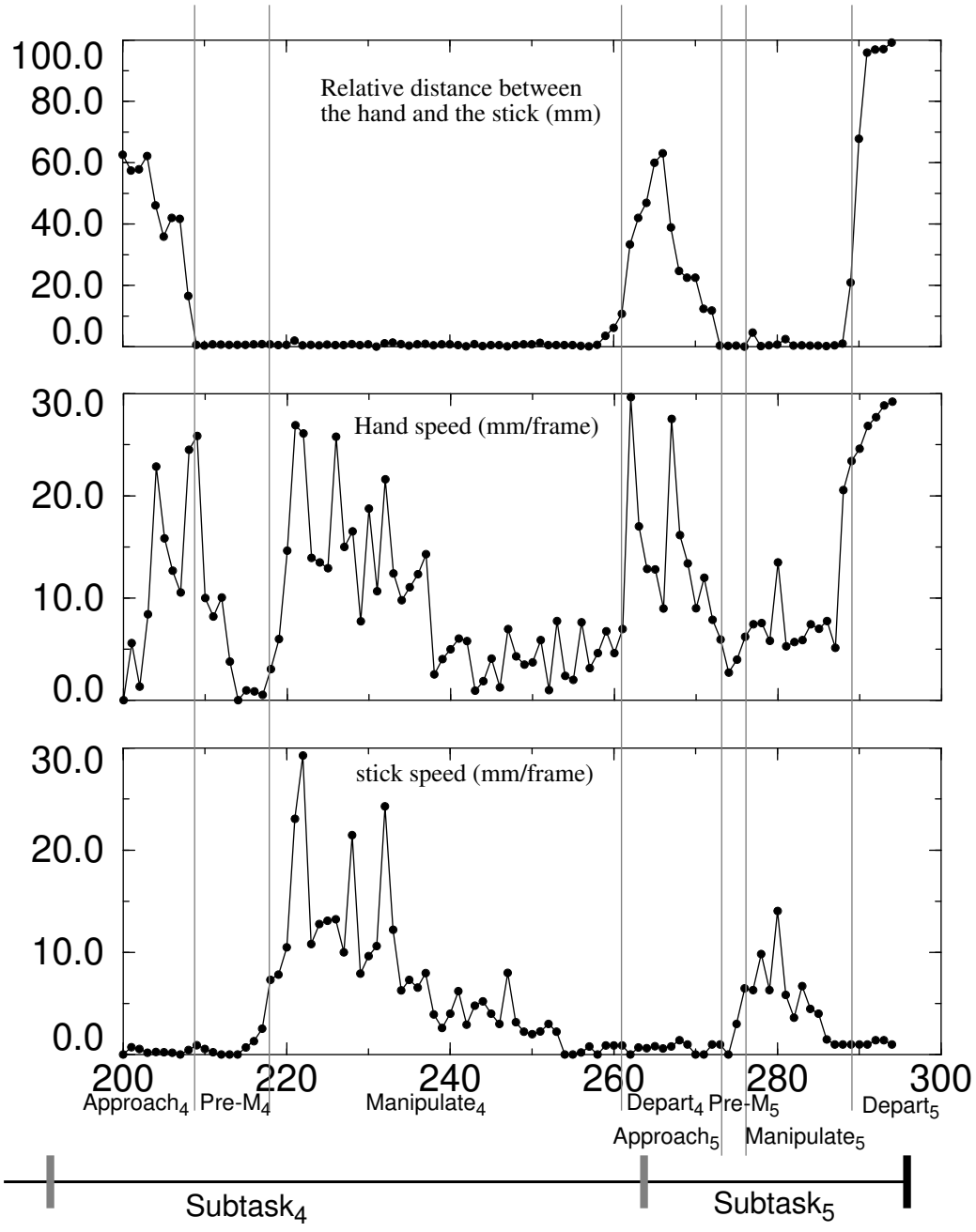


Figure 25 The segmentation of manipulating the stick

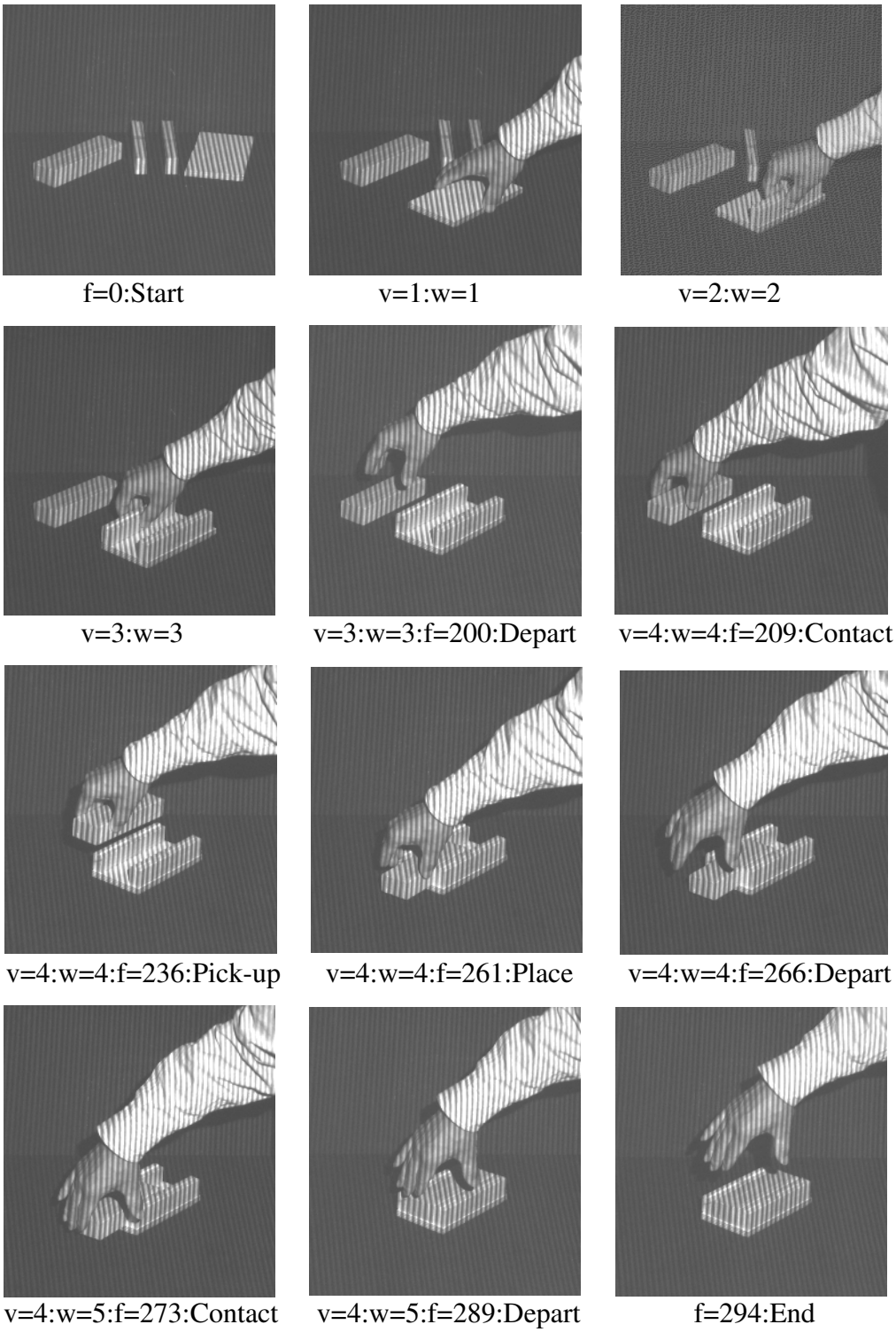


Figure 26 Image sequence of the assembly example 3

Table 3: The TSDT of the example 3

Subtask (w) Object (v)	Frame	Action	Parameter
	0	Start	
w = 1 v = 1	1- 53-	-Approach ..... Depart-	
w = 2 v = 2	-85 126-	-Approach ..... Depart-	
w = 3 v = 3	-144 196-	-Approach ..... Depart-	
w = 4 v = 4	-209 210 211-217 218 219-261 262-	Approach grasp Pick-Up Depart-	Contact ( ) Grasp ( )
w = 5 v = 4	-274 275 276-285 286 287-288 289-	Approach Grasp Pick-Up Depart-	Contact ( ) Grasp ( )
	-280	-End	

7.2 Robot Replication

We have used the same approach on the robot world for reporting what is going on in the robot world and evaluating the robot hand action. We have performed the experiments with the PUMA 560 arm with the Utah/MIT dextrous hand to pick up the block and place it to the destination. The analysis process is the same as in the human world. The experimental results of two are shown in Figure 27, where manipulation is successful, and Figure 29 in failure where the block is dropped while manipulating.

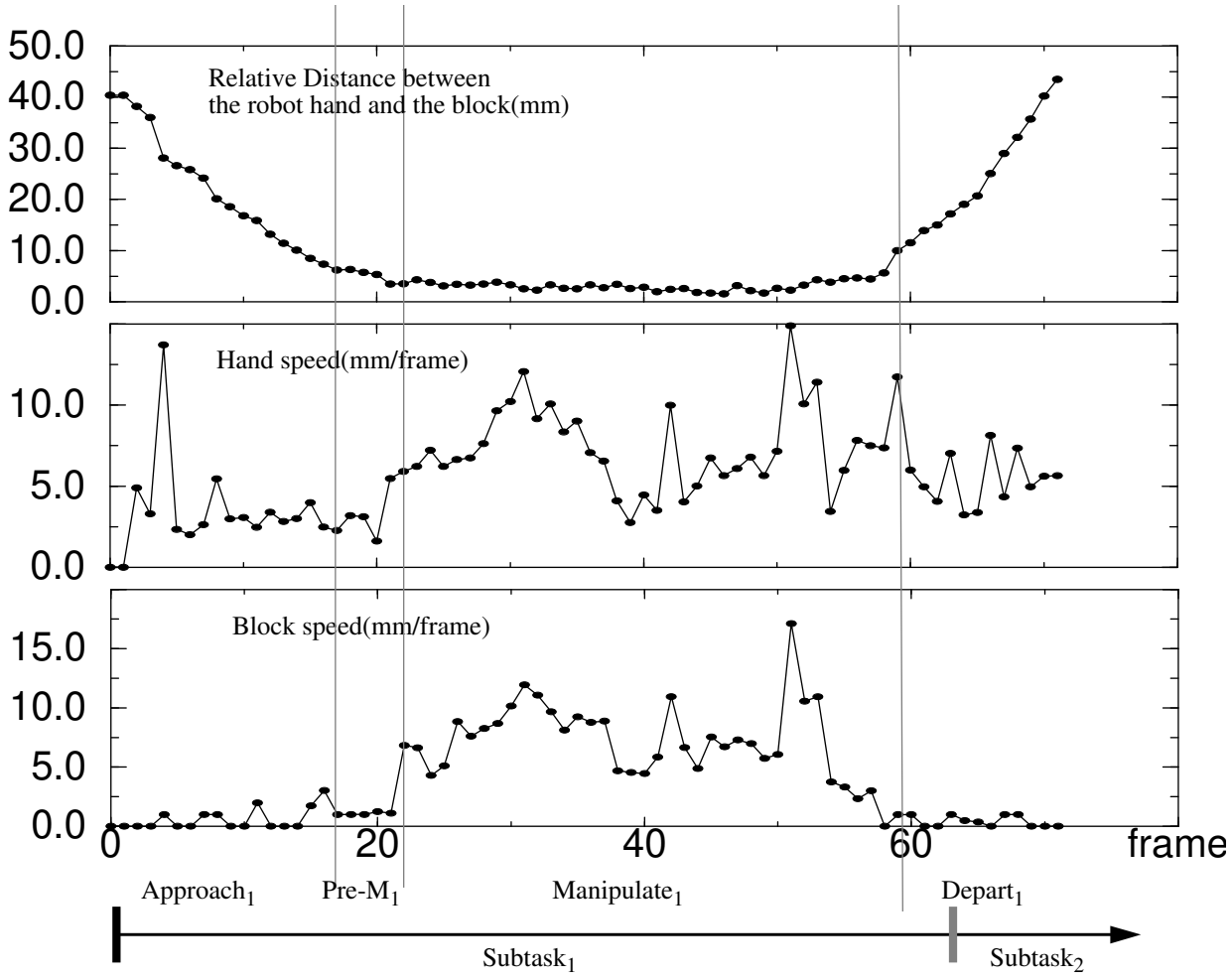


Figure 27 The segmentation of robot manipulating the block in success

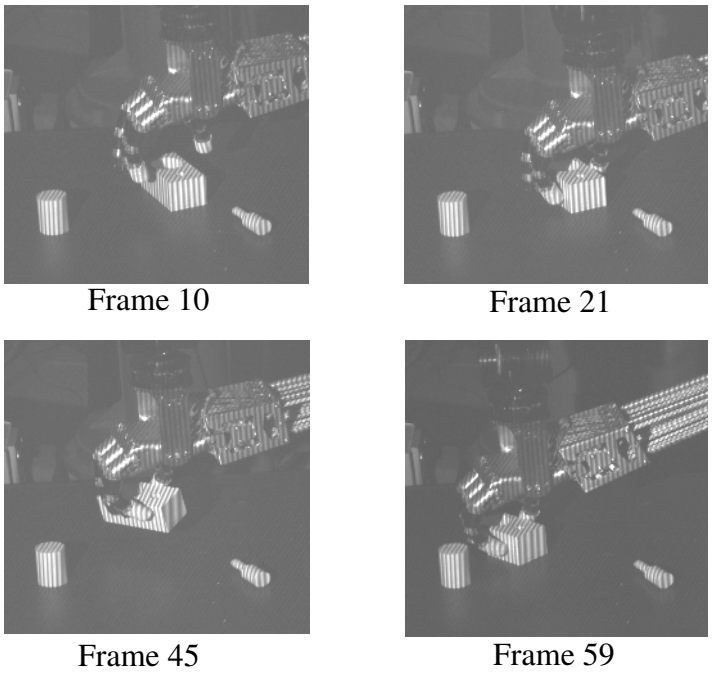


Figure 28 Image sequence of the robot grasping the block in success

Table 4: The TSDT of grasping the block in success

Subtask (w) Object (v)	Frame	Action	Parameter
	0	Start	
w = 1 v = 1	1-13 14 15-21 22 23-66 67-68	Approach grasp Pick-Up Depart	 Contact ( ) Grasp ( )
	69	-End	

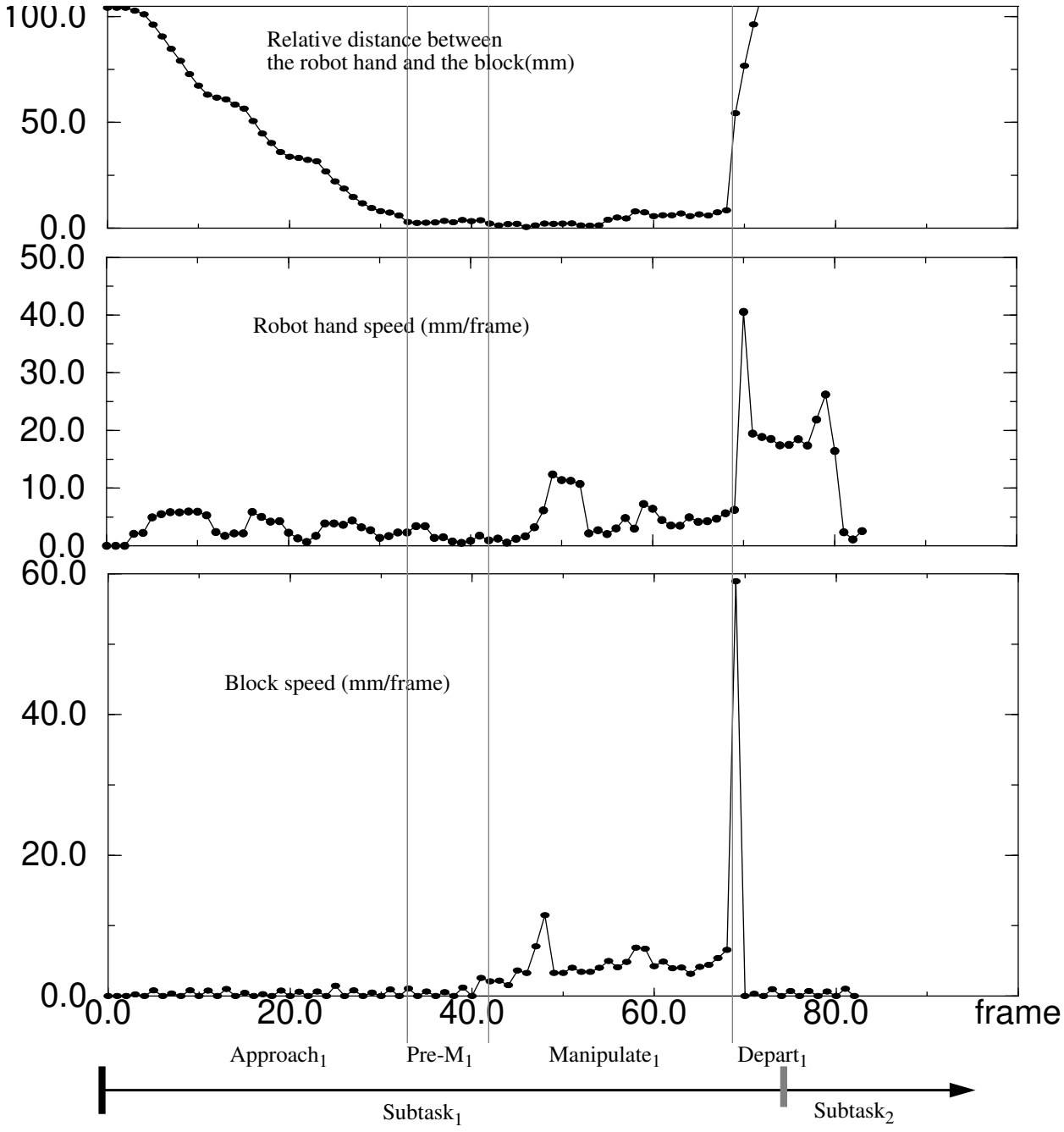
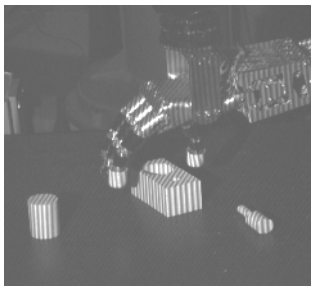
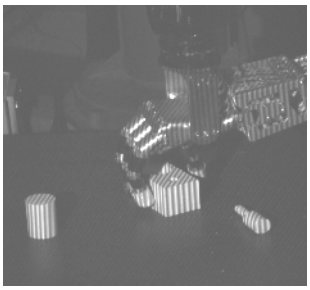


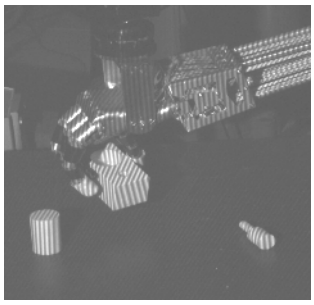
Figure 29 The segmentation of grasping block in failure



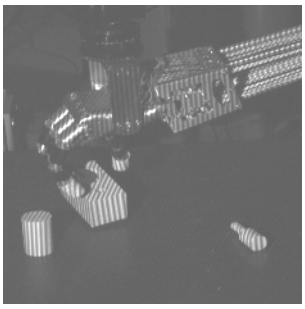
Frame 25



Frame 33



Frame 69



Frame 71

Figure 30 Image sequence of the robot moving the block in failure

Table 5: The TSDT of moving the block in failure

Subtask (w) Object (v)	Frame	Action	Parameter
	0	Start	
w = 1 v = 1	1-13 14 15-21 22 23-66 67 68-68	Approach grasp Pick-Up Depart	 Contact ( ) Grasp ( ) drop ( )
	69	-End	

8 Conclusions

We have discussed and analyzed the framework of automatic robot programming based on hand action perception via stereo systems. We have successfully implemented this approach with many examples of assembly in the human world for robot instruction and robot world for reporting what is going on in the workspace. We use a computational expensive vision algorithm for tracking human hand, so it is difficult to realize the real-time requirements of the robot world. We have two future works planned. The first is to establish simple vision system in the robot world for feedback of the robot hand actions. The second is to develop learning and skill transferring frameworks based on the hand action perception.

9 Acknowledgments

Many thanks to George Paul, Larry Zitnick, Yoichi Sato, and Hiroshi Kimura for helping us and Harry Shum, Kotaro Ohba and Santiago Conant for meaningful discussions.

References

[1] H.Asada and Y.Asari, "The direct teaching of tool manipulation skill via the impedance identification of human motion," Proc. IEEE Int. Conf. on Robotics and Automation, 1988, pp.1269-1274.

[2] H.Asada and S.Hirai, "Towards a symbolic-level force feedback: Recognition of assembly process states", Proc. Int. Symp. on Robotics Research, 1992, pp341-346.

[3] N.Delson and H.West, "Robot programming by human demonstration: The use of human variation in identifying obstacle free trajectories," Proc.IEEE Int. Conf. on Robotics and Automation, 1994, pp.564-571.

[4] O.D.Faugeras and M.Hebert, "The representation, recognition, and locating of 3-D objects," The International Journal of Robotics research, Vol.5, No.3, 1986, pp27-52.

[5] W.A.Gruver et al, "Evaluation of commercially available robot programming languages," Proc.13th Int. Conf. Symp. on Industrial Robots, 1983, pp.12-58.

[6] T.Hamada, K.Kamejima and I.Takeuchi, "Image based operation: A human-robot interaction architecture for intelligent manufacturing," Proc.15th Annual Conf. of IEEE Industrial Electronics Society, 1989, pp.556-561.

[7] H.Hashimoto and M.Buss, "Skill acquisition for the Intelligent assisting system using virtual reality simulator," Proc. 2nd Int. Conf. on Artificial Reality and Tele-existence, Tokyo, Japan, 1992,pp.37-46

[8] T.Iberall, J.Jackson, L.Labbe, and R.Zampano, "Knowledge-based prehension: capturing human dexterity," Proc. IEEE Int. Conf. of Robotics and Automation, 1988, pp.82-87.

[9] K.Ikeuchi and J.Robot,"Modelling sensor detectability with the VANTAGE geometric/ sensor modeler," IEEE Trans. on Robotics and Automation, Vol.7, No.6, 1991, pp777-784.

[10] K.Ikeuchi and T.Suehiro,"towards an assembly plan from observation, Part I: Task recognition with polyhedral objects," IEEE Trans. on Robotics and Automation, Vol.10, No.3, 1994, pp.368-385.

[11] M. Inaba and H.Inoue, "Visual-based robot programming", Proc. 5th Int. Symp. on Robotics Research, 1989, pp.129-136.

[12] S.B.Kang, "Robot instruction by human demonstration," Ph.D. Thesis, Robotics Institute, Carnegie Mellon University, 1994.

[13] S.B.Kang and K.Ikeuchi, "Toward automatic robot instruction from perception -Recognizing a grasp from observation," IEEE Trans. on Robotics Robots and Automation, Vol.9, No.4 1993,pp.432-443.

[14] S.B.Kang, J.Webb, C.L.Zitnick, and T.Kanade, "An active mutibaseline stereo system with real-time image acquisition," Tech. Rep. CUM-CS-94-167, Carnegie Mellon University.

[15] J.J.Kuck and T.S.Huang, "Vision based hand modeling and tracking for virtual teleconferencing and telecollaboration," Proc. Int. Conf. on Computer Vision, pp. 666-671, 1995, Cambridge, U.S.A.

[16] J. Lee and T.L.Kunii, "Model-based analysis of hand posture", IEEE Computer Graphics and Applications, 1995.9, pp. 77-86

[17] S.Liu and H.Asada, "Teaching and learning of deburring robots using neural networks", Proc. IEEE Int. Conf. on Robotics and Automation, Vol.3, 1993, pp.339-345.

[18] Y. Kuniyoshi, M.Inaba and H. Inoue, "Learning by watching: extracting reusable task knowledge from visual observation of human performance, IEEE Transaction on Robotics and Automation, Vol 10, No. 6, 12, 1994. pp.799-822.

[19] T.Lozano-Perez, "Automatic planning of manipulator transfer movements," IEEE Trans. on System, Man and Cybernetics, Vol.SMC-11,No.10,1981,pp.681-689.

[20] M.Okutomi and T.kanade, "A multiple-baseline stereo," IEEE Trans. on Pattern and Machine Intelligence, Vol.15, No.4, 1993, pp.353-363.

[21] P.J.Narayanan, P.rander, and T.Kanade, "Synchronous capture of image sequences from multiple cameras, CMU-RI-TR-95-25, Robotics Institute, Carnegie Mellon University, 1995.

[22] H.Ogata and T.Takahashi, "Robotic assembly operation teaching in a virtual environment," IEEE Transaction on Robotics and Automation, Vol.10, No3, 1994. pp.391-399.

[23] G.V.Paul and K. Ikeuchi, "Modelling Planar Assembly Paths from Observation. CMU-RI-TR-96-13, Robotics Institute, Carnegie Mellon University.

[24] J.M.Rehg and T.Kanade, "Model-based tracking of self-occluding articulated objects," Proc. of 5th Int. Conf. on Computer Vision, 1995. pp 612-617.

[25] H.Rrijpkema and M.Girard, "Computer animation of Knowledge Human Grasping", Computer Graphics, Vol.25, No.4, 1991.

[26] B.Shepherd, "Applying visual programming to robotics," Proc. IEEE Int. Conf. on Robotics and Automation, Vol.2, 1993,pages707-712.

[27] J.M.Siskind, "Naive physics, Event perception, Lexical semantics, and Language acquisition. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1992.

[28] D.B.Stewart and P.K.Khosla, Chimera 3, The real-time programming operating system for reconfigurable sensor-based control system, Carnegie Mellon University, 1993.

[29] T.Suehiro and K.Ikeuchi, "Towards an assembly plan from observation: Fine localization based on face contact constraints," Technical Report, CMU-CS-91-168, Carnegie Mellon University, 1991.

[30] R.Thibadeau, “Artificial perception of actions,” Cognitive Science 10(2), pp. 117- 149, 1986.

[31] R.Y.Tsai, “A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses”, IEEE Journal on Robotics and Automation, Vol-R.3 No4, 1987, pp.323-344.

[32] C.P.Tung and A.C.Kak, “Integrating sensing, task planning, and execution for robotic assembly”, IEEE Trans. on Robotics and Automation, Vol.12, No.2, 1996, pp.187-201.

[33] C.P.Tung and A.C.Kak, “Automatic learning of assembly tasks using a dataglove system, Proc. of Int. Conf. on Intelligent Robots and Systems, pp. 1- 8, 1995.

[34] G.Turk and M.levoy, “Zippered polygon meshes from range images,” Computer Graphics proceedings, Annual Conference series (SIGGRAPH’94), pp.311-318.

[35] M.D.Wheeler and K.Ikeuchi, “Sensor modelling probabilistic hypothesis generation, and robust localization for object recognition,” IEEE Trans. on Pattern Analysis and Machine Intelligent, Vol.17. No.3, 1995, pp.252-265.

[36] H.Zak and H.Das, “Towards a training methodology for skilled teleoperation”, IEEE Trans. on System, Man, and Cybernetics, Vol.25, No.2, 1995, pp.313-327