Toward Automatic Robot Instruction from Perception—Mapping Human Grasps to Manipulator Grasps

Sing Bing Kang and Katsushi Ikeuchi, *Senior Member, IEEE*

Abstract— **Conventional methods for programming a robot either are inflexible or demand significant expertise. While the notion of automatic programming by high-level goal specification addresses these issues, the overwhelming complexity of planning manipulator grasps and paths remains a formidable obstacle to practical implementation. Our approach of programming a robot is by direct human demonstration. Our system observes a human performing the task, recognizes the human grasp, and maps it onto the manipulator. Using human actions to guide robot execution greatly reduces the planning complexity. Subsequent to recording the human task execution, temporal task segmentation is carried out to identify task breakpoints. This step facilitates human grasp recognition and object motion extraction for robot execution of the task. This paper describes how an observed human grasp can be mapped to that of a given general-purpose manipulator for task replication. Planning the manipulator grasp based upon the observed human grasp is done at two levels: the functional and physical levels. Initially, at the functional level, grasp mapping is achieved at the virtual finger level; the virtual finger is a group of fingers acting against an object surface in a similar manner. Subsequently, at the physical level, the geometric properties of the object and manipulator are considered in fine-tuning the manipulator grasp. Our work concentrates on power or enveloping grasps and the fingertip precision grasps. We conclude by showing an example of an entire programming cycle from human demonstration to robot execution.**

I. Introduction

ROBOT programming is the act of specifying actions or goals for the robot to perform or achieve in order to carry out a useful task. It is an essential and necessary component of task automation. Robot programming methods can be generally subdivided into four categories (which are not mutually exclusive): teleoperation, teaching, robot-level textual programming, and automatic programming. Teleoperation basically refers to the direct control of a remote manipulator (the slave) by manipulating a master device; the control signals are usually low-level and are difficult to interpret. The teaching and textual programming methods are by far the most pervasive in both the industrial and academic environments. In teaching methods, the robot or manipulator learns its trajectory either through a teach pendant or actual guidance through the sequence of operations. Portability between different systems is a problem in both teleoperation and teaching methods. Robot-level textual programming refers to the approach of hand-coding programs to enable the robot to execute motions in robot joint and task spaces. While this approach of robot programming is flexible, it requires expertise and often a long development time. In contrast, automatic programming conceptually requires only the object description and high-level task specifications in order to generate the control command sequences to the robot. In general, the realization of a practical system with automatic programming is difficult because of the complexity of path and grasp planning and high-level task goal interpretation.

The approach that we adopt in robot programming is the *Assembly Plan from Observation* (APO) paradigm proposed by Ikeuchi and Suehiro [9]. In this approach, task programming is performed by demonstrating the task to the system rather than by the traditional method of hand-coding. The key idea is to enable a system to observe a human performing a task, analyze it, and perform the task with minimal human intervention. This method of task programming would obviate the need for a programmer to explicitly describe the required task.

A similar approach to APO was taken by Kuniyoshi *et al.* [18] who developed a system which emulates the performance of a human operator. However, their system is restricted to pick-and-place operations. Takahashi and Ogata [31] use the virtual reality environment as a robot teaching interface. The operator's movements in the virtual reality space via the VPL dataglove are interpreted as robot task-level operations by using a finite automaton model. Hamada *et al.* [7], on the other hand, specify commands such as "carry(cap, path, body)" to interactively carry out operations. This is first simulated in a "task mental image" comprising *a priori* action knowledge and a graphical display. Subsequently, the operations are carried out by the manipulator with the aid of a vision system that matches the "mental image" models with the real objects.

Manuscript received January 11, 1995; revised April 8, 1996. This work was supported in part by the Advanced Research Projects Agency under the Department of the Army, the Army Research Office under Grant DAAH04-94-G-0006, the Advanced Research Projects Agency under the U.S. Air Force, the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), Wright-Patterson AFB under Contract F33615-90-C-1465, ARPA Order 7597, and the National Science Foundation under Contract CDA-9121797. This work was presented in part at the IEEE/RSI/GI International Conference on Intelligent Robots and Systems (IROS'94), San Diego, CA, May 1994. This work was performed at Carnegie Mellon University, Pittsburgh, PA. This paper was recommended for publication by Associate Editor R. Howe and Editor A. Goldenberg upon evaluation of reviewers' comments.

S. B. Kang is with the Digital Equipment Corporation, Cambridge Research Lab., Cambridge, MA 02139 USA.

K. Ikeuchi is with the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213 USA.

Publisher Item Identifier S 1042-296X(96)07243-6.

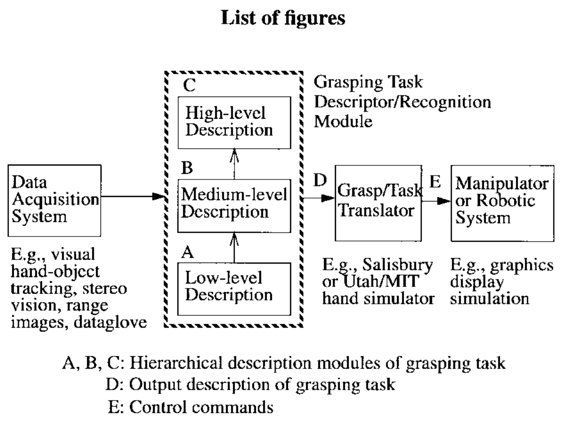

List of figures
C
High-level Description
Grasping Task Descriptor/Recognition Module
B
Data Acquisition System
Medium-level Description
D
Grasp/Task Translator
E
Manipulator or Robotic System
A
Low-level Description
E.g., visual hand-object tracking, stereo vision, range images, dataglove
E.g., Salisbury or Utah/MIT hand simulator
E.g., graphics display simulation
A, B, C: Hierarchical description modules of grasping task
D: Output description of grasping task
E: Control commands

Fig. 1. System block diagram.

A. Automatic Robot Instruction via APO

In the APO approach to robot programming, the human provides the intelligence in choosing the hand (end-effector) trajectory, the grasping strategy, and the hand-object interaction by directly performing them. This alleviates the problems of path planning, grasp synthesis, and task specification. Our system, which uses the APO approach, is shown in Fig. 1. We provide an overview of the entire system here.

The data acquisition system extracts data from the environment; it provides low-level information on the hand location and configuration, objects on the scene, and contact information between the hand and object of interest. The grasping task descriptor/recognition module translates low-level data into higher levels of abstraction to describe both the motion and actions taken in the task. The output of the grasping task descriptor module is subsequently provided to the task translator which in turn creates commands for the robot system to replicate the observed task. The representations given in submodules A, B, and C are independent (up to some level of task specification) of the manipulator used in the backend of the system, while the converse is true of the translator. The translator decides the information level at which the robot is programmed. At the lowest level, it produces the manipulator trajectory based directly on that of the human hand. At intermediate and high levels, it uses the concept of the virtual finger[1] and identity of the human grasp to synthesize the manipulator grasp. This paper shows how human grasp description (output of grasping task descriptor module) can be used by the grasp/task translator to map the human grasp to that of the manipulator. We are interested in using human cues as hints in planning manipulator grasps and paths.

B. Organization of Paper

This paper starts with a brief discussion on grasp planning issues. Before grasp mapping can be accomplished, human grasp descriptions have to be acquired first. A brief review of how a human grasp is recognized from observation is thus given. The general approach of planning manipulator grasps from human grasp description is delineated next. Grasp planning depends on the type of grasp used—both approaches of planning volar grasps and fingertip precision grasps are given in this paper. Example grasp mappings are subsequently shown. The robot system (comprising the PUMA arm and Utah/MIT hand) used as part of the testbed for proof of concept is also described here. We also describe an example of a full programming cycle, from human demonstration to robot execution, before presenting concluding remarks.

A major component in task planning is planning the manipulator grasp. In the following section, we briefly discuss some of the issues involved in grasp planning and reiterate the justification for our approach.

[1]Defined as a collection of real fingers acting in a similar manner against the surface of an object in a grasp [1].

II. Grasp Planning

Grasp planning involves determining the hand placement relative to the object of interest as well as the posture of the hand assumed in grasping the object, both in a manner consistent to the requirements of the task (examples include [5], [19], [24], [29]). It is computationally intensive due to both the large search space and the incorporation of geometric and task-oriented constraints. Globally optimal grasp planning using comprehensive or exhaustive search is not feasible, especially with a high number of joints of the robot arm and hand.

The search space for an optimal grasp can be reduced in size or dimensionality by using certain heuristics or structural constraints. For a two-finger gripper, for example, the surface element pair selection can be guided by a series of constraints, such as face parallelism, permissible face separation, and nonzero overlap between projected faces [34]. By imposing fixed contact locations or fixed types of grasp, Pollard computes a 6-D projection of the 15 DOF configuration space for the Salisbury hand [27]. This reduced space represents the space of wrist poses for geometrically feasible grasps, and is used to search for a globally optimal grasp. In addition, a popular strategy for reducing the complexity of planning the dextrous hand grasp is by using a small set of predetermined grasps or grasping behaviors (e.g., [3], [6], [28]).

In our approach of robot programming by human demonstration, grasp planning is made more tractable by using cues extracted from human execution of the grasp. Subsequent to temporal segmentation of the recorded task, the human grasp is recognized and the approximate grasp contact locations recovered. This is followed by determining the initial approximate grasp which satisfies kinematic and geometric constraints, and subsequently performing local optimization of the grasp using the approximate grasp as the starting condition as well as a task-oriented objective function. Our grasp mapping approach is described in detail in the following sections. Prior to grasp mapping, however, we need to produce descriptions of the observed human grasp.

III. Recognizing a Grasp from Observation

In this section, we briefly describe how an observed human grasp is recognized (full details can be found in Kang and Ikeuchi [16]). The descriptions of the human grasp are subsequently used as cues for the robot system to plan the

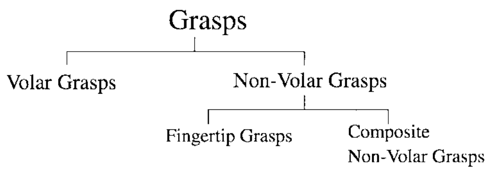


Fig. 2. Grasp classification for recognition.

manipulator grasp. Central to the human grasp recognition scheme is a 3-D grasp representation called the *contact web*.

A. The Contact Web

A contact web is defined as a 3-D graphical structure connecting the effective points of contact between the hand and the object grasped. When parts of a finger or palm make contact with the grasped object, the actual contact area is finite. A point contact is useful in conceptually representing the contact between the phalangeal segments and palm and the object because of ease of representation and analysis, and accommodation of uncertainty in grasping. The shape and cardinality of the contact web yield important information about the type of grasp effected.

B. Grasp Classification and Recognition

The main classification of grasps using the contact web is shown in Fig. 2. Grasps are initially dichotomized into volar and nonvolar grasps according to whether there is contact between the palm and object. Nonvolar grasps are further categorized as either fingertip grasps (in which only the fingertips are involved) or composite nonvolar grasps (in which a mix of fingertips and more proximal finger segments are involved).

Nonvolar grasp classification and recognition follows Fig. 3; classification and recognition is based on the number of fingers and finger segments contacting the object as well the shape of the contacts. (In Fig. 3, only the represented finger segments in contact with the object are shown. The small filled circles represent the fingertips, excluding the thumb, and the small hollow circles represent the proximal finger segments. The small filled box represent the tip of the thumb.) Volar grasp classification and recognition is not as straightforward due to the many finger segments involved in the grasp. To aid in volar grasp recognition, we use the notion of the *virtual finger*.

C. The Virtual Finger and Grasp Abstraction Hierarchy

Arbib *et al.* [1] introduced the concept of the virtual finger: a functional unit comprised of at least one real physical finger (which may include the palm). The real fingers comprising a virtual finger act in unison to apply an opposing force on the object and against the other virtual fingers in a grasp. This concept replaces the analysis of the mechanical degrees of freedom of individual fingers by the analysis of the functional roles of forces being applied in a grasp [1].

By analyzing the contact web, the medium level grasp concept of the virtual finger space can be described. The virtual finger is used in characterizing the type of grasp and indicating the functionality of the grasp.

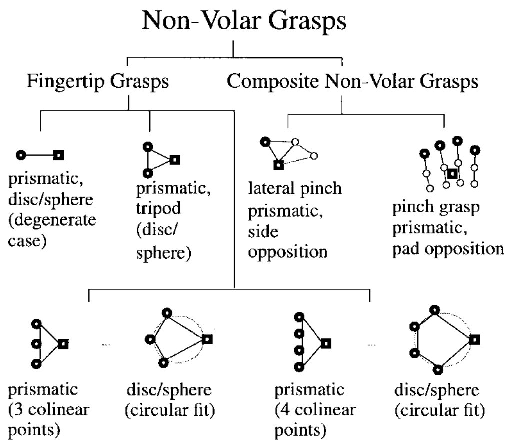

Non-Volar Grasps
Fingertip Grasps
Composite Non-Volar Grasps
prismatic, disc/sphere (degenerate case)
prismatic, tripod (disc/ sphere)
lateral pinch prismatic, side opposition
pinch grasp prismatic, pad opposition
prismatic (3 colinear points)
disc/sphere (circular fit)
prismatic (4 colinear points)
disc/sphere (circular fit)

Fig. 3. Nonvolar grasp classification.

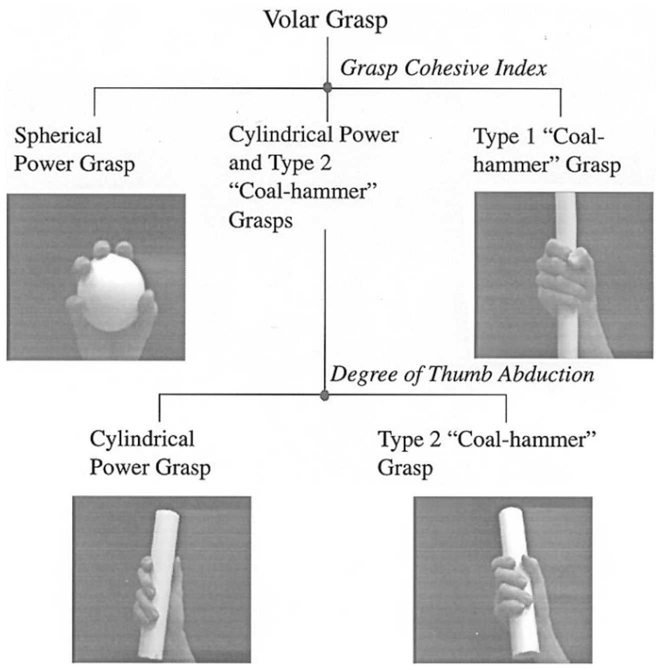

Volar Grasp
Grasp Cohesive Index
Spherical Power Grasp
Cylindrical Power and Type 2 “Coal-hammer” Grasps
Type 1 “Coal-hammer” Grasp
Degree of Thumb Abduction
Cylindrical Power Grasp
Type 2 “Coal-hammer” Grasp

Fig. 4. Volar grasp classification.

Mapping the real fingers to virtual fingers has been described in Kang and Ikeuchi [16]. The mapping results in what we call the *grasp cohesive index*, which indicates how well the fingers grouped as virtual fingers act in similar manner against the surface of the object. Using the grasp cohesive index and the degree of thumb abduction, we can then classify volar grasps as shown in Fig. 4. (For a detailed description of the types of volar grasps, see [16].) Nonvolar grasps are, on the other hand, classified according to the number of finger segments touching the object and the shape assumed by fingertip position as shown in Fig. 3.

Once the grasp has been identified, the grasp abstraction hierarchy can be constructed as depicted in Fig. 5 [15]. The levels of description correspond to those depicted in the grasping task module shown in Fig. 1. The hierarchy contains information from the low-level of finger segment pose and joint angles to the identity of the grasp itself. This hierarchy is used by the task translator module of the robot system to plan the manipulator grasp.

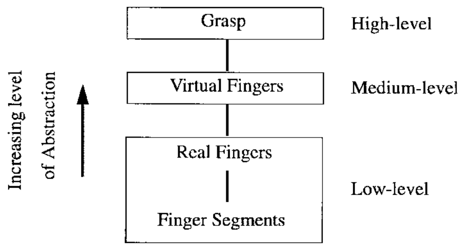


Fig. 5. Grasp abstraction hierarchy.

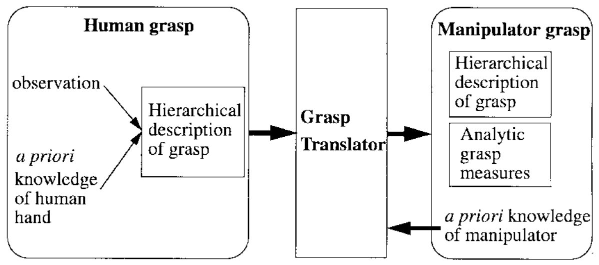

Human grasp
observation
a priori knowledge of human hand
Hierarchical description of grasp
Grasp Translator
Manipulator grasp
Hierarchical description of grasp
Analytic grasp measures
a priori knowledge of manipulator

Fig. 6. Grasp mapping strategy.

IV. Mapping Grasps

A. General Approach

Once we have extracted the human grasp descriptions from observation, the robot system then proceeds to plan the manipulator grasp. Planning the manipulator grasp comprises the following steps:

1 *Local Functional Mapping*: This is done by observing the type of grasp used by the human, and the functions and grouping of the palm and individual fingers. The latter is determined using the real finger-to-virtual finger mapping described in Kang and Ikeuchi [14], [16]. The mapping of the manipulator end-effector to the virtual fingers depends on the effective cohesive index.

2 *Adjustable or Gross Physical Mapping*: This operation is carried out using the outcome of the local functional mapping and the initial location of the manipulator (near the pose of the human hand while carrying out the task). It results in an approximate form of the manipulator grasp of the object which is geometrically feasible.

3) *Fine-Tuning or Local Adjustment of Grasp*: The grasp determined from steps 1 and 2 may not exhibit static stability or that it may not be locally optimal (according to a chosen task-related criterion). By analyzing the criterion and iteratively perturbing the initial grasp configuration, we arrive at a locally optimal grasp. In the case of the volar grasp which involve high kinematic constraint, this step is skipped.

Step 1 corresponds to grasp planning at the functional level whereas steps 2 and 3 correspond to grasp planning at the physical level.

In order to provide both a means for locally adjusting the grasp and for checking the appropriateness of the resulting manipulator grasp, we make use of certain grasp analytic measures.

B. Analytic Measures of a Grasp

Cutkosky and Howe [5] summarizes a number of analytic measures that have been used by various researchers in their analyzes of grasps, several of which are for the purpose of grasp synthesis. Examples of such analytic measures are manipulability, grasp isotropy, force and form closure, internal forces, and resistance to slipping.

The overall strategy of mapping grasps is depicted in Fig. 6. The human grasp can be described in a hierarchical fashion. With *a priori* knowledge of the human hand and data derived from observation, certain analytic grasp measures can be extracted (such as connectivity and mobility). Armed with these information, together with *a priori* kinematic, structural and geometric knowledge of the manipulator, the grasp translator would then attempt to produce a manipulator grasp with compatible analytic grasp measures. In our work, the manipulability measure is used as the criterion used in generating the locally optimal manipulator grasp.

The functions of the hierarchical description of the human grasp are:

1.) *Local functional mapping*;
2.) *Gross physical mapping.*

The functions of the analytic grasp measures are:

1) *Verification of appropriateness of gross physical mapping*;
2) *Local fine-tuning or perturbation of grasp to a locally optimal posture.*

C. Local Functional Mapping

This first step in mapping grasps is common to all types of grasps, i.e., volar and precision grasps. The local functional mapping associates the manipulator fingers to those of the hand, and is generally different for different grasps. To guide the local functional mapping, we first assign manipulator fingers as either a primary finger, secondary finger, or a palm.

Assigning Manipulator Fingers: Prior to mapping the manipulator fingers to those of the human in the observed grasp, the manipulator finger has to be assigned as one of the groups of fingers:

- *Primary finger/s*: A finger in this group would correspond to the human thumb, which has at least the same number of degrees of freedom as the other fingers. It serves not only to secure the hold of the object against the palm and/or other fingers, but also to manipulate the object.
- *Secondary finger/s*: This group would correspond to the fingers of the hand other than the thumb. This group of fingers would serve to secure the hold of the object against the palm and/or primary finger/s.

A special case of a "finger" which is also assigned is the palm of the robot hand; it is designated as such because it is passive with no local degree of freedom (i.e., discounting the global 6 DOF of the manipulator).

Manipulators are designed with specific functions or level of dexterity in handling objects. The repertoire of grasping abilities may be as limited and simple as just clamping the object between its jaws as a means of securing a hold

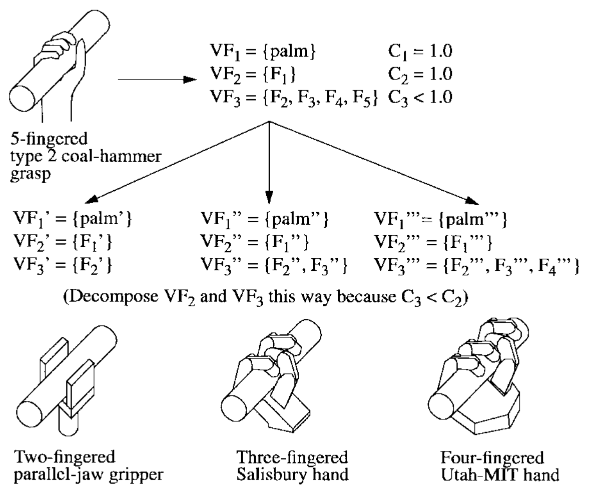

VF1 = {palm} C1 = 1.0
VF2 = {F1} C2 = 1.0
VF3 = {F2, F3, F4, F5} C3 < 1.0
5-fingered type 2 coal-hammer grasp
VF1' = {palm'} VF2' = {F1'} VF3' = {F2'}
VF1'' = {palm''} VF2'' = {F1''} VF3'' = {F2'', F3''}
VF1''' = {palm'''} VF2''' = {F1'''} VF3''' = {F2''', F3''', F4'''}
(Decompose VF2 and VF3 this way because C3 < C2)
Two-fingered parallel-jaw gripper
Three-fingered Salisbury hand
Four-fingered Utah-MIT hand

Fig. 7. Mapping grasps according to cohesive indices of virtual fingers.

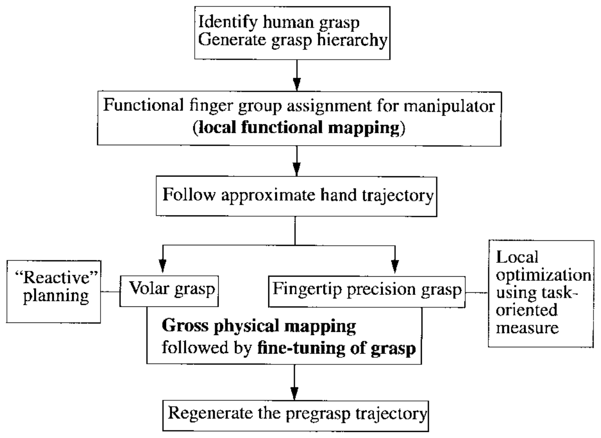

Identify human grasp
Generate grasp hierarchy
Functional finger group assignment for manipulator
(local functional mapping)
Follow approximate hand trajectory
"Reactive" planning
Volar grasp
Fingertip precision grasp
Local optimization using task-oriented measure
Gross physical mapping
followed by fine-tuning of grasp
Regenerate the pregrasp trajectory

Fig. 8. Scheme for generating the manipulator grasp and pregrasp trajectory (the grasp mapping steps in bold).

on it (as in the parallel-jaw gripper). The manipulator may be anthropomorphic to some degree and be able to mimic certain grasps of the human hand; examples of such hands are the Utah/MIT hand [10], Salisbury hand [21], and Okada hand [25]. It is usually simple to label *a priori* each of the manipulator fingers as either a primary or secondary finger. It is also conceivable that the fingers can also be classified by comparing the similarity in shape and orientation of the velocity or manipulability ellipsoids of the fingers with the hand assuming a certain pose (for example, at "mid-posture," with all the finger joint angles midway between the joint limits). We chose to specify the type of finger *a priori*.

Once the primary and secondary fingers have been identified, the hand-to-manipulator finger mapping can proceed with the aid of another type of mapping. This mapping relates real fingers to groups of functionally equivalent fingers called virtual fingers. A consequence of this real finger-to-virtual finger mapping is the *cohesive index*. In short, the cohesive index is associated with the degree to which the real fingers in a virtual finger act in a similar manner; this is determined by comparing the object normals at the contact points. A cohesive index of a virtual finger is maximum at unity, and would indicate that the fingers within that virtual finger act exactly alike.

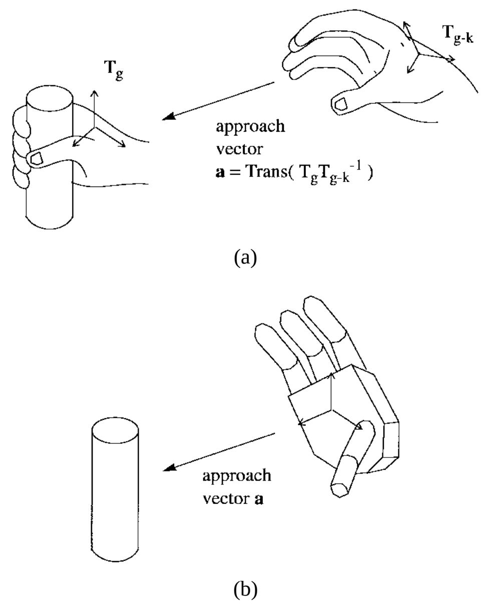

T_g
$T_{g\text{-}k}$
approach vector
$\mathbf{a} = \text{Trans}(T_g T_{g\text{-}k}^{-1})$
(a)
approach vector $\mathbf{a}$
(b)

Fig. 9. Approach alignment: (a) Determining approach vector $\mathbf{a}$, (b) Orienting robot hand to align with approach vector $\mathbf{a}$ (Trans(T) refers to the translation component of transform T).

Using the Cohesive Index to Guide Mapping of Fingers: An indication of how the cohesive index is used to guide the virtual finger mapping between those of the human hand and the manipulator is illustrated in Fig. 7. (F_k is the kth real finger, and VF_n refers to the nth virtual finger with C_n being its associated cohesive index.) The palms, primary fingers, and secondary fingers are mapped appropriately as seen in Fig. 7. The cohesive index serves as a check—the virtual finger corresponding to the group of secondary fingers normally has a cohesive index less than that of the collection of primary fingers. The virtual finger composition determined from the real finger-to-virtual finger mapping take precedence, however.

Once the functional finger assignments have been accomplished, the more detailed manipulator grasp planning can then proceed.

D. Approach in Generating Grasp and Pregrasp Trajectory

The diagram representing the entire manipulator grasp and trajectory planning approach is shown in Fig. 8. Prior to the more detailed manipulator grasp planning, the manipulator assumes a fixed pose and posture relative to the human hand throughout the task execution; local planning is carried out within the vicinity of this pose and posture. Note that by virtue of the different form and function of the volar and fingertip precision grasps, they are planned in a different manner elucidated in subsequent sections.

Once local planning (i.e., fine-tuning of grasp) is done, starting with the pose and posture of the manipulator during the grasping stage, the pregrasp trajectory is then planned backward, with the goals of a smooth and collision-free path. This path would closely follow that assumed by the human hand trajectory. A basic issue in all grasp and task planning, collision detection, is described in [11].

Depending on the type of grasp, the approach in manipulator grasp planning differs. In the subsequent description of such

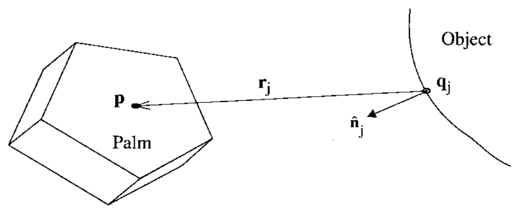


Fig. 10. Determining direction of translation.

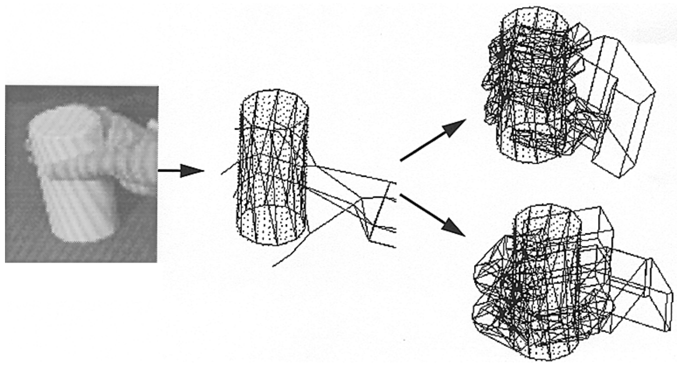


Fig. 11. Results of mapping a cylindrical power grasp for Utah/MIT hand (top-right) and Salisbury hand (bottom-right).

mapping techniques, we concentrate on two classifications of grasps, namely the volar grasp and the fingertip precision grasp.

E. Generating the Volar Grasp

The primary purpose of the volar grasp, in which the hand generally envelops the object, is to ensure a secure hold of the held object at the expense of dexterity [23].[2] As such, we are primarily concerned with the manipulator providing complete or high kinematic restraint on the object. Hence, in the case of the volar grasp, we can dispense with the third step of local grasp adjustment.

General Approach: The approach taken to generate the volar or "enveloping" grasp follows that taken in "reactive" planning (similar to that described by Brock and Salisbury [3]). Note that this approach is taken in generating the grasp, and is not used in the actual execution of the grasp itself. The approach is characterized with the following phases:

1) Aligning the dextrous hand with the approach vector assumed by the human hand just prior to grasping (approach alignment phase).
2) Translating the hand (preserving its orientation) toward the object until the palm touches the object (approach phase).
3) Adjust the hand to "fit" the object to the palm (adjustment phase).
4) Flex the fingers to envelope the object (wrapping phase).

Central to the alignment motions of the dextrous hand is the pivot point. It is predefined to be a point on the surface of the palm between the fingers. Its frame is oriented such that its z-direction points away from the palm and the x-direction points from the thumb (or a primary finger/s) to the rest of the fingers.

[2]Napier [23] used the term "power grasp."

Approach Alignment of Dextrous Hand: In order to minimize the problem of collision with the object during the pregrasp and post grasp phases, we first align the pivot frame of the dextrous hand to the approach vector assumed by the human hand just prior to grasping the object. This is known as the *approach alignment phase*. The approach vector is the translation component of the differential transformation ΔT, where $\Delta T = T_g\, T_{g-k}^{-1}$, T_g is the transformation of the hand at the grasp frame and T_{g-k} is the transformation of the hand k frames prior to the grasp frame (k is chosen to be 2). The alignment is depicted in Fig. 9.

Once the rotational transformation (about the pivot point) has been determined, the dextrous hand is rotated in steps, with the intermediate collision detection with the object. This initial rotation is performed to approximately center the grasp aperture over the object to be grasped. If collision is detected, the dextrous hand is translated or "repulsed" away (preserving its orientation) from the surface of object that makes contact with the hand. The normals of the surface or surfaces in question determine the direction of the repulsion vector.

Adjusting Hand and Enveloping Object: Once the desired orientation of the hand is assumed, planning proceeds with the following behaviors:

1) *Pure Translation Toward the Object (Approach Phase)*: The translational motion is dictated by attractive forces by the object on the pivot point on the dextrous hand. This behavior is active prior to any collision between the object and the manipulator palm. Note that the object is represented as a collection of points sampled at a regular but small interval (about 1.5 mm) on its surface. Associated with each point is the object normal.
   $\mathbf{p}$ is the pivot point on the manipulator palm, and $\mathbf{q}_j$ is the jth sampled object surface point. $\mathbf{r}_j$ is the vector from point $\mathbf{q}_j$ to $\mathbf{p}$, i.e., $\mathbf{r}_j = \mathbf{p} - \mathbf{q}_j$. $\hat{\mathbf{n}}_j$ is the object normal at $\mathbf{q}_j$. The motion at each stage is M

$$M = \delta \frac{\mathbf{d}}{\|\mathbf{d}\|}$$

   where δ is the step size in 3-D space,

$$\mathbf{d} = -\sum_{\hat{\mathbf{n}}_j \cdot \mathbf{r}_j > 0} \omega_j \mathbf{r}_j.$$

   ω_j is the weighting factor dependent on $\|\mathbf{r}_j\|$; the function used is $\omega_j = \|\mathbf{r}_j\|^{-2}$.
2) *Translation and Rotation About the Pivot Point (Adjustment Phase)*: Once collision between the object and the manipulator palm has been detected, this behavior becomes active. The motion associated with this behavior is defined by both the point or points of contact and the entire object. The points of contact determine the motion which tend to push away the manipulator (repulsive forces) while the entire object tends to pull the manipulator (via the pivot point) toward the object. This has the property of trying to align the manipulator relative to the object prior to grasping.

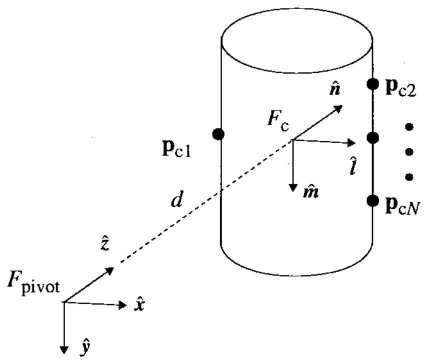


Fig. 12. Orienting the pivot frame with respect to the contact frame.

It should be noted that while these behaviors are active, the posture of the manipulator is that of full extension of the fingers (for parallel jaw gripper, maximally distanced fingers).

The rotation about the pivot point is $\hat{Q}$,[3] where

$$Q = \sum_{\mathbf{q}_k \subset P} Q_k.$$

P is the palm, Q_k is the quaternion associated with the rotation due to the kth contact point on the palm and is given by

$$Q_k = \left[\cos\frac{1}{2}\theta_k,\ \mathbf{m}_k \sin\frac{1}{2}\theta_k\right].$$

$\theta_k = \delta/\|\mathbf{r}_k\|$ and $\mathbf{m}_k = -\mathbf{r}_k \times \mathbf{n}_k$.

3) *Flexion of Fingers Around the Object (Wrapping Phase)*: Once the manipulator has been adequately aligned with respect to the object, the manipulator would then proceed to wrap its fingers around it. This is done by flexing successively distal joints (starting with the most proximal joint) until contact is made between the associated link and the object. Note that the action of successive joint flexion is done only in the planning and is not actually executed by the robot hand. Once the final hand configuration has been attained, the hand pose and configuration in the frames prior to the grasp are interpolated between this final hand configuration and a valid hand configuration (that does not collide with the object).

The assumption taken in this technique is that the dextrous hand, when mapped to the coarse pregrasp trajectory, moves it to a reasonably near vicinity to the object at an appropriate posture. This should guarantee that, under the behaviors defined in the "reactive" planning (which is basically local), the manipulator posture should converge toward the desired volar (enveloping) grasp.

F. Generating Fingertip Precision Grasp

General Approach: The approach taken to plan the fingertip precision grasp is different than that for the volar grasp, primarily because this precision grasp does not envelope the object. As with the volar grasp, there are basically two stages:

[3]The circumflex ^ over the symbol indicates its normalized version.

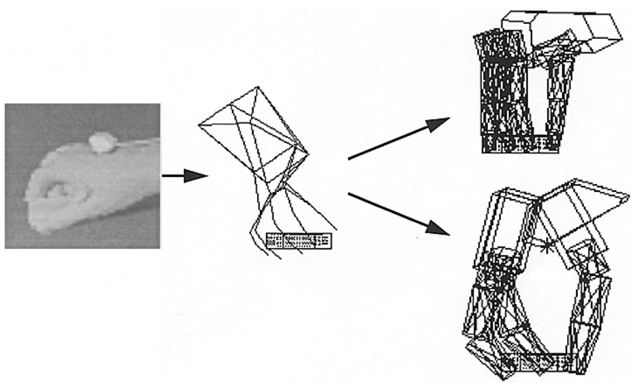


Fig. 13. Results of mapping a fingertip precision grasp for Utah/MIT hand (top-right) and Salisbury hand (bottom-right).

1) *Initial Grasp Pose and Posture Determination Phase*: Let the N contact points on the object be denoted $\mathbf{p}_{c1}, \mathbf{p}_{c2}, \ldots, \mathbf{p}_{cN}$. The initial grasp pose is determined by calculating the pose of the contact frame F_c and orienting the pivot frame F_{pivot} centered at the pivot point on the hand by a certain fixed transformation relative to F_c (Fig. 12).

Subsequently, the pose of the hand is searched at this neighhorhood until a kinematically feasible grasp is found. The contact frame F_c is centered about point $\mathbf{c}$, defined to be

$$\mathbf{c} = \frac{1}{N-1}\sum_{i=2}^{N}\frac{1}{2}(\mathbf{p}_{c1} + \mathbf{p}_{ci}) = \frac{1}{2}\left(\mathbf{p}_{c1} + \frac{1}{N-1}\sum_{i=2}^{N}\mathbf{p}_{ci}\right).$$

The orientation of F_c is determined as follows:

$$\mathbf{l} = \sum_{i=2}^{N}(\mathbf{p}_{ci} - \mathbf{p}_{c1}) = \sum_{i=2}^{N}\mathbf{p}_{ci} - (N-1)\mathbf{p}_{c1}$$
$$\hat{\mathbf{l}} = \frac{\mathbf{l}}{\|\mathbf{l}\|}.$$

If the best fit plane Π to the contact points is $\mathbf{r}\cdot\hat{\mathbf{v}} = d$, where $\mathbf{r}$ is any point on Π, and $\hat{\mathbf{v}}$ its unit normal, then

$$\hat{\mathbf{n}} = \begin{cases} \hat{\mathbf{v}}, & \text{if } (\hat{\mathbf{v}}\times\hat{\mathbf{l}})\cdot(\mathbf{p}_{cN} - \mathbf{p}_{c1}) > 0 \\ -\hat{\mathbf{v}}, & \text{otherwise} \end{cases}$$

and $\hat{\mathbf{m}} = \hat{\mathbf{n}} \times \hat{\mathbf{l}}$.

2) *Local Grasp Pose and Posture Adjustment Phase*: This is done by locally optimizing a task-oriented criterion with respect to the pose and posture of the hand. The local adjustment is done at two levels; the two levels are associated with the optimization with respect to the pose (outer level) and with respect to the joint angles (inner level). Note that the inner level optimization requires iterations only if redundancy of finger DOF's exists. For the Utah/MIT hand, each finger has a redundant DOF since there are 4 DOF's per finger with only the

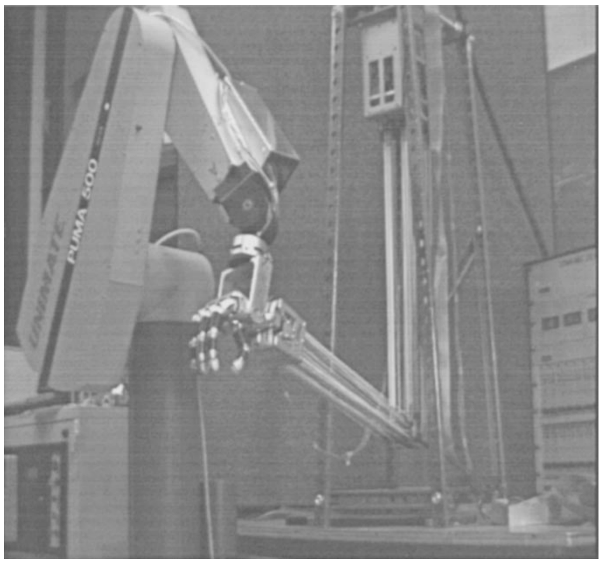


Fig. 14. PUMA 560 and Utah/MIT hand system.

3-D position of each fingertip being fixed. There is no redundancy of DOF for the three fingers of the Salisbury hand.

Let $\Delta_j = (\mathbf{t}_j, \mathbf{r}_j)$ be the vector representing the hand pose with $\mathbf{t}_j$ and $\mathbf{r}_j$ the translation and rotation vectors respectively, Θ_{jk} be the vector of joint angles representing the hand posture, and $C()$ be the task-oriented criterion to be optimized. Then the optimization can be symbolically represented as

$$\max_{\Delta_j}\left(\max_{\Theta_{jk}} C(\Delta_j, \Theta_{jk})\right) = \max_{\Delta_j} C(\Delta_j, \Theta_{j,\mathrm{opt}}).$$

The outer optimization level uses the Nelder-Mead simplex algorithm (see, for example, [22]), while the inner optimization is accomplished using Yoshikawa's scheme for optimizing manipulator postures with redundant DOF's [34].

For a given initial hand pose Δ_j, the locally optimal hand posture $\Theta_{j,\mathrm{opt}}$ is determined using the following pseudocode:

```
Repeat until either the joint angle limitations are exceeded
or the magnitude of joint angle change vector $\dot{\Theta}\delta t$ is below
a threshold {
    $\kappa_{jk} = \lambda \frac{\partial}{\partial \Theta} C(\Delta_j, \Theta)\Big|_{\Theta=\Theta_{jk}}$
    where $\lambda$ is a positive multiplicative constant (set to 0.002 here)
    $\dot{\Theta}_{jk} = J^+\dot{\mathbf{x}} + (I - J^+J)\kappa_{jk} = (I - J^+J)\kappa_{jk}$
    since $\dot{\mathbf{x}} = \mathbf{0}$, the contact points being fixed at each iteration.
    $\Theta_{jk} \leftarrow \Theta_{jk} + \dot{\Theta}_{jk}\delta t$
}
```

where J is the hand Jacobian (which is simply the linear transformation that maps the robot joint velocity into task velocity) and J^+ is the pseudoinverse of the Jacobian, i.e.,

$$J^+ = J^{\mathrm{T}}(JJ^{\mathrm{T}})^{-1}.$$

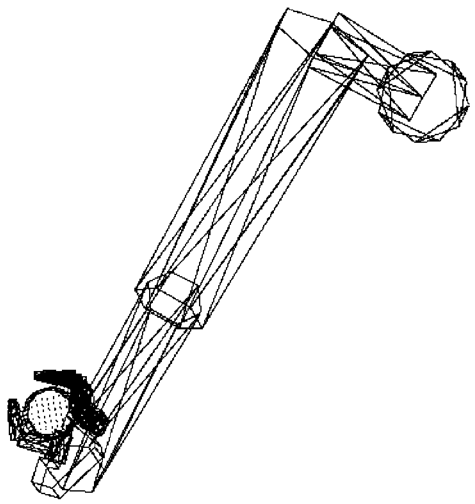


Fig. 15. A snapshot of cylindrical power grasp planning—see left image of Fig. 16.

Estimating Dextrous Hand-Object Contact Points: In order to plan the robot fingertip grasp, we need to determine the points of contact between the robot hand and the object to be grasped. This is done by following these steps:

1) Determine the rough positions of the human hand-object contact points: Fit these contact points with a plane Π: $\mathbf{p} \cdot \hat{\mathbf{n}} = d$. This is done using the observation that human fingertip grasps result in contact points that approximately lie on a plane [16].
2) Cull off planes and edges as potential grasping places by imposing the constraint that the associated object normal should not be too close in direction to $\hat{\mathbf{n}}$: At each point on edge i (edge point) and point on face j (face point), let the associated normals be $\hat{\mathbf{n}}_{e_i}$ and $\hat{\mathbf{n}}_{f_j}$, respectively. Then, disregard points on edge i if $|\hat{\mathbf{n}}_{e_i} \cdot \hat{\mathbf{n}}| > \cos\theta_e$ and points on face j if $|\hat{\mathbf{n}}_{f_j} \cdot \hat{\mathbf{n}}| > \cos\theta_f$. Since we favor faces over edges, $\theta_e > \theta_f$, with $\theta_e = 75°$ and $\theta_f = 45°$.
3) For each virtual finger with more than 1 real finger, fit a line to the human hand contact points, and space the hypothesized dextrous hand-object contact points equally along this line.
4) Find the object surface points nearest to the hypothesized dextrous hand-object points; for each face point, impose a margin β_m from the nearest edge ($\beta_m = 1.0$ cm) if possible; other wise arrange its position in the center between opposite edges.

Results of mapping a fingertip grasp to those of the Utah/MIT and Salisbury hands (subsequent to estimating dextrous hand-object contact points) are shown in Fig. 13.

The Optimization Criterion: The precision grasp may be generated using a variety of task-oriented measures such as the manipulability measure [34], task compatability index [4], sum of force minimization [20], and measures based on the

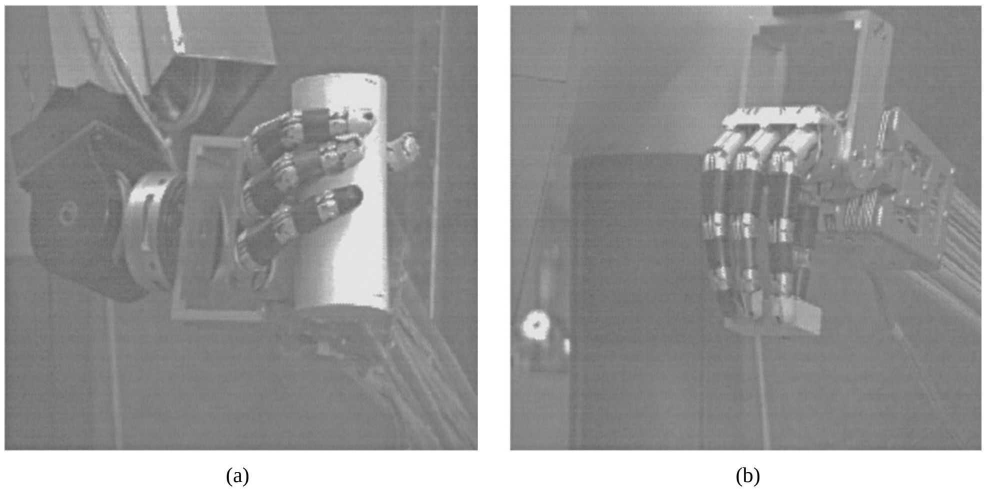


Fig. 16. Cylindrical power grasp (left) and fingertip precision grasp (right) by the Utah/MIT hand.

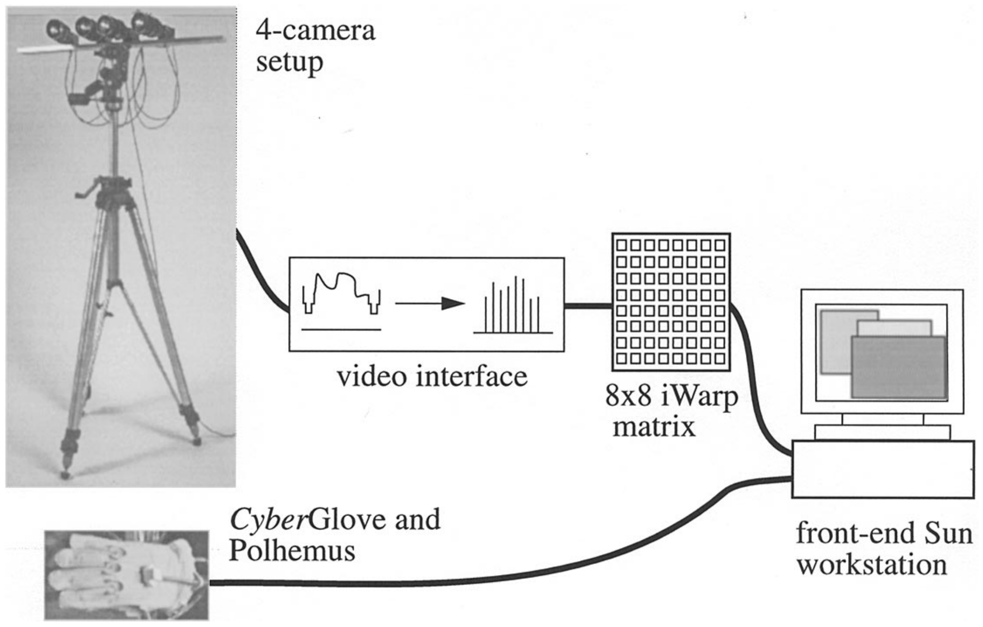


Fig. 17. Observation system.

task ellipsoid [19]. We use Chiu's task compatability index [4] to generate precision grasps.

V. System Implementation

A. Robot System

The arm/hand system used to verify the grasp mapping technique is shown in Fig. 14. The arm used is the PUMA 560 arm while the dextrous hand used is the Utah/MIT hand. Both the arm and hand are controlled via Chimera, which is a real-time operating system for sensor-based control developed at Carnegie Mellon University [30]. The joint angles of the arm and the hand are fed from the grasp simulator to the arm/hand system via Chimera.

B. Grasp Simulator

The grasp simulator was written in C and the interface in XView and PHIGS (Programmer's Hierarchical Interactive Graphics System). The robot arm and hand models were created in this environment; the object collision detection scheme described in [11] is implemented in this grasp simulator as well, for the purpose of robot grasp planning. A snapshot of the simulator is shown in Fig. 15. The two types of grasp performed by the Utah/MIT hand are shown in Fig. 16.

In general, depending on the task to be performed, a library of autonomous functions would be required. A good example is the task of turning a screw into a threaded hole; the exact motions used by a human in turning a screw may not be suitable for any specifically given manipulator. A skill library would be required—such as that described by Hasegawa, Suehiro, and Takase [8]—to execute fine manipulative actions not transferable directly by the human operator. This is still consistent with our notion of programming by human demonstration; the system basically recognizes certain actions and passes that high-level information to trigger the relevant skill library (which may include sensing strategies) to execute the task. The notion of skill library is not addressed in this paper.

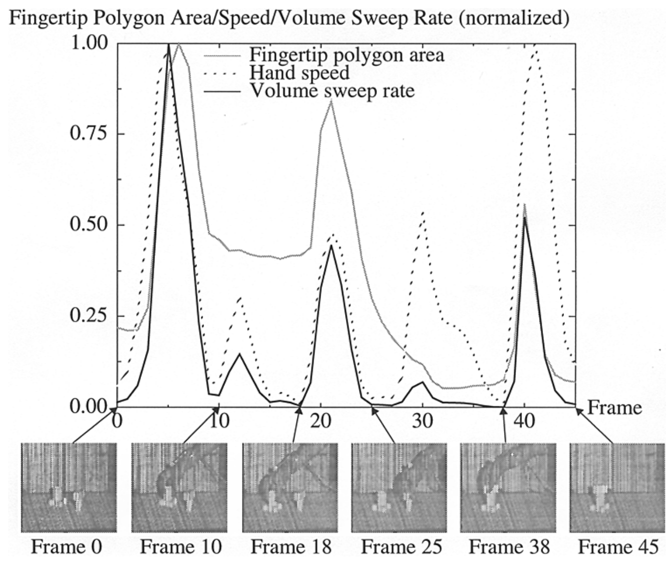


Fig. 18. Recovered task breakpoints.

Some work on detecting the high-level repetitive actions (e.g., turning the screw) using spectrogram (local Fourier) analysis on human hand motion profiles has been reported [13].

C. Typical Execution Times

The execution time for the entire system is greatly limited by the recovery of depth images from multibaseline stereo. Currently, each stereo depth image of resolution 480×512 is calculated at the rate of about 8 min per frame on an iWarp machine [2]. The total time taken to track an object is highly dependent on the complexity of the object and the number of frames to be tracked. Typical times for object tracking are between 3–6 min per frame on a SUN SparcStation 1+. On the same workstation, task segmentation typically takes about 2–10 s while grasp mapping and planning (which includes intermittent hand redisplay) for the Utah/MIT hand takes between 5–10 min.

VI. Robot Programming to Execution Example

In this section, we describe a complete cycle that starts from human demonstration of a task to the system, task analysis of observed data, and finally robot execution of the observed task. The human demonstration of the task is observed using Version 2 of the observation system shown in Fig. 17.

A. Observation System

Our observation system consists of a four-camera active[4] multibaseline system that is connected to the iWarp array via a video interface (Fig. 17). The iWarp system, by itself, is capable of reading digitized video signals from the four cameras at video rate, i.e., at 30 frames/s.

Normally, two cameras are sufficient to recover depth from triangulation. However, having a redundant number of cameras facilitates correct matches between images, which is critical to accurate depth recovery [26]. Details of the video interface are given by Webb *et al*. [32] while the depth recovery scheme is described at length by Kang *et al*. [17]. The cameras are verged so that their viewing spaces all intersect at the volume of interest, maximizing camera viewspace utility. In addition, a sinusoidally varying intensity pattern is projected onto the scene to increase the local discriminability at each image point. Results [17] have indicated that the average errors in fitting planes and cylinders to stereo range data are less than 1 mm at distances of 1.5–3.5 m away from the cameras.

[4]The word "active" refers to the addition of projected structured lighting during image capture.

TABLE I
VIRTUAL FINGER ASSIGNMENT

Grasp mapping #	Virtual Fingers	Human Hand Composition	Utah/MIT Hand Composition
1	VF_1	$\{F_1\}$	$\{F_1\}$
(Four-fingered	VF_2	$\{F_2\}$	$\{F_2\}$
disc grasp)	VF_3	$\{F_3, F_4\}$	$\{F_3, F_4\}$
2	VF_1	$\{F_1\}$	$\{F_1\}$
(Five-fingered	VF_2	$\{F_2\}$	$\{F_2\}$
prismatic grasp)	VF_3	$\{F_3, F_4, F_5\}$	$\{F_3, F_4\}$

Both the serial line outputs of the *Cyber*Glove and Polhemus device are connected to the host (front-end) Sun workstation. During image capture, the video outputs of the camera system (where all the four cameras are synchronized) are digitized, distributed, and stored in iWarp DRAM's using systolic communication. Once image capture is complete, these images are then chanelled to the front-end workstation to be stored in a storage disk for later depth recovery. We are able to achieve sampling rates of about 4.5–5 frames/s.

Depth of the scene at every frame is extracted using the algorithm described in Kang *et al*. [17]. Noise (especially at the object borders) are removed by filtering out outliers. An example of extracted stereo data extracted from a scene is shown in Fig. 19(c).

B. Segmenting Task Sequence

The task sequence has been identified by the system as a 2-task, namely a task with two manipulation phases. The result of the temporal task segmentation is shown in Fig. 18. In Fig. 18, the fingertip polygon is the polygon formed with the fingertips as its vertices and the volume sweep rate is the product of the fingertip polygon area and hand speed. Task segmentation is based on the fact that the hand motion profiles (fingertip polygon area, hand speed, and volume sweep rate) normally assume a bell curve [12]. The task involved picking and placing a receptacle, followed by picking up a peg and inserting it into the hole of the receptacle.

C. Identifying the Grasped Object

The initial gross positions of the objects were manually determined by the user; the three–dimensional template matching (3DTM) program subsequently refines these poses [33]. The results of object localization are shown in Fig. 19.

D. Tracking 3-D Object

By using the *Cyber*Glove and Polhemus data and the results of temporal task segmentation, the object that is being

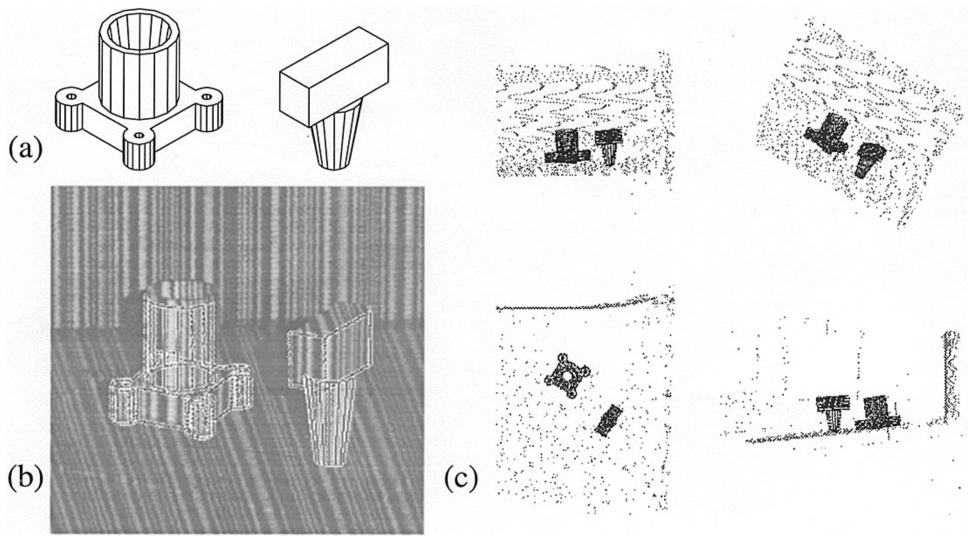


Fig. 19. Object model fitting. (a) Geometric models of objects (receptacle, at left, and peg, at right). (b) Superimposed object models in a scene. (c) Corresponding 3-D views.

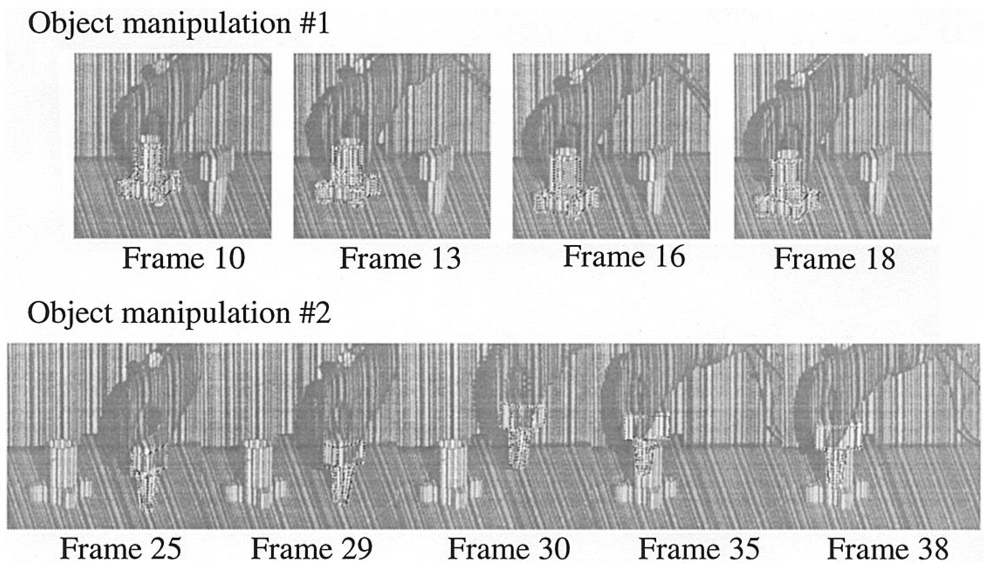


Fig. 20. Object tracking results (selected frames shown). Object manipulation #1 involves translating the receptacle closer to the bottom left hand side of the image; object manipulation #2 involves picking the peg and inserting it into the receptacle. Note that significant object motion may occur between frames, e.g., between frames 29 and 30.

moved during each manipulation phase can be automatically identified. The object that is being moved is identified based on the proximity of the fingertip polygon centroid of the human hand to the objects in the scene during the grasp phase. The smallest distance of each object to the fingertip polygon centroid is first calculated. The object whose closest distance is minimum is deemed to be the grasped object. Object motion during the subsequent manipulation phase affects only the identified object.

To track the identified object, the 3DTM algorithm [33] is again used to finely localize its pose at each frame. While 3DTM is relatively robust to a certain degree of object occlusion, it fails under significant object occlusion (such as in Frames 35 and 38 in Fig. 20). This problem is avoided by masking out range pixels of other known environmental objects and by imposing geometric constraints during pose estimation to avoid object interpenetration. The results of object tracking are shown in Fig. 20.

E. Human Grasp Recognition

Once the object trajectory has been extracted, the two human grasps (the first with the receptacle and the second with the peg) have to be recognized. Prior to that, however, the hand poses have to be adjusted to account for sensor inaccuracies. Hand readjustment is accomplished through local search within a neighborhood of hand orientations that minimizes geometric violations between the hand and the grasped object [13]. The results of hand adjustment are shown in Fig. 21 (first grasp) and Fig. 22 (second grasp).

The first grasp (with the receptacle) has been recognized as a "four-fingered disc grasp." Note that the extracted measure-

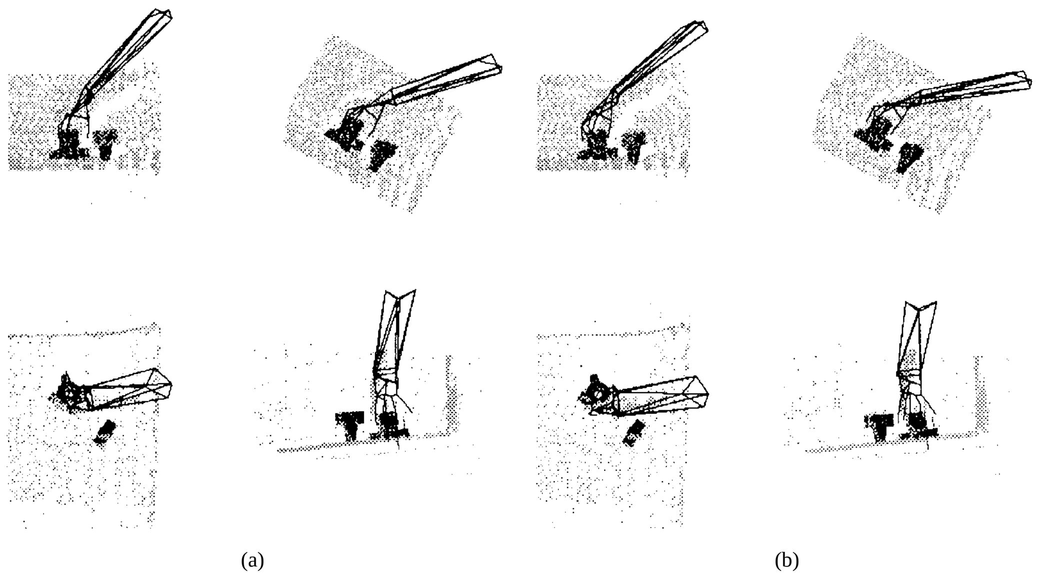


Fig. 21. Hand pose (a) before and (b) after adjustment (first grasp). The hand is grasping the receptacle, corresponding to frame 10 in Fig. 20.

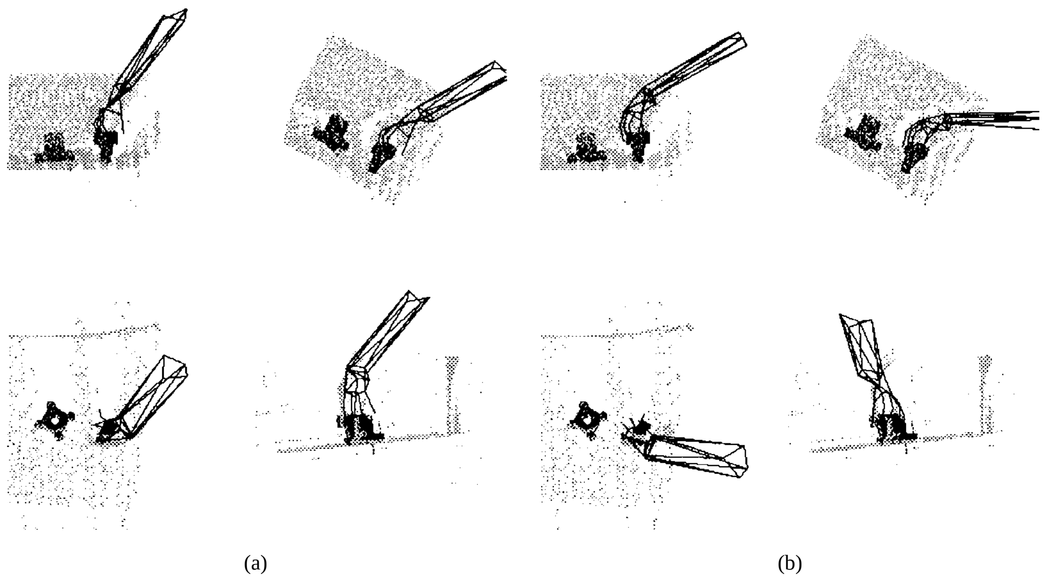


Fig. 22. Hand pose (a) before and (b) after adjustment (second grasp). The hand is grasping the peg, corresponding to frame 25 in Fig. 20.

ment of the last finger resulted in the analysis that it did not "touch" the object, when it actually did during the task. This is a problem due to both imperfect glove calibration and hand model. The second grasp (with the peg) has been recognized as a "five-fingered prismatic grasp."

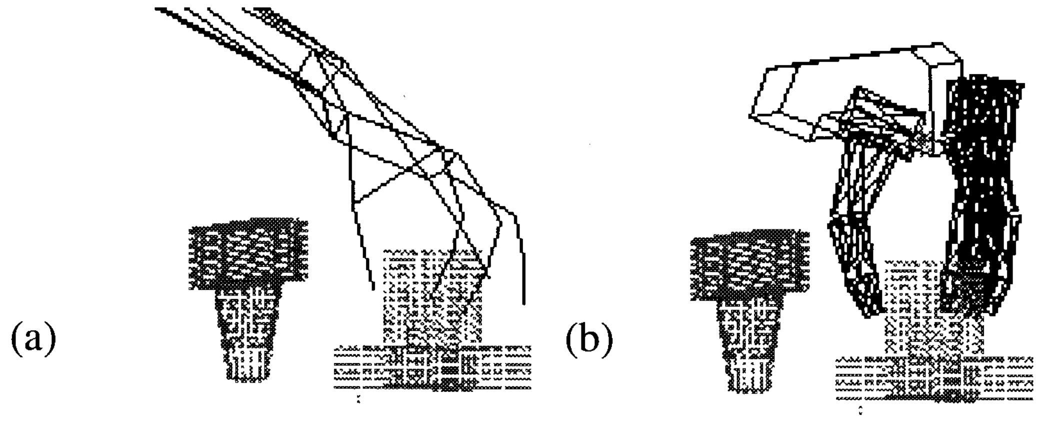
Subsequent to grasp recognition, the human grasps are mapped to manipulator grasps as shown in Figs. 23 and 24. The assignment of real fingers to virtual fingers in both human and robot hands are shown in Table I. (VF_j is the jth virtual finger while F_k is the kth real finger.)

F. Robot Execution

Once the grasp mapping has been performed, the robot system comprising the PUMA arm and Utah/MIT hand is used to perform the task as shown in Fig. 25. The robot hand trajectory was planned such that object motion during human demonstration is replicated. As described in Section VI-D, the object trajectories were obtained by tracking the grasped object at every frame from stereo range data. Full details of each operation in the programming to robot execution cycle are given in Kang [11].

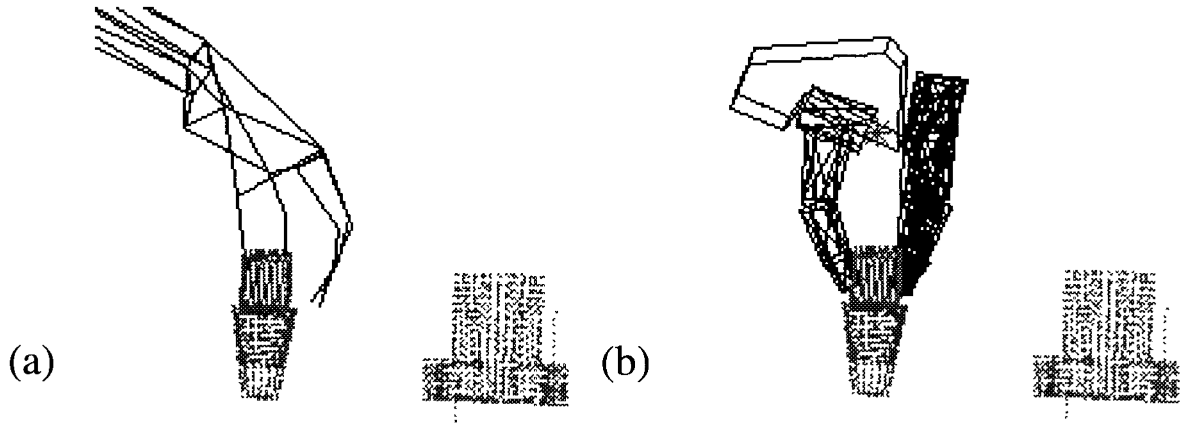
G. Summary: System Flow of Information

This section briefly reviews the types of information used by our system for the process of robot programming by human demonstration to work. As shown in Fig. 26, there are three types of information used in the system: *raw input data* that are sampled measurements taken during human task demonstration, *user given information* that is necessary *a priori* information used by the system, and *system generated*



(a) (b)

Fig. 23. Result of grasp mapping #1: (a) Human grasp, and (b) Utah/MIT grasp.



(a) (b)

Fig. 24. Result of grasp mapping #2: (a) Human grasp, and (b) Utah/MIT grasp.

information that is internal knowledge (generated based on the first two types of information) used to enable robot execution of the observed task.

For our system, the raw input data are:

1) Stereo images (from which depth images are recovered, Section VI-A [17]), and
2) *Cyber*Glove and Polhemus data (used to recover task breakpoints, Section VI-B [12]).

The user defined information comprises:

1) 3-D object models (used to track objects, Section VI-D [11]),
2) Human hand model and grasp taxonomy (used to recognize grasp and extract grasp abstraction hierarchy, Section III-B [15], [16]), and
3) Robot arm and hand models, group assignments of robot fingers, and Chiu's task compatability index (used in conjunction with the system generated information of object trajectory and grasp abstraction hierarchy to plan the robot grasp and trajectory, Section IV).

VII. Discussion

Robot programming by human demonstration as described in this paper constitutes an attractive alternative means of teaching a robot to perform repetitive assembly tasks. This is especially true if the assembly tasks involve many different small to medium batch jobs that may require programming of dextrous manipulators. Programming can be done easily and quickly, since it is reduced to merely demonstrating the task. This work is a first step in showing that such a method works.

While mostly positional and geometric information is used in our work, the central idea of robot programming by demonstration can be extended to involve other modes of sensing, such as force sensing. Such a mode of sensing would obviously be necessary in tasks that involve objects with tight tolerances and compliant force/motion strategy. Equally important are skill libraries and the ability to replan (the latter in case of initial failure). These additional features are necessary if this approach of robot programming is to be made practical and viable.

One salient characteristic of this approach is that task planning is done based on the observation of human execution and is hence *human-centric*. The optimality of the resulting plan cannot be easily verified. A problem with this approach of robot programming is that very specialized grippers (such as suction cups for lifting flat pieces of glass and miniature end-effectors on a turret head) cannot be easily programmed with this approach. Also, programming by human demonstration

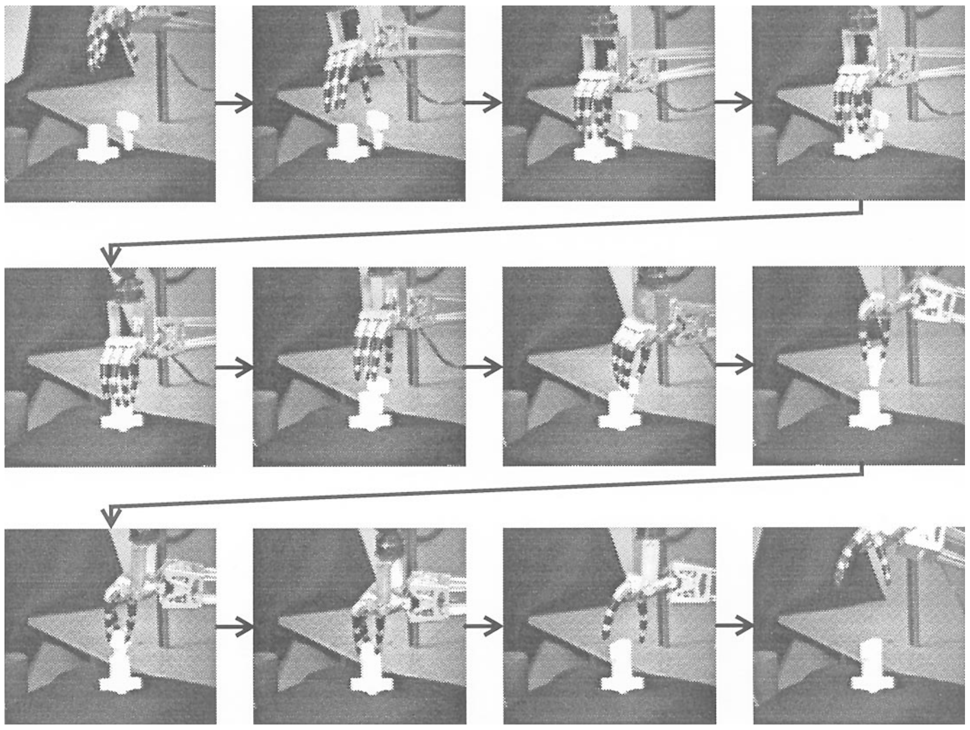
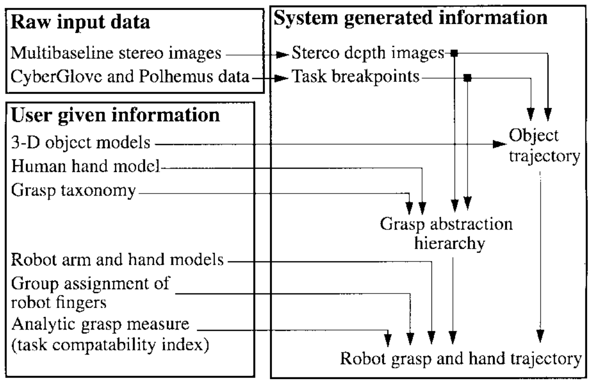


Fig. 25. Different temporal snapshots of task execution by PUMA arm and Utah/MIT hand system.


Raw input data
Multibaseline stereo images
CyberGlove and Polhemus data
System generated information
Stereo depth images
Task breakpoints
User given information
3-D object models
Human hand model
Grasp taxonomy
Object trajectory
Grasp abstraction hierarchy
Robot arm and hand models
Group assignment of robot fingers
Analytic grasp measure (task compatability index)
Robot grasp and hand trajectory
Fig. 26. Information flow of our robot programming by human demonstration system. Each line points to the product with the requisite input on the other end of the line.

may not be suitable for tasks that require high precision and high volume, where optimality, in terms of speed and power consumption, is critical.

VIII. Summary

In this paper, we have described our method of mapping the observed human grasp to that of the manipulator. The mapping is performed at two consecutive levels—the functional level and the physical level. In the functional level mapping, functionally equivalent groups of fingers are mapped. Once this is accomplished, at the physical level of mapping, the location and pose of the robot hand is refined to accommodate the different hand geometries in grasping the object.

The approach of robot programming by human demonstration greatly simplifies grasp planning. It illustrates our philosophy of easing the programming burden to mostly demonstrating the task; it reduces the complexity of programming and planning the task by taking advantage of the cues derived from observing a human.

Acknowledgment

The authors would like to thank R. Voyles for his help with the PUMA arm and Utah/MIT hand system.

References

[1] M. A. Arbib, T. Iberall, and D. M. Lyons, "Coordinated control programs for movements of the hand," in *Hand Function and the Neocortex*, A. W. Goodwin and T. Darian-Smith, Eds. Berlin, Germany: Springer-Verlag, 1985, pp. 111–129.
[2] S. Borkar *et al.*, "Supporting systolic and memory communication in iWarp," in *Proc. 17th Int. Symp. Computer Architecture*, Seattle, WA, 1990, pp. 70–81.
[3] D. L. Brock and J. K. Salisbury, "Implementation of behavioral control on a robot hand/arm system," in *Proc. 2nd Int. Symp. Experimental Robotics*, 1991, pp. 1–19.
[4] S. L. Chiu, "Task compatibility of manipulator postures," *Int. J. Robot. Res.*, vol. 7, no. 5, pp. 13–21, 1988.
[5] M. R. Cutkosky and R. D. Howe, "Human grasp choice and robotic grasp analysis," in *Dextrous Robot Hands*, S. T. Venkataraman and T. Iberall, Eds. Berlin, Germany: Springer-Verlag, 1990, pp. 5–31.
[6] E. Grosso and G. Vercelli, "Grasping strategies for reconstructed unknown 3D objects," in *Proc. IEEE/RSJ Workshop Intelligent Robots and Systems*, Osaka, Japan, 1991, pp. 70–75.
[7] T. Hamada, K. Kamejima, and I. Takeuchi, "Image based operation: A human-robot interaction architecture for intelligent manufacturing," in *Proc. 15th Conf. IEEE Industrial Electron. Soc.*, 1989, pp. 556–561.
[8] T. Hasegawa, T. Suehiro, and K. Takase, "A model-based manipulation system with skill-based execution," *IEEE Trans. Robot. Automat.*, vol. 8, pp. 535–544, Oct. 1992.
[9] K. Ikeuchi and T. Suehiro, "Toward an assembly plan from observation, Part I: Task recognition with polyhedral objects," *IEEE Trans. Robot. Automat.*, vol. 10, pp. 368–385, June 1994.
[10] S. C. Jacobson, J. E. Wood, D. F. Knutti, and K. B. Biggers, "The Utah/MIT dextrous hand: Work in progress," *Int. J. Robot. Res.*, vol. 3, no. 4, pp. 21–50, 1984.

[11] S. B. Kang, “Robot instruction from human demonstration,” Ph.D. dissertation, The Robotics Institute, Carnegie Mellon Univ., Pittsburgh, PA, Dec. 1994 (also as CMU-RI-TR-94-44).

[12] S. B. Kang and K. Ikeuchi, “Toward automatic robot instruciton from perception—Temporal segmentation of tasks from human hand motion,” in *IEEE Robot. Automat.*, vol. 11, pp. 670–681, 1995.

[13] ——, “Grasp recognition and manipulative motion characterization from human hand sequences,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, San Diego, CA, May 1994, pp. 1759–1764.

[14] ——, “Toward automatic robot instruction from perception — Recognizing a grasp from observation,” *IEEE Trans. Robot. Automat.*, vol. 9, pp. 432–443, Aug. 1993.

[15] ——, “A grasp abstraction hierarchy for recognition of grasping tasks from observation,” in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Yokohama, Japan, July 1993, pp. 621–628.

[16] ——, “A framework for recognizing grasps,” Tech. Rep. CMU-RI-TR-91-24, Carnegie Mellon Univ., Pittsburgh, PA, Nov. 1991.

[17] S. B. Kang, J. A. Webb, C. L. Zitnick, and T. Kanade, “A multibaseline stereo system with active illumination and real-time image acquisition,” *Int. Conf. Computer Vision*, Cambridge, MA, June 20–23, 1995, pp. 88–93.

[18] T. Kuniyoshi, M. Inaba, and H. Inoue, “Teaching by showing: Generating robot programs by visual observation of human performance,” in *Proc. 20th Int. Symp. Industrial Robots*, 1989, pp. 119–126.

[19] Z. Li and S. Sastry, “Task oriented optimal grasping by multifingered robot hands,” in *Proc. IEEE Int. Conf. Robotics and Automation*, 1987, pp. 389–394.

[20] X. Markenskoff and C. H. Papadimitriou, “Optimum grip of a polygon,” *Int. J. Robot. Res.*, vol. 8, no. 2, pp. 17–29, 1989.

[21] M. T. Mason and J. K. Salisbury, *Robot Hands and the Mechanics of Manipulation*. Cambridge, MA: MIT Press, 1985.

[22] J. H. Mathews, *Numerical Methods for Computer Science, Engineering, and Mathematics*. Englewood Cliffs, NJ: Prentice-Hall, 1987.

[23] J. Napier, “The prehensile movements of the human hand,” *J. Bone and Joint Surgery*, vol. 38B, no. 4, pp. 902–913, Nov. 1956.

[24] V. Nguyen, “Constructing stable grasps in 3-D,” in *Proc. IEEE Int. Conf. Robotics and Automation*, 1987, pp. 234–239.

[25] T. Okada, “Object-handling system for manual industry,” *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, pp. 79–89, 1979.

[26] M. Okutomi and T. Kanade, “A multiple-baseline stereo,” in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 1991, pp. 63–69.

[27] N. Pollard, “Planning grasps for a robot hand in the presence of obstacles,” in *Proc. Int. Conf. Robotics and Automation*, 1993, pp. 723–728.

[28] K. Rao, G. Medioni, H. Liu, and G. A. Bekey, “Robot hand-eye coordination: Shape description and grasping,” in *Proc. Int. Conf. Robotics and Automation*, 1988, pp. 407–411.

[29] S. A. Stansfield, “Robotic grasping of unknown objects: A knowledge-based approach,” *Int. J. Robot. Res.*, vol. 10, no. 4, pp. 314–326, 1991.

[30] D. B. Stewart and P. K. Khosla, *Chimera 3.1, The Real-Time Programming Operating System for Reconfigurable Sensor-Based Control Systems*, Carnegie Mellon Univ., Pittsburgh, PA, 1993.

[31] T. Takahashi and H. Ogata, “Robotic assembly operation based on task-level teaching in virtual reality,” in *Proc. IEEE Int. Conf. Robotics and Automation*, 1992, pp. 1083–1088.

[32] J. A. Webb, T. Warfel, and S. B. Kang, “A scalable video rate camera interface,” Tech. Rep. CMU-CS-94-192, Carnegie Mellon Univ., Pittsburgh, PA, 1994.

[33] M. D. Wheeler and K. Ikeuchi, “Sensor modeling, probabilistic hypothesis generation, and robust localization for object recognition,” in *Proc. 2nd CAD-Based Vision Workshop*, 1994, pp. 46–53.

[34] J. D. Wolter, R. A. Volz, and A. C. Woo, “Automatic generation of gripping positions for assembly,” in *Robot Grippers*, D. T. Pham and W. B. Heginbotham, Eds. Berlin, Germany: Springer-Verlag,1986, pp. 55–74.

[35] T. Yoshikawa, “Manipulability of robotic mechanisms,” *Int. J. Robot. Res.*, vol. 4, no. 2, pp. 3–9, 1985.

Sing Bing Kang received the B.Eng. and M.Eng. degrees in electrical engineering from National University of Singapore in 1987 and 1990, respectively, and the M.Sc. and Ph.D. degrees in robotics from Carnegie Mellon University, Pittsburgh, PA, in 1992 and 1994, respectively. His doctoral work was in enabling robot systems to observe, recognize, and replicate grasping tasks.

He is currently a member of research staff Cambridge Research Lab., Digital Equipment Corp., Cambridge, MA, where he is working on scene reconstruction and modeling from image sequences. His research interests include 3-D computer vision, human hand motion and grasp analysis, and automatic task learning.

Dr. Kang’s paper on the complex extended Gaussian image (coauthored with K. Ikeuchi) won the IEEE Computer Society Outstanding Paper award at the 1991 IEEE International Conference on Computer Vision and Pattern Recognition (CVPR’91). He also served on the program committee of CVPR’96.

Katsushi Ikeuchi (SM’95) received the B.Eng. degree in mechanical engineering from Kyoto University, Kyoto, Japan, in 1973, and the Ph.D. degree in information engineering from the University of Tokyo, Tokyo, Japan, in 1978.

After working at the AI Laboratory, Massachusetts Institute of Technology, Cambridge, and the ETL of MITI, Japan, he he joined the School of Computer Science, Carnegie Mellon University in 1986, where he is now Principal Research Scientist (Research Full Professor), Department of Computer Science and Robotics Institute. He has developed the “smoothness constraint,” a constraint to force neighboring points to have similar surface orientations, by which he iteratively recovered shape from shading and shape from texture. He pioneered the use of specular reflections to recover surface orientations. Instead of discarding specular reflections, he actively used them for recovering shape and reflectance. Recently, this method evolved into Photometric Sampling, which determines not only the object shape but also surface characteristics. He has also been working to develop object representations for vision-guided manipulation for such tasks as bin-picking of man-made objects and sampling of natural objects for planetary rovers or clean-up of nuclear accident sites. The representations he has proposed include the Extended Gaussian Image (EGI), the Complex Extended Gaussian Image (CEGI), the Spherical Angle Image (SAI), and a frame-based geometric/sensor modeling system (VANTAGE). Currently, he has been developing vision algorithm compilers, which automatically convert object and sensor models into vision programs.

Dr. Ikeuchi was the Program Chairman for the 1993 IEEE CAD-Based Vision Workshop and the General Co-Chairman of the 1995 Intelligent Robotics and Systems (IROS) conference and the Program Co-Chairman of the 1996 International conference on Computer Vision and Pattern Recognition (CVPR). He served on the program committees of several international conferences. He is on the editorial board of the IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, the *International Journal of Computer Vision*, the *Journal of Manufacturing Systems*, and the Optical Society of America. He has received several awards, including the David Marr Prize in computational vision, and an IEEE Outstanding Paper award. In addition, in 1992, his paper, “Numerical Shape from Shading and Occluding Boundaries,” was selected as one of the most influential papers to have appeared in *Artificial Intelligence* within the past 10 years.