Acquiring hand-action models in task and behavior levels by a learning robot through observing human demonstrations

Koichi Ogawara[1], Jun Takamatsu[1], Soshi Iba[2] Tomikazu Tanuki[3], Hiroshi Kimura[4], and Katsushi Ikeuchi[1]

[1] Institute of Industrial Science, Univ. of Tokyo, Tokyo, 106-8558, JAPAN,
ogawara,j-taka,ki@iis.u-tokyo.ac.jp,
WWW home page: http://www.cvl.iis.u-tokyo.ac.jp/
[2] The Robotics Institute, Carnegie Mellon University, Pittsburgh PA, USA,
iba+@cmu.edu
[3] Research Division, Komatsu Ltd. Kanagawa, 254-8567, JAPAN,
tomikazu_tanuki@komatsu.co.jp
[4] Univ. of Electro-Communications, Tokyo, 182-8585, JAPAN,
hiroshi@kimura.is.uec.ac.jp

Abstract. This paper describes our current research on learning tasks (what to do) and behaviors (how to do it) by a robot through observation of human demonstrations. We focus on human hand actions and represent such hand actions in symbolic behavior and task models. We propose frameworks of such models, derived a method to acquire those models through observation, evaluated them by using a human-form robot, and considered a method to improve such models automatically.
We approach the problems on two levels: task and behavior. At task level acquisition, which obtains what-to-do information, we propose a method for constructing a human task model by attention point (AP) analysis, which can efficiently integrate multiple input from several observation devices distributed in space and time, and then construct an abstract task model. As behavior level acquisition, which obtains how-to-do information, we present a method for segmenting a human assembly behavior into primitive behavior models, referred to as sub-skills; this method consists of analyzing face-contact transitions of manipulated objects in a human demonstration. We also describe the structure of the robot that we developed for learning and performing such hand-actions, and present experimental demonstrations performed by the robot.

1 Introduction

One of the most important issues in robotics is how to program robot behaviors. Several methodologies for programming robots have been proposed. We can classify them into the following three categories: static textual programming, manipulation by a human through a control device, and automatic programming. In static textual programming, a human programs all the behavior of a robot by hand with or without some tool aids (teaching pendant or simulator, etc.) before operating the robot. Typical industrial robots are programmed using the method in this category. Manipulation by a human, including teleoperation, is a method to control a robot directly by a human using some control devices, such as master arms or a head mount display (HMD). This method usually offers some auxiliary low-level semi-autonomic functions such as walking or impedance-controlling functions to stabilize robot motions by compensating time-critical controls while a human sends high-level commands to perform desired behaviors. These two methods require human intervention throughout the entire task. In contrast, automatic programming is intended to reduce human aid and to generate an entire robot program automatically. Given the necessary initial knowledge, robots try to acquire their behavior automatically from observation, simulation or learning.

This paper presents our latest research on automatically acquiring robot behavior, in particular, hand-actions, from observation. The method is based on the automatic programming approach. We divide the acquisition process of human tasks into two levels: task level, e.g, what-to-do and behavior level, e.g., how-to-do it. In Chapter 2, we discuss the necessity of this categorization into task level and behavior level acquisition. In Chapter 3, we present a method for constructing a human task model by attention point (AP) analysis for task level acquisition. In Chapter 4, we present a behavior level acquisition to recognize a human hand behavior by analyzing face-contact relations of objects. To verify the obtained model, we utilize a test-bed robot, described in Chapter 5, which has similar capabilities to that of humans, for learning human tasks (especially human hand-actions) from observation. In Chapter 6, we introduce descriptions of this robot performing demonstrated tasks and assisting a human by cooperative behaviors. Chapter 7 contains our conclusions and remarks on future work. These types of robot dramatically extend the area of robot applications in a human-robot co-existent environment, because they can automatically learn human behavior and increase their behavior repertoire with the help of ordinary people who have no professional programming skills..

2 Acquisition of Human Task

2.1 Task Level Acquisition

Ikeuchi and Suehiro, and Kuniyoshi et al. studied vision based task acquisition [1, 2]. In that research, the acquisition system observed a human performing an assembly task and constructed a high-level task model. Then, using that model, a robot performed the same task. Kimura et al. proposed a task model which could be used to realize cooperation between a human and a robot [3]. In this method, the robot first observes sequential human operations, referred to as events, by vision and analyzes mutual event dependencies (pair of pre-conditions and results) in the task. Based on this analysis, the robot is able to properly assist a human according to the current task event and the knowledge of what is to be done next. The robot changes its assistant behavior while adapting to human behavior. The result is a large number of cooperative schemes from a single sequence of task models that represent a chain of events, thereby demonstrating the effectiveness of task models. However, these models are environment-dependent; therefore, the robot can perform tasks only in the same environment as that of the learning stage, and so such models cannot be applied in different environments.

Our study enables us to overcome this environmental dependency in task models and to enlarge their applicability. Our task models consist of high level symbolic descriptions, e.g. ”Power-grasp” an object which has ”Shape A” and ”Texture B” at ”Location C” by ”Right hand”. This is important, because by describing the top level of the task model in a high-level human-readable way, a human and a robot can have a common understanding about the target task; this understanding, we think, is essential to enable a robot to act in a human co-existent environment. For reducing the environmental dependency in task models, we introduce priorities in descriptions of task models. By reducing the priority of some descriptions, impediments to completion of the task (such as a location term in a different environment or which hand should be utilized) are sidestepped, and thus the models become applicable in different environments from those existing at demonstration time.

We also introduce the concept of attention point for the robot analysis. In previous research, once a robot has analyzed a human task, the robot’s attention never turns back for a closer analysis. However, it is impractical to apply a detailed analysis over the entire human task sequence captured from input devices for obtaining a human task model. It is also true that a rough analysis may turn out to be inadequate for some parts; these parts may require a more detailed analysis. Therefore, we introduce a two-step analysis and propose a novel method of constructing a human task model by attention point (AP) analysis.

2.2 Behaviour Level Acquisition

Behavior level acquisition is defined as obtaining a specific behavior in one task. Each behavior is composed of primitive motions and requires a detailed analysis plus repeated training to master it. Atkeson et al. utilized a parametric model to describe the motion of a pendulum and made a robot learn swinging-up-a-pendulum behavior from a single human demonstration [5]. This approach modeled the motion of the pendulum with specified parameters (pendulum angle, angle velocity, etc.) and tried to change the values of the parameters to reduce the difference between a generated robot motion and the human demonstration. These parametric model-based approaches are effective when the target behavior is simple and the adjustable parameters in the model are known in advance, but such direct imitation approaches have several drawbacks. They will easily fail for acquiring complicated behaviors, because it is difficult to construct a suitable parametric model by hand or automatically. Even if a model is constructed, the system, when confronted with a large number of parameters, may have difficulty in determining which parameters are appropriate for adjustment .

Complicated behaviors, such as human hand-actions, require not only a parametric model analysis, but also a context base analysis. Here, a context base explicitly describes the relationship between hands and objects and its transition along the time series to constrain the entire behavior. We have been developing the Assembly Plan from Observation (APO) [1] system, which provides a robot the ability to observe assembly tasks from observation and subsequently generate a program to perform those same tasks. The APO system employs a context-based analysis to obtain a chain of assembly tasks, such as ”insert-into” and then, ”put-on.” In the current system, as a behavior based approach, we focus on the transition of the face-contact relations within an assembly task, e.g., how face-contacts transit to achieve an ”insert-into” assembly task. We describe face-contact transitions between the manipulated and environmental objects in one assembly task and the development of a system that automatically generates symbolized robot behaviors by analyzing such face-contact transitions [4]. These symbolized behaviors are then mapped to robot motions (sub skills).

3 Task Level Acquisition

In this chapter, we describe the construction method of a human task model based on attention point (AP) analysis (Fig. 1) and the implementation of human task modeling using data gloves and a 9-eye stereo vision system. (An experimental result which demonstrates the applicability of the model is described in Chapter 6.)

3.1 Attention Point (AP)

APs, which require close observation to learn a particular behavior, are set around specific time and position in a sequence of a human task.

I: AP for arbitration between input data separated in space

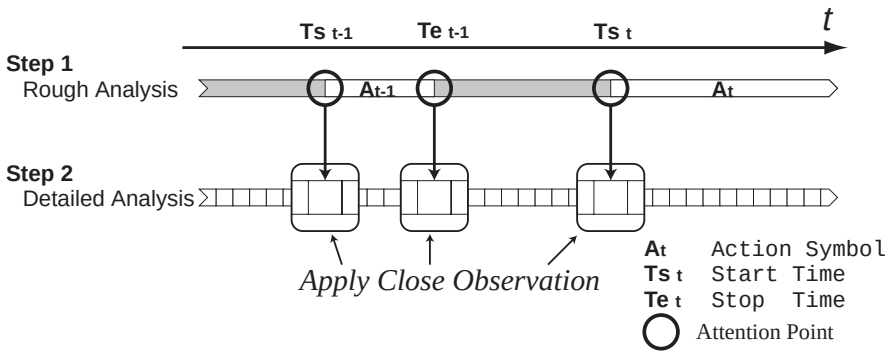


Fig. 1. Two Steps Analysis using Attention Point

When several input sensors are available, it is generally ineffective to process all the data along the entire human task. This study records all the raw data from all the sensors along the entire human task and employs a two-steps analysis of a human task (Fig. 1).

First, by using a minimum set of input data, a robot globally analyzes a human task and segments the entire task into meaningful behaviors. With this analysis, the robot constructs a rough human task model and, for closer analysis, sets APs which point to specified time and position, on the boundaries of each segmented behavior.

Second, by using remaining (or all) input data, the robot applies a detailed analysis around each AP and locally enhances the human task model.

II: AP for arbitration between input data separated in time

When a robot performs a task and fails at some point, the corresponding human behavior must be analyzed carefully. We set APs around such failure points. By observing repeated human tasks and analyzing in detail around the extracted APs, the robot can enhance the human task model. This type of AP analysis requires quantitative evaluations of a number of demonstrations for the same task, and determines quantitative task models. In this way, the robot can perform a longer and deeper analysis of the necessary parts of the human behavior so as to be able to build a task model efficiently.

In this paper, we mainly focus on the first AP (for arbitration between input data separated in space, Fig. 1).

3.2 Recognition of Human Task

We limit the possible tasks to human hand-action on a table and describe the method of AP analysis using data gloves and stereo vision.

We can acquire depth and color images from the stereo vision and hand motion data from the data gloves. The image processing is much more time-consuming as opposed to the processing of the data gloves, so we adopted a two-stepped AP analysis to effectively handle two different sets of input data. Using the data gloves, the robot first constructs a rough human task model and extracts APs. Then, to enhance the task model, the robot analyzes depth and color images only around those Aps.

Rough Human Task Model We classified possible finger actions into three actions: ”Power Grasp”, ”Precision Grasp”[6] and ”Release.” We describe human hand actions as a finite set of ”Action Symbols,” which are combinations of above finger actions and local hand motion. By excluding hand actions composed of independent finger motion, we can segment the entire hand-action into meaningful ”Action Symbols”. As a result, we can construct a rough human task model (Fig. 2) as a sequence of discrete Hand Actions. These Hand Action models also contain additional attributes (”Time Stamp”, ”Hand”, ”Position”), as shown in Table 1.

We assign each “Action Symbol” to a corresponding gesture and segment left-and-right dual hand actions by hidden markov model (HMM) based gesture spotting technique. (This technique is described in Chapter 3.3.)

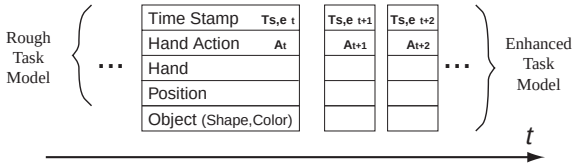


Fig. 2. Human Task Model

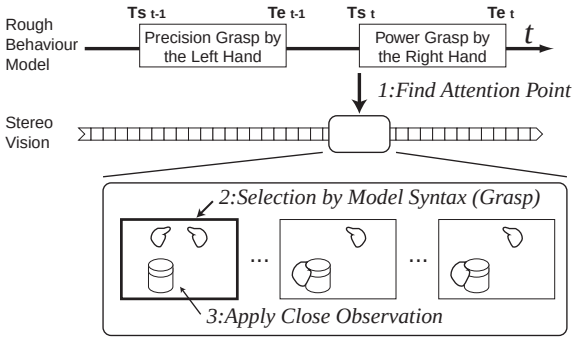


Fig. 3. Example of Applied Model

Table 1. Attributes of Hand Action

Attributes	Priority	Value
Time Stamp	1(low)	Absolute Time (start and stop time)
Action Symbol	3(high)	Power Grasp, Precision Grasp Release, Pour, Hand Over
Hand	2	Right, Left, Both
Position	1	Absolute Position in 3D space
Object Model	3	Type of the Manipulated Object

Detailed Analysis around APs A rough task model contains information only about hand motion and has no knowledge about the manipulated objects. Since the model contains information about time and position of each grasping point as APs, the robot can obtain such knowledge through a detailed analysis invoked by APs. A detailed analysis is applied to the depth and color images recorded at the time of AP for recognizing the manipulated objects.

By setting APs just before the grasping action is performed, we can obtain images in which the target object is not occluded by the grasping hand (Fig. 3). Recognition of the objects is processed by calculating shape histogram and color histogram, and the robot adds this histogram information as a new attribute ”object model” into the corresponding hand action. (This process is described in Chapter 3.4.)

Priority Each attribute has a priority term to specify the degree of its importance. First, the robot plans to perform the task exactly as the model specifies. If that does not work, the robot ignores the attributes of a lower priority for completing the task, and sets up a plan using only the attributes of a higher priority. For example, if, for some reason, the robot determines that it cannot perform the hand action, “Grasp the object A at the place X by the Left Hand”, it omits the attribute “Position” and “Hand” in order, and tries to grasp the object A by any hand available. Thus, by reducing the constraint before giving up the entire task, the robot can avoid discontinuance of the task.

The AP analysis constructs a human task model which is highly abstract and is able to change the degree of abstraction adapting to the environment by the priority term. So the model can be applicable in a different environment. (We show an example of this applicability in Chapter 6.)

3.3 Rough Analysis by Gesture Spotting

To obtain "Action Symbol" for a rough human task model, we aim at spotting human gestures while a human is performing some hand-actions. In this study, we select 6 gestures (5 described in Table 1 + OK-sign for training) as "Action Symbols" and try to symbolize a human task performed with the two data gloves by gesture spotting based on Hidden Markov Models (HMMs). Gestures from each hand are spotted in parallel, while two-handed gestures are spotted by combining results from both hands.

In this chapter, we will go over the general concept of a HMM, gesture spotting, and the description of the gesture spotting system.

HMM HMMs are used to model a signal with variability in parameter space and time. HMMs model doubly stochastic processes that are first-order Markov processes whose internal states are not directly observable and thus, the term "hidden" is used. The observable output signal depends on probability distributions, fixed for each internal state. Since the model can disregard noise through a stochastic framework, it allows us to deal with the highly stochastic underlying structure of the process [16].

Gesture Spotting Gesture spotting refers to the recognition and extraction of a meaningful segment corresponding to gestures from input signals that vary in both time and space. By using a gesture spotter, the user is able to interact with the system without keeping the start and end of gestures in mind. HMM-based pattern spotting is done by placing keywords modeled by HMMs and filler models in parallel in a loop. This way, the keyword can be recognized while rejecting non-keywords through filler models[17].

Recognition using Data Gloves We use right and left data gloves (CyberGlove), and 6-DOF position sensors (Polhemus) as input devices to perform HMM-based gesture spotting. Part of the system is based on the Hidden Markov Model Toolkit (HTK)[18].

As observable features of the HMMs, we are using 48 dimensional features per hand at time t. The feature vector consists of 18-dimension joint angles, $\{r_1 \ldots r_{18}\}_t$, 6-dimension hand velocity, ${}^{t-1}P_t = {}^{t-1}\{x, y, z, \alpha, \beta, \gamma\}_t$ which in fact is a velocity referenced from the previous hand coordinate, and differentials of the above 24 features (Fig. 4).

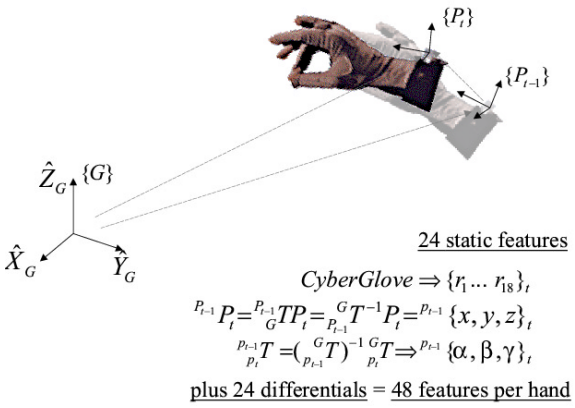


Fig. 4. Definition of Feature Vectors

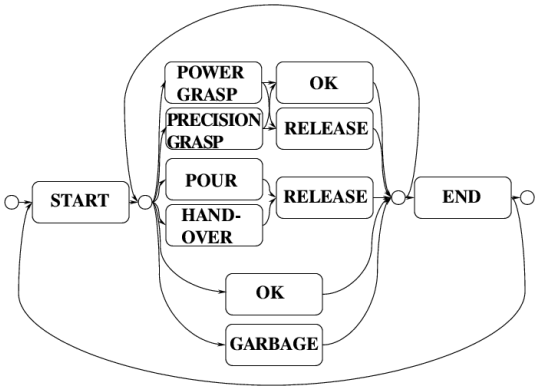


Fig. 5. Gesture Transition Network

We define a gesture as an attachment of primitive HMMs (Table 2). By sharing primitives, each gesture can use a small number of training data with better efficiency. We define 9 primitives: *cls, prc, roll, forw, opn, ok, gb, sil, sp*. *cls, prc, roll, forw, opn, ok* are defined as 5-state left-right HMMs (models with a single way transition from the start to the end). *sil* is a silent state used at a time of training, *sp* is a short pause which tends to be there at the end of the gesture, and *gb* is a garbage collector that is trained on arbitrary non-gesture movements.

Our system can sample the data from the right and left data gloves in 30Hz and spot gestures in parallel without delay. Fig. 5 displays information on the HMM grammar network used in gesture spotting.

Table 2. Gesture definitions

Gesture	Primitives	Action
Power Grasp	cls+sp	Power-grasp from open position
Precision Grasp	prc+sp	Precision-grasp from open position
Pour	cls+roll+sp	Power-grasp, and roll the wrist
Hand-over	prc+forw+sp	Precision-grasp, move forward, and back
Release	opn+sp	Open a grasp hand
OK-sign	ok+sp	Make a circle with thumb and index finger
Garbage	gb	A filler model for spotting
Start,End	sil	Silence at the start and end

3.4 Detailed Analysis by Stereo Vision System

To enhance a rough human task model, we utilize shape histogram and color histogram as an object model. We assume that a human task is demonstrated on a table whose geometric information is known. By subtracting the table surface from the acquired depth image, we can extract depth regions corresponding to each object on the table. Shape histogram is calculated as a list of matching likelihood between an object extracted in the depth image and each prearranged geometric object model. Matching likelihood is calculated by the 3D template matching technique.

Color histogram is calculated as a normalized hue histogram which counts pixels with large saturation value among the area of the object on the color image. These depth and color images are produced at 5 fps (up to 30 fps) by 9eye multi-baseline stereo vision system [8].

In this chapter, we describe 3D Template Matching (3DTM) technique and describe the method of calculating shape and color histogram. Then we show a histogram-based object recognition method.

3D Template Matching (3DTM) 3D Template Matching (3DTM)[12] is the technique used to find the precise position and orientation of the target object in depth data by projecting the corresponding 3D model.

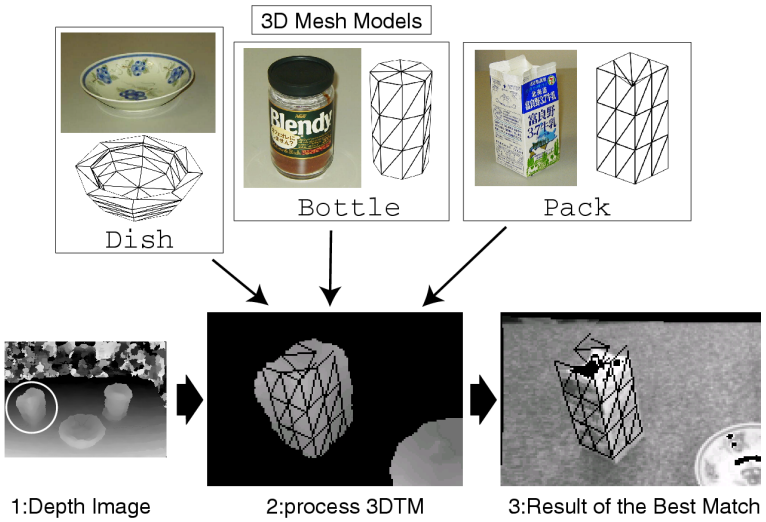


Fig. 6. Recognition of objects with 3DTM

The technique assumes that the 3D geometric model (template) of a target object and the initial position of the target object is known in advance. It projects the 3D model into the 3D space generated from a depth image.

Then it calculates matching likelihood between the 3D model and the 3D data by summing up weighted distance between each vertex in the template model and the closest 3D point. We adopt M-estimator, which is a generalized least squared method, as a weight function.

3DTM iteratively moves the model in 6D parameter space (position and orientation) to decrease the distance until it converges. The settled parameters are the estimated position and orientation.

Shape Histogram The robot builds a shape histogram as follows: (i) It extracts regions corresponding to objects by removing the background and the table surface from the depth data. (ii) It applies 3DTM on the extracted region using a set of known geometric models (Fig. 6) and calculates matching likelihood.

Table 3. Detected Shape Histogram (Result of 3DTM)

Objects (in Depth Data)	Models		
	Pack	Dish	Bottle
Pack Histogram	0.25	1.30	0.55
Dish Histogram	2.08	0.65	1.43
Bottle Histogram	0.92	1.20	0.37

3DTM is sensitive to the initial position of the projected model and produces a better result when the target object is not occluded. From the rough human task model, the robot can select the proper depth image at the AP and get the initial position for 3DTM (from “Position” attribute).

Table 3 shows the result of 3DTM applied to the objects used in our experiment. Each value indicates weighted distance by M-estimator, i.e., matching likelihood, and the underlined value is the best matching result. Each row in the table corresponds to the shape histogram. This result shows that the objects were correctly registered. But if some of the models have similar shapes, it occasionally fails to register the correct model because of the noise-contaminated depth image and the error of the initial position. Using the shape histogram instead of a single resultant value is much more robust in those situations.

Color Histogram When the objects on the table have similar shapes, color information has the great benefit of avoiding miss-recognition. We divide the hue space into equally separated twelve areas and find to which areas each color pixel of the target object belongs excluding pixels of lower saturation. Then, by summing up the number of pixels for each area and normalizing the result, we can obtain a color histogram.

Histogram Based Object Recognition When the robot is to perform the same task after constructing a task model, it searches for objects on the table and, for each object, it calculates mean square distance between the shape and color histogram of the object on the table and those of the object in the model. The smallest value means the best matching objects. In this way, the robot finds the correct object described in the model.

4 Behaviour Level Acquisition

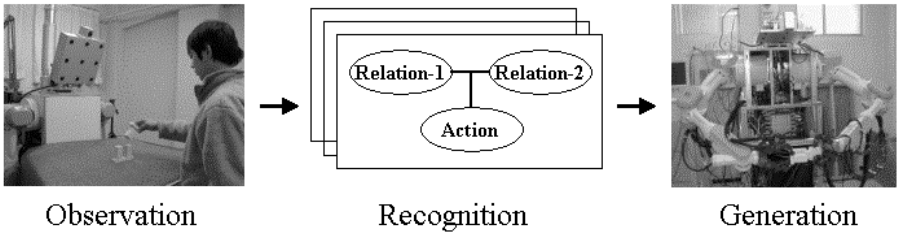


Fig. 7. Outline of APO

In this chapter, we introduce a behavior level acquisition of a human assembly task, which is a part of the Assembly Plan from Observation (APO) [1] system. The APO system consists of following 3 parts: (i) observation of a human task (ii) recognition of that human task (iii) generation of a robot program (Fig. 7) to accomplish that task.

In the observation phase, the trajectory of each object is recorded as sequential depth data streams by the stereo vision system. From the depth data, using 3DTM object recognition method [12], the system extracts the trajectory of each object.

In the latter sections, we present a summary of the recognition phase based on face-contact transition analysis. (For a detailed description, please refer to [4].) We also show an experimental result of peg-insertion task in which the robot performed the same task by generated behaviour programs in Chapter 6.

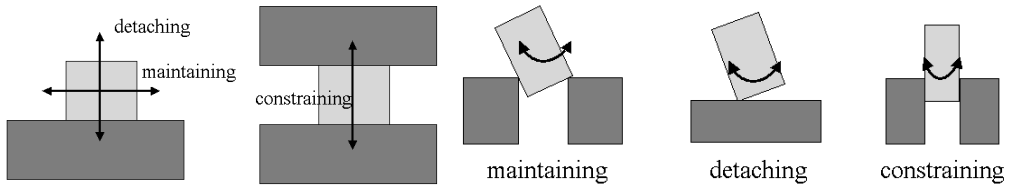
4.1 Representation of a Possible Motion

The assembly tasks can be represented by the transitions of the topological contact-relation [13]. For achieving the aimed transitions, the recognition system needs to analyze the possible motion of a manipulated object. The possible motion is represented as non-linear equations. By employing the screw theory, the possible motion can be approximated to linear equations [15], which makes the analysis much easier.

4.2 Features of a Possible Motion

The previous APO system assigns the skills using features that consist of maintaining, detaching, and constraining DOF in translation [1] (Fig. 8 (a)). We extend the analysis by including other 3 DOFs in rotation:

- Maintaining: The DOFs of axis directions are able to rotate maintaining the contact relation (Fig. 8 (b)).
- Detaching: The DOFs of axis directions are not able to rotate maintaining the contact relation (Fig. 8 (b)).
- Constraining: The DOFs of axis directions are not able to rotate (Fig. 8 (b)).



(a) Translation (b) Rotation

Fig. 8. Maintaining, Detaching, and Constraining DOFs in Translation and Rotation

4.3 Singular Cases

We refer to contacts between two convex vertices, between a convex vertex and a convex edge, between two parallel convex edges as *singular contacts*. (See Fig. 9) We assume that those singular DOFs are treated as DOFs having no *singular contact* in the contact state transition network (Fig. 10), so as to analyze contact relations by using the same method. In this case, we refer to these DOFs *singular maintaining*, *singular detaching* and *singular constraining*.

4.4 DOFs Analysis

The change of contact relations leads to change in those DOFs. Among those transitions of DOFs (shown in Fig. 10), the following 3 transitions of DOFs occur toward the direction of movements: *maintaining to detaching*, *maintaining to singular maintaining* and *maintaining to singular detaching*. These 3 transitions are important in the design of sub-skills. The remaining transitions occur in perpendicular dimensions to the direction of movement. The transitions are important for monitoring those actions using sensor feedback.

4.5 Designing Sub-skills

Maintaining to Detaching The motion shown in Fig. 11, 12 leads to the transition from maintaining to detaching. We call the motion shown in Fig. 11 *make-contact in translation* and the motion shown in Fig. 12 *make-contact in rotation*.

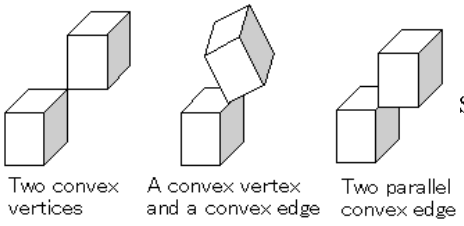


Fig. 9. Three Singular Contacts

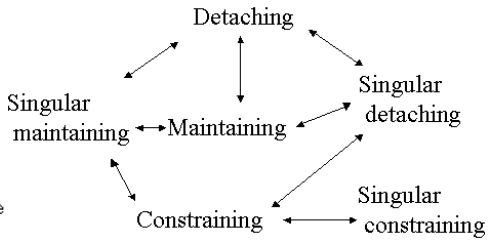


Fig. 10. possible Transitions of DOFs

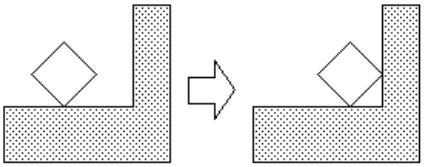


Fig. 11. Make-contact in Translation

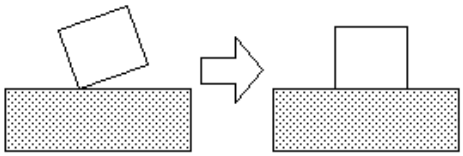


Fig. 12. Make-contact in Rotation

Maintaining to Singular Maintain The motion shown in Fig. 13 and 14 leads to the transition from maintain to singular maintain in translation. We call the motion shown in Fig. 13 *slide in translation* and the motion shown in Fig. 14 *slide in rotation.*

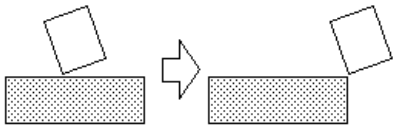


Fig. 13. Slide in Translation

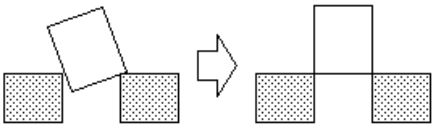


Fig. 14. Slide in Rotation

Maintaining to Singular Detaching The motion shown in Fig. 15 leads to the transition from maintain to singular detaching in translation. This motion looks like the motion combining make-contact and slide in translation. In the case of rotation, the situation is the same. But since this motion can be performed similarly by the make-contact method, we do not take up this motion.

4.6 Rule for Assigning Sub-skills

The rule for assigning a make-contact sub-skill is following:

- If maintaining DOF in translation is changed into (singular) detaching DOF, a make-contact in translation sub-skill is assigned.
- If maintaining DOF in rotation is changed into (singular) detaching DOF, a make-contact in rotation sub-skill is assigned.

In the case of a slide sub-skill as shown in Fig. 13 and 14, maintaining DOF in both translation and rotation is changed into singular maintaining DOF, so we cannot distinguish between translation and rotation from that information. But using Restricted DOF[14], we can distinguish them. The rule for assigning a slide sub-skill is as following:

When maintaining DOF is changed into singular maintaining DOF,

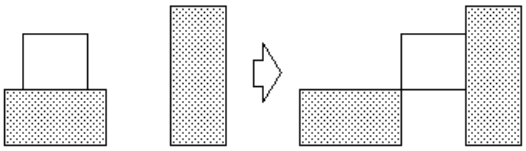


Fig. 15. Maintaining to Singular Detaching

– if restricted DOF in translation increase, a slide in translation sub-skill is assigned.
– if restricted DOF in rotation increase, a slide in rotation sub-skill is assigned.

5 Platform

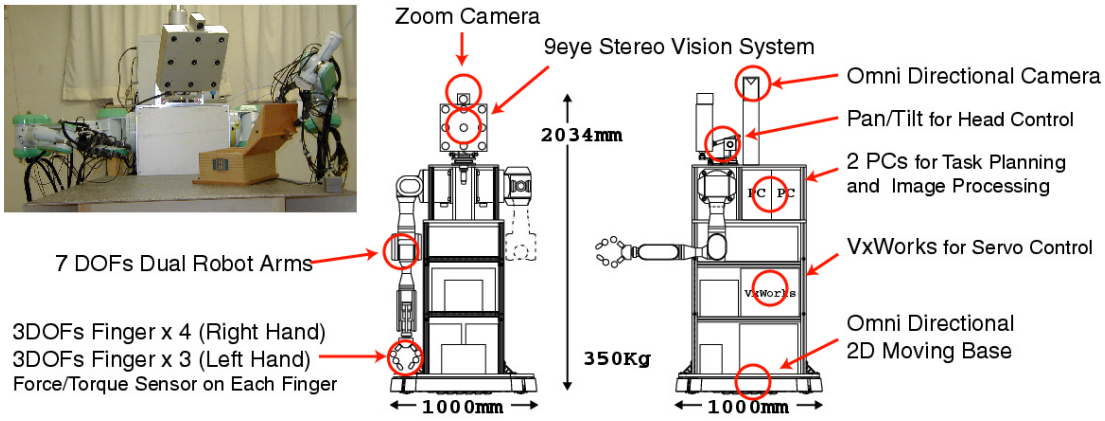


Fig. 16. Platform

We have developed a robot (Fig. 16) as an experimental platform for robot learning and performing human hand-action tasks. For that purpose, we designed the robot to have similar capabilities to humans, including vision, dual arms and upper torso.

The main features of this robot are summarized as follows.

– It is equipped with a 9-eye stereo vision system and other camera systems for object recognition (vision).
– It has dual 7 degrees-of-freedoms (DOFs) robot arms. The right arm has a hand with 4 fingers and the left arm has a hand with 3 fingers. Each finger has 3 DOFs and a Force/Torque sensor on its tip (arms and hands).
– The omni directional moving base allows the robot move freely on a 2D plane in order to move the view point and the arms in any position (upper torso).
– CORBA-based [10] software architecture enables the robot to be programmed easily on multi-machines connected by a network and to be connected from new exterior devices such as data gloves.

5.1 Vision

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

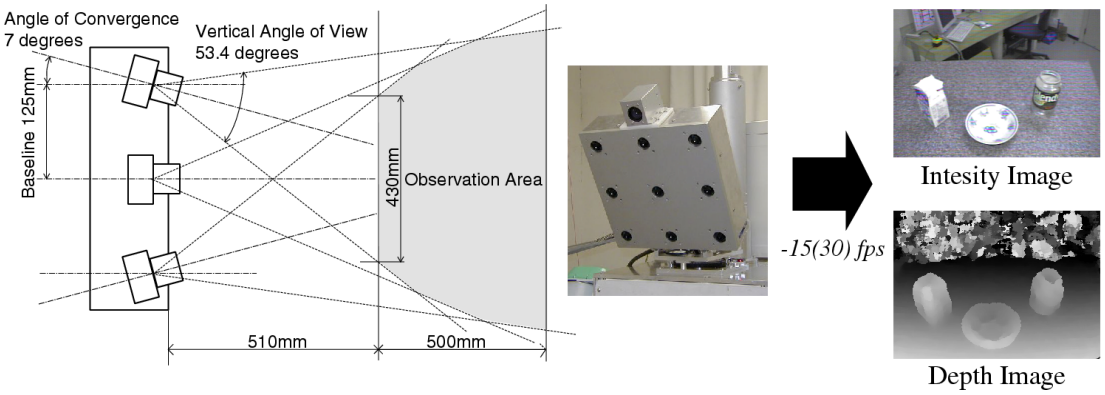


Fig. 17. Stereo Vision System

9-eye Stereo Vision System 9-eye Stereo Vision System The 9-eye stereo vision system is utilized to analyze the environment around the robot precisely in real time. The system is a product of Komatsu Ltd. [8] which adopted a multi-baseline approach [7] and has following features.

- **Robust stereo matching**
  It processes 8 stereo pairs (the center camera and the exterior cameras) simultaneously and chooses the most reliable stereo pair on each pixel in the depth data.
- **Real-time processing on hardware chip**
  The stereo calculation is processed on a hardware chip and can produce the resulting depth data (280×200 points) in real time (15fps, up to 30fps).
- **Easy customization of camera configuration**
  Users can easily rearrange the camera configuration to match their requirements. We extended the baseline and tilted the exterior cameras inward so that the stereo system can produce high resolution depth images of short distances, which are suitable for our robot. The measurable range is changeable and in this study we set up the valid range from 510mm to 1010mm, which is the closest. Fig. 17 shows the camera configuration and the produced intensity and depth image.

5.2 Arms and Hands

As described in the previous chapter, we focus on the learning and performing of human hand-actions by a robot; the manipulation capability of the robot is essential in performing the same task as that performed by a human. This robot has dual 7DOFs PA10 robot arms (made by Mitsubishi Heavy Industry Ltd.) which have enough DOF to move the robot hand to a wide area of 3D space (position and orientation). As a substitution for a human hand, a robot hand with fingers is attached to the tip of the robot arm. The right hand has 4 fingers and the left hand has 3 fingers. Each finger has 3 joints (3 DOFs) and is equipped with a 6 axises Force/Torque sensor on its tip. These fingers are arranged to face each other so as to enhance the grasping and manipulation capability. Fig. 18 shows the limit of working area of the robot arm in the level of the robot shoulder.

5.3 Upper Torso

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

To give the robot similar ability, we utilized the holonomic omni-directional vehicle [9] at the bottom of the robot. With this vehicle, the robot can move and rotate in any direction at any position on the floor. So the robot can change its point of view and the task space dynamically according to the task context.

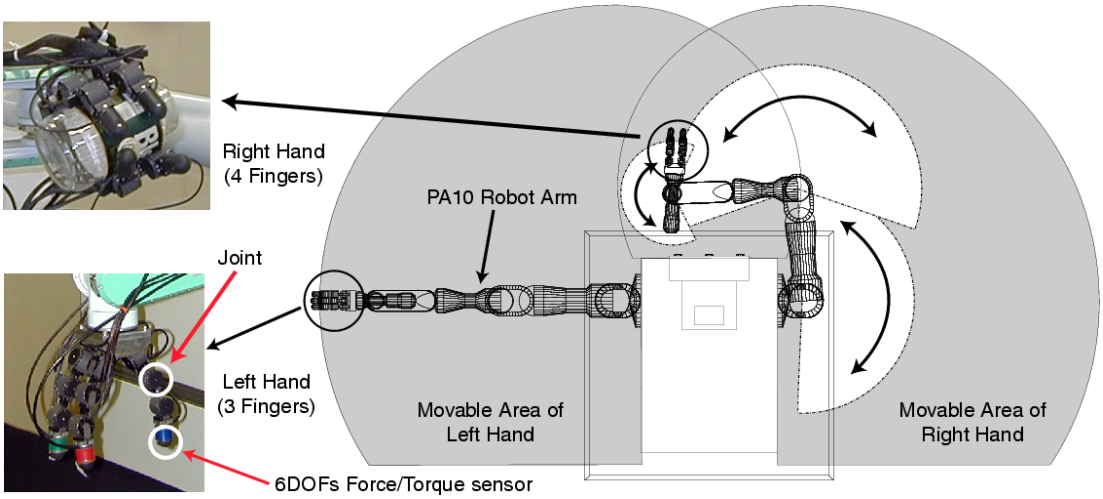


Fig. 18. Arm and Hand Configuration

5.4 CORBA based Software Architecture

Software Architecture The robot is controlled by distributed software components on different machines connected by a network. Each hardware device equipped to the robot has its specific control software process (abbreviated to "CS", Component Server), and a brain process chooses the necessary CSs and accesses each CS across a network (LAN) to control the hardware resources in the robot. These hardware resources include PA10 robot arms, 9-eye stereo vision system, data gloves, etc. (Table. 4). The main reasons for constructing the robot software architecture by these distributed components are described below:

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

Each CS manages a specific hardware device directly (Table 4), so it must reside in the same machines in which the device is installed, while a brain can be on any machine. Each CS provides a set of Application Program Interfaces (APIs) in order to be accessed by a brain process. We implemented these APIs by means of Common Object Request Broker Architecture (CORBA) technology.

CORBA For a communication middle-ware between brains and CSs, we adapted CORBA [10] technology which is a distributed object computing infrastructure being standardized by the Object Management Group (OMG).

In CORBA, a communication between a client object (brain) and a server object (CS) is handled by an Object Request Broker (ORB). Using an ORB, a client can transparently invoke a method on a server object, which can be on the same machine or across a network. The ORB accepts the call and is responsible for finding an object that can implement the request, pass it the parameters, invoke its method, and return the results. The interface of the server object is strictly defined by Interface Definition Language (IDL) by an architecture-independent manner and the client does not have to be aware of where the object is located, its programming language, its operating system, or any other system aspects that are not part of an object's interface. In so doing, the ORB provides interoperability between applications on different machines in heterogeneous distributed environments.

The robot consists of different operating systems (Linux and Windows NT) on separate machines and the combination of brains and CSs are vary according to tasks and number of users. For this reason, we utilize CORBA technology as a base infrastructure of the robot software and define interfaces (APIs) of CSs by CORBA. By defining interfaces in IDL, we can easily attach new exterior devices (such as data gloves) to the robot.

Table 4. Component Servers

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

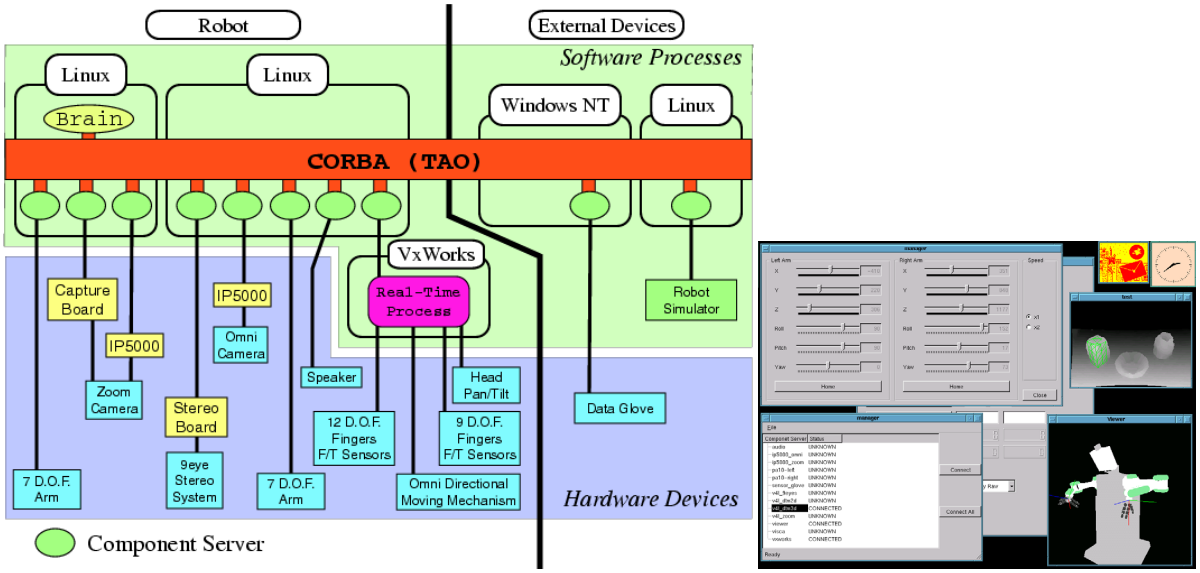


Fig. 19. Robot Architecture

We utilized TAO ORB [11] as an implementation of CORBA. TAO is the result of research on high-performance and real-time CORBA which is freely available. TAO supports a wide range of operating systems (including Linux and Windows NT) on several hardware architectures. The supported programming language is C++; therefore, the robot software architecture has been developed in C++. The left side of Fig. 19 shows our software architecture described in this section. The right side of Fig. 19 shows the GUI controller of the robot. The robot can either behave autonomously or be controlled by a human with those GUI controllers.

6 Experiment of Task Acquisition

To validate the presented two level acquisition methods, we performed the following two experiments with the developed platform.

6.1 Task Level Acquisition: Recognition and Performance by Robot

To determine the validity of the human task model, we set up an experiment. In this experiment, a human held the container A in one hand and poured the content of B ,which was held by the other hand, into the container A. The robot observed the task and constructed a human task model. Then, using the constructed task model, the robot performed the same task in a different environment.

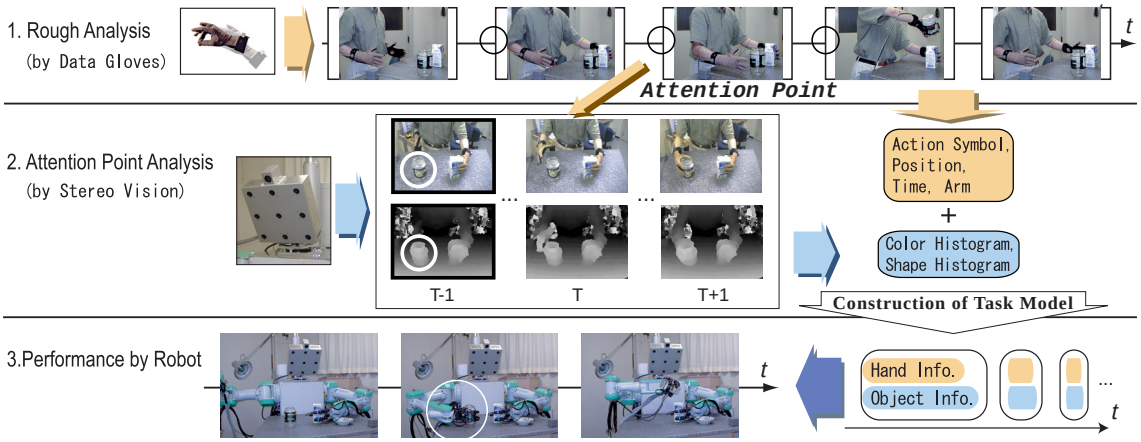


Fig. 20. Experiment

Recognition of Human Task First, the robot observed the human task through data gloves. The robot constructed a rough human task model by HMM-based gesture spotter in real-time and then found APs (first part of Fig. 20).

Second, the robot applied detailed analysis on the depth and color images at each AP and calculated shape and color histogram (second part of Fig. 20). This analysis added the object model information to the human task model.

Performance by Robot Then the robot performed the same task using the constructed human task model (third part of Fig. 20).

To examine the applicability of this abstract human task model in a different environment, we added a new object "Dish" which was not present at the time of demonstration, and changed the arrangement of the objects on the table.

The result shows that the robot properly chose the correct objects and completed the same task.

We performed several experiments and noticed that, when the target object was out of reach of the arm described in the model, the robot omitted the "Hand" attribute whose priority was low and tried to reach the object by using its other arm. This shows the effectiveness of the priority term.

The priority term changes the degree of the abstraction of each hand action adapting to the environment, so that the robot can complete the task while maintaining the model description.

6.2 Behaviour Level Acquisition: Recognition of Peg-in-Hole Task

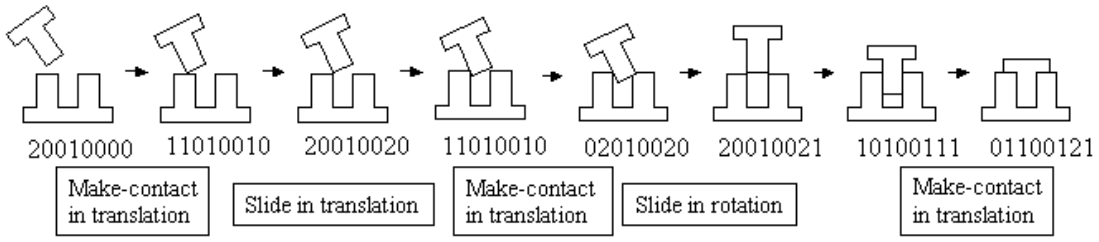


Fig. 21. Maintaining, detaching, and constraining DOFs of translation and rotation, and Restricted DOFs in translation and rotation

Consider the peg-in-hole task shown in Fig. 21. First, the system acquired the transitions of topological contact relation by observing human performance. Next, it computed each DOF in each contact relation. Then, it assigned sub-skills in the robot by applying the rule. Finally, the robot executed the same task using assigned sub-skills (shown in Fig. 22).

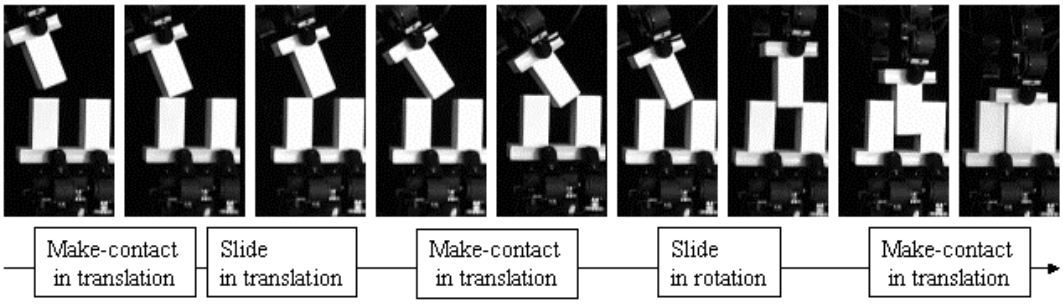


Fig. 22. peg-in-hole task

7 Conclusion

We have developed a task and behavior learning robot. We mainly concentrated on human hand-action tasks and developed methods to obtain such task and behavior models through observation. The approach has two layers of analysis: task level (what to do), and behavior level (how to do it).

For a task level acquisition, which obtains what-to-do information, we proposed a novel method of constructing a human task model by attention point (AP) analysis using data gloves and vision system. This task model is flexible enough to be applicable in a different environment from those demonstrated. We showed the validity of this model by an experiment in which a real robot constructed a model of human hand-action from observation and performed the same task successfully in a different environment using the acquired task models.

For a behavior level acquisition, which obtains how-to-do information, we proposed a system which analyzes a trajectory of observed human hand motion and segments the entire motion into classified states according to the transitions of the face-contact relations of the target objects. These classified states were mapped to specific sub-skills (robot motions). Then the system automatically generated the robot behavior by assigning sub-skills and the robot performed the peg-insertion behavior successfully.

As an experimental platform for robot learning and performing of human tasks, we developed a human form robot which has similar capabilities to the human upper body; the robot is equipped with multi vision systems, dual dexterous arms and hands, and an omni directional moving base.

In the future, we are planning to combine those two-layered acquisition methods as follows: First, the task level acquisition constructs task models to perform the entire task to be adapted to the environment. It also extracts special APs which require behavior level acquisition. Second, the behavior level acquisition analyzes those APs closely and obtains suitable motion sequence (sub-skill). This two-layer approach should complete a learning robot that can acquire a human task through observation.

Acknowledgment

This work is supported, in part, by Japan Society for the Promotion of Science (JSPS) under the grant RFTF 96P00501, and, in part, by Japan Science and Technology Corporation (JST) under Ikeuchi CREST project.

References

1. K. Ikeuchi and T. Suehiro: “Toward an Assembly Plan from Observation Part I: Task Recognition With Polyhedral Objects,” *IEEE Trans. Robotics and Automation*, 10(3):368–384, 1994.
2. Y. Kuniyoshi, M. Inaba, and H. Inoue: “Learning by watching,” *IEEE Trans. Robotics and Automation*, 10(6):799–822, 1994.
3. H. Kimura, T. Horiuchi and K. Ikeuchi: “Task-Model Based Human Robot Cooperation Using Vision,” *IROS ’99*, 2:701–706, 1999.
4. J. Takamatsu, H. Tominaga, K. Ogawara, H. Kimura and K. Ikeuchi: “Symbolic Representation of Trajectories for Skill Generation,” *IEEE Trans. Robotics and Automation*, 4:4077-4082, 2000.
5. C. G. Atkeson, S. Schaal: “Learning Tasks From A Single Demonstration,” *IEEE Trans. Robotics and Automation*, 1706–1712, 1997.

6. M. R. Cutkosky, “On Grasp Choice, Grasp Models, and the Design of Hands for Manufacturing Tasks,” *IEEE Trans. on Robotics and Automation*, 5(3):269–279, 1989.
7. S. Kimura, T. Kanade, H. Kano, A. Yoshida, E. Kawamura, and K. Oda: “CMU video-rate stereo machine,” *Proc. of Mobile Mapping Symposium*, 1995.
8. http://www.komatsu.co.jp/research/study56.htm
9. S. Hirose and S. Amano: “The VUTON: High Payload High Efficiency Holonomic Omni-DirectionalVehicle,” *Proc. Int. Symp. on Robotics Research*,pp.253-260, 1993.
10. Common Object Request Broker Archictecture, OMG, July, 1995.
11. http://www.cs.wustl.edu/ schmidt/TAO.html
12. M. D. Wheeler and K. Ikeuchi: “Sensor Modeling, Probabilistic Hypothesis Generation, and Robust Localization for Object Recognition”, *IEEE Trans. PAMI*, 17(3):252–265, 1995.
13. S. Hirai, H. Asada, and H. Tokumaru: “Kinematic Analysis of State Transitions in Assembly Operations and Automatic Generation of Transition Network,” *SICE*, Vol.24, No.4, pp.84-91, 1988. (in Japanese)
14. T.Suehiro: “Study of an advanced manipulation system,” *Researches of the Electrotechnical Laboratory*, No.912, June, 1990. (in Japanese)
15. B.Roth: “An Extension of Screw Theory,” Journal of Mechanical Design, Vol.103, pp.725-735, 1981.
16. T. Starner and A. Pentland: “Real-time American Sign Language recognition from video,” *IEEE International Symposium on Computer Vision*, Coral Gables, FL. 265–270, 1995.
17. K. M. Knill and S. J. Young: “Speaker Dependent Keyword Spotting for Accessing Stored Speech,” *Cambridge University Engineering Dept., Tech. Report*, No. CUED/F-INFENT/TR 193, 1994.
18. S. J. Young: “Hidden Markov Model Toolkit V2.2.,” Entropic Research Lab Inc., Washington DC, January 1999.