A Robot System that Observes and Replicates Grasping Tasks

Sing Bing Kang
Digital Equipment Corp.
Cambridge Research Lab.
One Kendall Square, Bldg 700
Cambridge, MA 02139
sbk@crl.dec.com

Katsushi Ikeuchi
Computer Science Dept.
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
ki@cs.cmu.edu

Abstract

To alleviate the problem of overwhelming complexity in grasp synthesis and path planning associated with robot task planning, we adopt the approach of teaching the robot by demonstrating in front of it. The system has four components: the observation system, the grasping task recognition module, the task translator, and the robot system. The observation system comprises an active multibaseline stereo system and a dataglove. The data stream recorded is then used to track object motion; this paper illustrates how complimentary sensory data can be used for this purpose. The data stream is also interpreted by the grasping task recognition module, which produces higher levels of abstraction to describe both the motion and actions taken in the task. The resulting information are provided to the task translator which creates commands for the robot system to replicate the observed task. In this paper, we describe how these components work, with special emphasis on the observation system. The robot system that we use to perform the grasping tasks comprises the PUMA 560 arm and the Utah/MIT hand.

Keywords: Robot task programming by human demonstration, grasp recognition, multisensor 3-D object tracking.

1 Introduction

Task automation requires some form of robot programming. The current methods for robot programming include teaching (e.g., [2][16]), textual programming (e.g., [5][6]), and automatic programming (e.g., [15][17]). The first two 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. On the other hand, automatic programming conceptually requires only the object description and the high-level task specifications in order to generate the control command sequences to the robot system. 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 task programming is the *Assembly Plan from Observation* (APO) paradigm proposed by Ikeuchi and Suehiro [7]. 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, understand 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.* [15] 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 [22] 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.

1.1 Automatic robot instruction via Assembly Plan from Observation

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. Details of the system have been given by Kang and Ikeuchi [12]. We provide only a summary here.

The observation 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 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 recognition module is subsequently provided to the task translator which in turn creates commands for the robot system to replicate the observed task. Fig. 2 illustrates succinctly how our programming-by-demonstration system works. It analyzes the stream of perceptual data associated with the task by recognizing the human grasp before planning and executing the manipulator grasp.

0-8186-7042-8/95 $4.00

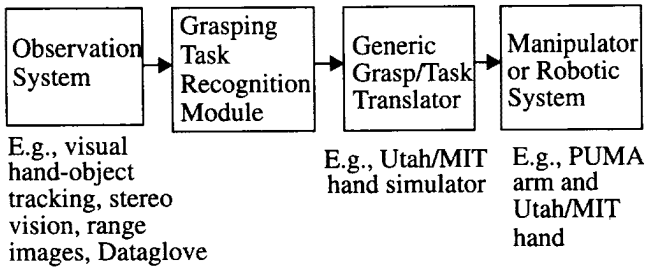

***Fig. 1* Programming-by-human-demonstration system**

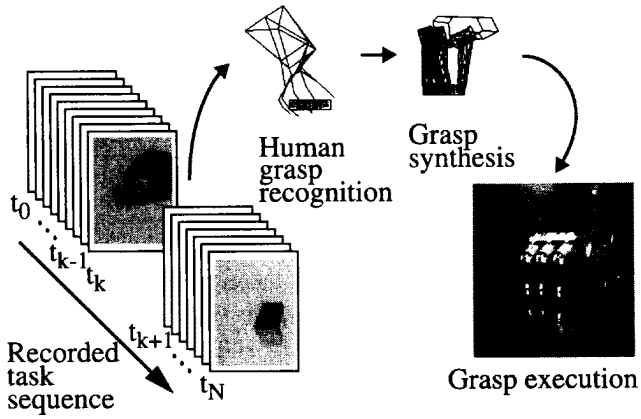

***Fig. 2* Operations in our programming-by-human-demonstration approach**

We first describe our observation system.

2 Observation system

The observation data of the task performance is captured using an active[1] multibaseline stereo system capable of rapid image capture, and a CyberGlove with a Polhemus device. The observation system is shown in Fig. 3.

We track the configuration (joint angles) and pose (position and orientation) of the hand using the *Cyber*Glove and Polhemus device respectively. The *Cyber*Glove measures 18 hand joint angles. The Polhemus 3Space Isotrak sensing device is attached to the back of the hand, and measures the position and orientation of the hand relative to the Isotrak source.

The four-camera active multibaseline system is connected to the iWarp[2] array via a video interface. On its own, the iWarp system is capable of reading digitized video signals from the four cameras at video rate, i.e., at 30 Hz.

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

2. The iWarp is a high-speed system architecture which is a result of joint effort between Carnegie Mellon University and Intel Corporation [4].

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 [20]. Details of the video interface are given by Webb *et al.* [23], while the depth recovery scheme is described at length by Kang *et al.* [14]. 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 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 (typical range) away from the cameras [14].

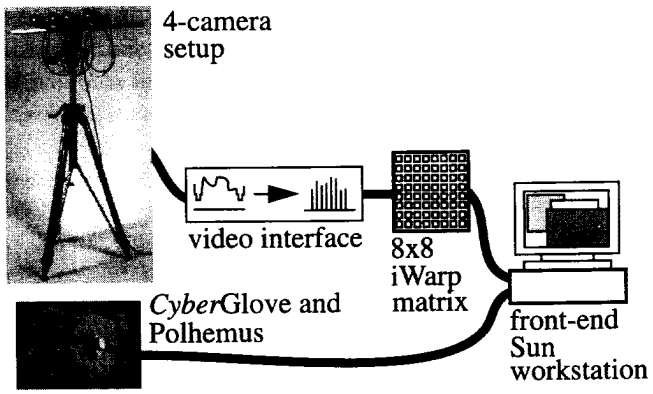

***Fig. 3* Observation system**

With this system, we are able to achieve sampling rates of about 4.5-5 Hz (stereo images, and *Cyber*Glove and Polhemus data).

3 Grasping task recognition module

The first process that the stream of perceptual data undergoes in this module is its division into separate and meaningful segments for individual analysis.

3.1 Temporal segmentation of a task

The temporal segmentation of a given task sequence divides the sequence into meaningful parts, namely reaching for the object, grasping the object, and manipulating the object (respectively the pregrasp, grasp and manipulation phases).

3.2 Features for segmentation of a task sequence

Numerous studies on human hand movement point to commonly established characterizations of the pregrasp phase. The pregrasp phase has been analyzed in terms of two simultaneous activities, namely the hand reaching activity (termed the *hand transportation* component), and the finger activity in anticipation of the grasp (termed the *hand preshape* component) (e.g., [9]). Typical profiles of the hand transportation

speed and grip aperture[3] during the pregrasp phase assume the characteristic bell-shaped curve ([8][19]).

Lyons [18], in describing a conceptual high-level control mechanism for a dextrous hand, defines the *approach area* as the surface formed by joining the fingertips of the preshaped hand by straight lines. We generalize this concept to all the phases in the grasping task, and call it the *fingertip polygon.*

3.3 Temporal segmentation of a task sequence

We can segment the entire task into meaningful subparts by analyzing both the fingertip polygon (hand preshape) area and the speed of the hand. As indicated by research on human hand movement, the hand speed and fingertip polygon area assumes the characteristic bell curve during the pregrasp phase (Fig. 4(a)). During the manipulation phase, assuming the same grasp used, the fingertip polygon area usually remains about constant while the speed varies.

It turns out that a more reliable profile to analyze is the profile of the product of the speed and fingertip polygon area at each frame which we called the *volume sweep rate* profile [11]. It measures the motion of the fingertip polygon area due to hand speed (Fig. 4(b)). While the hand reaches to grasp the object, both the speed and fingertip polygon area profiles are bell-shaped; this results in its volume sweep rate having an accentuated peak (Fig. 4(a)).

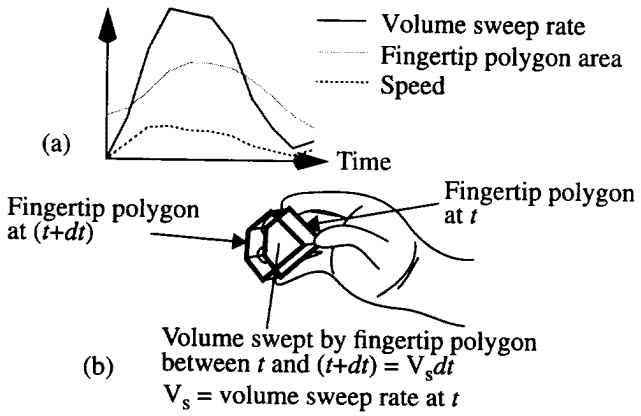

Fig. 4 **(a) Typical hand motion profiles; (b) volume sweep rate**

The algorithm to segment a task sequence into the pregrasp, grasp, and manipulation phases starts with a list of breakpoints comprising local minima in the speed profile. The initial breakpoints are extracted from the speed profile. The global segmentation procedure essentially finds the set of breakpoints that yields the minimum gaussian curve fit error in the hypothesized pregrasp phases [11]. It is assumed that the pregrasp, grasp, and manipulation phases follow one another.

3.3.1 Experiment results

The temporal task segmentation algorithm has been successfully applied to a number of typical task sequences, and the results of a sequence can be seen in Fig. 5. The task in Fig. 5 involved picking and placing a receptacle, followed by picking up a peg and inserting it into the hole of the receptacle.

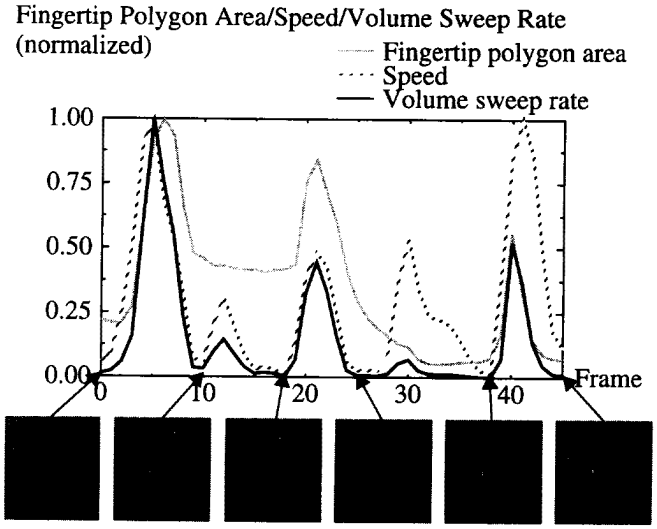

Fig. 5 **Recovered breakpoints for task involving peg and receptacle**

After temporally segmenting the task sequence, we need to recover object motion, following which the human grasp is identified. The results of grasp recognition (Section 3.5) are used in planning the manipulator grasp.

3.4 Object motion extraction

Since there may be more than one object manipulation in the task sequence, additional processing has to be performed to identify which object has been manipulated at each manipulation phase.

3.4.1 Identifying the grasped object

The object models are created using Vantage, a geometric modeler developed at Carnegie Mellon University [3]. The objects in the scene are first approximately localized by the user using an interface that features both 2-D images and 3-D scene points as shown in Fig. 6(b) and (c). Once this is done, a 3-D template matching program called 3DTM [24] is used to automatically finetune the localization of the object poses. The results for the task involving the peg and receptacle are shown in Fig. 6.

The object being manipulated is identified as the one nearest to the human fingertip polygon during the grasp phase.

3. The grip aperture in this context is defined as the separation between the tips of the thumb and index finger.

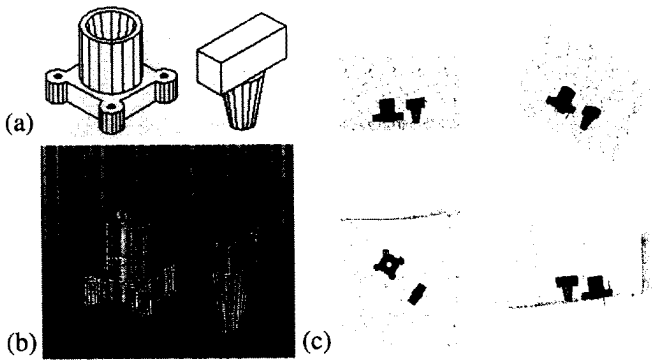

Fig. 6 **Object model fitting: (a) Geometric models of objects, (b) superimposed object models in a scene, and (c) four different 3-D views**

3.4.2 Tracking 3-D Object

Once each object has been identified as the object being moved in each manipulation phase, it is then tracked. For this, we employ the 3DTM pose refinement program [24].

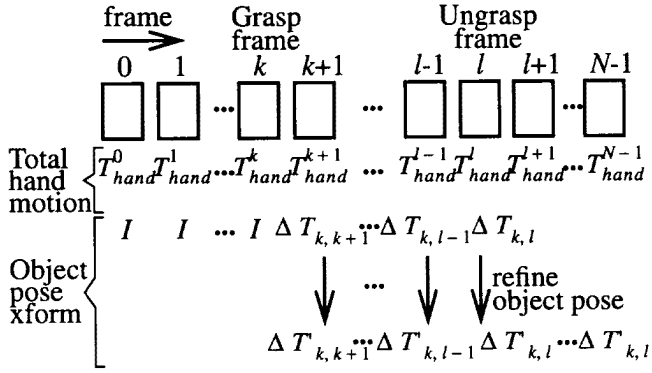

Fig. 7 **Determining pose of object**

Using *Cyber*Glove and Polhemus Data as Approximation

Because we require reasonable starting poses as input to the 3DTM program, we use the recorded *Cyber*Glove and Polhemus data as an approximation. The 3-D object tracking scheme (for one object manipulation for clarity) is depicted in Fig. 7. T^k_{hand} is the transform associated with the pose of the hand at the kth frame, and

$$\Delta T_{k,k+j} = T^{k+j}_{hand} (T^k_{hand})^{-1}$$

is the estimated object pose change between frames k and $(k+j)$. $\Delta T_{k,k+j}$ is the associated refined object pose change after applying the 3DTM program.

Direct tracking in this manner works if there is little or no object occlusion. Additional processing has to be added to tackle the problem of significant occlusion between objects, as in the case of the task involving the peg and receptacle.

Handling Significant Occlusion Between Known Objects

While the 3DTM program is robust to some degree of object occlusion, significant occlusion may occur in the course of the task performance, as can be seen in Fig. 8. In this example, the task is to pick and insert a peg into a receptacle. In cases such as this, the program fails. The following steps were employed to deal with this problem:

1. *Masking out range pixels corresponding to stationary objects of known poses*
2. *Detecting and avoiding object collision*

   We use a simple collision detection and avoidance scheme to both aid in localization and to ensure that objects do not interpenetrate at any time during task execution. Each object is represented by surface points of a predetermined spacing (of about 2.5 mm apart) [24]. Corners and edges are definitely sampled. Associated with each surface points is the object normal at that point. The normal at an edge or a corner is calculated to be the mean of the normals at adjacent faces. With these information, we can determine to a reasonably good approximation, if collision occurs, and where.

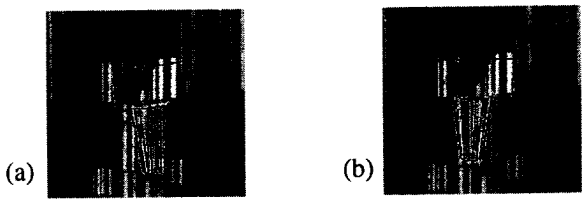

Fig. 8 **Result of detecting and avoiding object collision in pose localization: (a) without collision detection; (b) with collision detection.**

Object manipulation #1

Object manipulation #2

Fig. 9 **Object tracking results (selected frames shown). Note that significant object motion may occur between frames, e.g., between frames 29 and 30.**

The object collision avoidance algorithm is as follows: When the object being moved ("non-stationary object") and a stationary object interpenetrates at some area, the

points at these areas "pushes" non-stationary object away in the direction of the normals at the affected points of the stationary object. The repelling motion is the net twist acting on the centroid of the non-stationary object.

The object collision detection and avoidance scheme has been incorporated into the iterative pose localization of 3DTM. A result is shown in Fig. 8. The object tracking result for the task involving the peg and receptacle is shown in Fig. 9.

3.5 Grasp identification

Once the input perceptual data has been performed, the system then proceeds to identify the human grasp using in the task. We require a grasp taxonomy which could provide a systematic way of identifying grasps based on the hand configuration and object shape. We propose a grasp taxonomy based on the analysis of the effective contact points of the hand with the grasped object. The resultant spatial pattern of contact points forms what we call a *contact web*.

3.5.1 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. The shape and cardinality of the contact web yield important information about the type of grasp effected.

3.5.2 Grasp classification and recognition

The main classification of grasps using the contact web is shown in Fig. 10. Grasps are initially dichotomized into volar and non-volar grasps according to whether there is contact between the palm and object. Non-volar grasps are further categorized as either fingertip grasps (in which only the fingertips are involved) or composite non-volar grasps (in which fingertips and more proximal finger segments are involved). To aid in volar grasp recognition, we use the notion of the *virtual finger*.

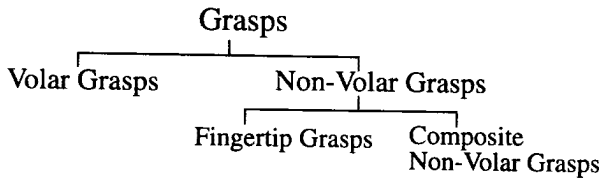

***Fig. 10* Grasp classification for recognition**

3.5.3 The virtual finger

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.

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.

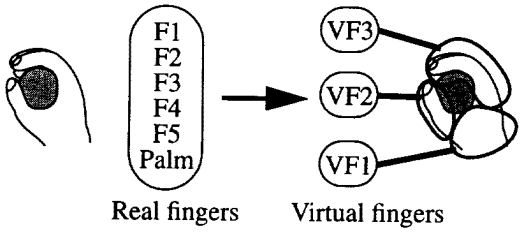

***Fig. 11* Mapping real fingers to virtual fingers**

Mapping the real fingers to virtual fingers has been detailed in Kang and Ikeuchi [12]. This mapping results in an index called 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. (For a detailed description of grasp classification, refer to Kang and Ikeuchi [12].)

3.5.4 The grasp abstraction hierarchy

We show two examples of grasp recognition from real data measured from two different tasks employing different kinds of grasps. Fig. 12 shows a precision grasp being recognized while Fig. 13 shows a type 2 "coal-hammer" grasp being recognized.

Once the grasp has been identified, the grasp abstraction hierarchy can be constructed as depicted in Fig. 14 [13]. 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 to plan the manipulator grasp.

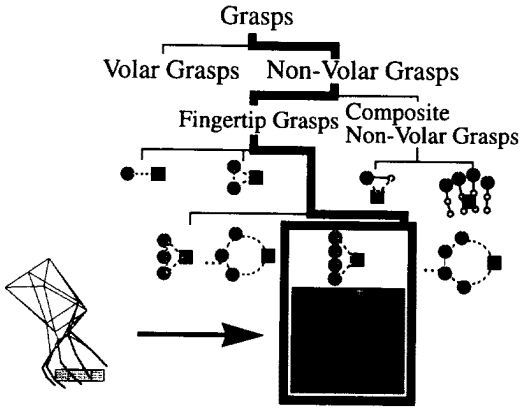

***Fig. 12* Example of precision grasp recognition**

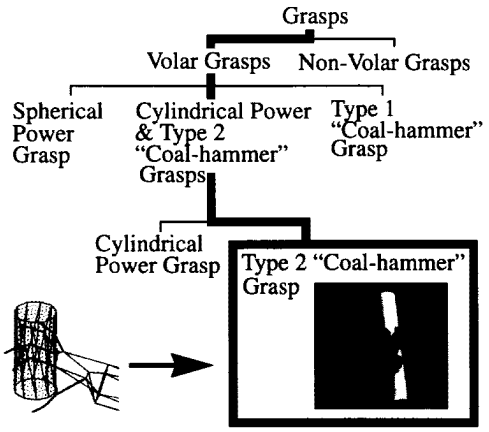

Fig. 13 **Example of power grasp recognition**

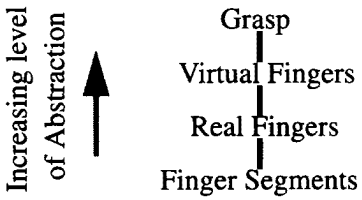

Fig. 14 **Grasp abstraction hierarchy**

4 Task translator and grasp mapping approach

The responsibility of the task translator is to use the descriptions of the task produced by the task recognition module to plan the manipulator trajectory and grasp. The task descriptions include the grasp abstraction hierarchy and the approximate grasp contact locations. The grasp mapping strategy is shown in Fig. 15. Details are given in Kang and Ikeuchi [10].

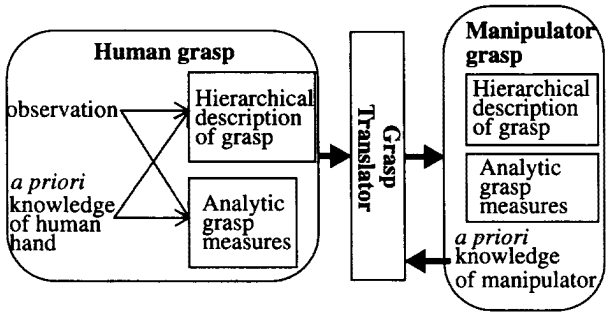

Fig. 15 **Grasp mapping strategy**

Two examples of power and fingertip precision grasp mapping are shown in Fig. 16.

Once the grasp has been planned, its path (which is approximately that taken by the human hand) is then verified in the simulator before control signals are sent to the robot system.

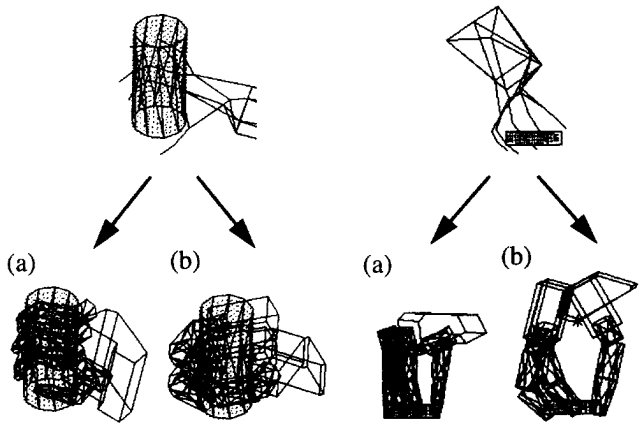

Fig. 16 **Results of mapping a (left) cylindrical power grasp and (right) a fingertip precision grasp for (a) Utah/MIT and (b) Salisbury hands**

5 System implementation

The arm/hand system used to verify the grasp mapping technique comprises the PUMA 560 arm and the Utah/MIT hand. Both the arm and hand is controlled via Chimera, which is a real-time operating system for sensor-based control developed at Carnegie Mellon University [21]. The joint angles of the arm and the hand are fed from the grasp simulator to the arm/hand system via Chimera. Two example grasps executed by the robot system are shown in Fig. 17.

Fig. 17 **Cylindrical power grasp (left) and fingertip precision grasp (right) by the Utah/MIT hand**

An example task that was executed by the robot system is shown in Fig. 18. This task is the recorded peg-and-receptacle task as shown in Fig. 5. The task breakpoints were identified, the grasps recognized and object motion extracted prior to robot planning and execution.

6 Concluding remarks

In this paper, we have described the four components of our programming-by-demonstrating system, namely the observation system, the task recognition module, the task translator, and the robot system. This system 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. We have also shown that by using complimentary sensory data from an active

multibaseline stereo system and a dataglove, we can more reliably track objects being manipulated during human task execution.

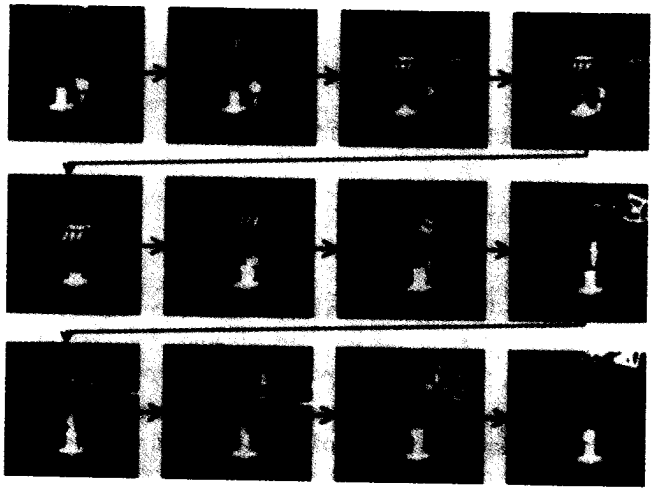

Fig. 18 **Different temporal snapshots of task execution by PUMA arm and Utah/MIT hand system**

Acknowledgments

Many thanks to Richard Voyles for helping us with the PUMA arm and Utah/MIT hand system.

This research is sponsored in part by the Advanced Research Projects Agency under the Department of the Army, Army Research Office under grant number DAAH04-94-G-0006, in part by 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 No. 7597, and in part by National Science Foundation under Contract CDA-9121797.

References

[1] M.A. Arbib, T. Iberall, and D.M. Lyons, "Coordinated control programs for movements of the hand," *Hand Function and the Neocortex*, eds. A.W. Goodwin and I. Darian-Smith, Springer-Verlag, 1985, pp. 111-129.

[2] H. Asada and Y. Asari, "The direct teaching of tool manipulation skills via the impedance identification of human motions," *Proc. IEEE ICRA*, 1988, pp. 1269-1274.

[3] P. Balakumar, J.C. Robert, R. Hoffman, K. Ikeuchi, and T. Kanade, *VANTAGE: A Frame-based Geometric Modeling System - Programmer/User's Manual V2.0*, Tech. Rep. CMU-RI-TR-91-31, Carnegie Mellon Univ., Dec. 1991.

[4] S. Borkar, R. Cohn, *et al.*, "Supporting systolic and memory communication in iWarp," *Proc. 17th Int'l Symp. on Computer Architecture*, Seattle, WA, 1990, pp. 70-81.

[5] R. Finkel, R. Taylor, R. Bolles, R. Paul, and J. Feldman, *AL: A programming system for automation*, Tech. Rep. AIM-177, Stanford Univ., AI Lab., 1974.

[6] W.A. Gruver, B.I. Soroka, J.J. Craig, and T.L. Turner, "Evaluation of commercially available robot programming languages," *Proc. 13th Int'l Symp. on Industrial Robots*, 1983, pp. 12-58.

[7] K. Ikeuchi and T. Suehiro, "Towards an Assembly Plan from Observation, Part I: Task recognition with polyhedral objects," *IEEE Trans. on Robotics and Automation*, vol. 10, no. 3, 1994, pp. 368-385.

[8] M. Jeannerod, "Intersegmental coordination during reaching at natural visual objects," *Attention and Performance IX*, eds. J. Long and A. Baddley, Erlbaum, Hillsdale, NJ, 1981, pp. 153-168.

[9] M. Jeannerod, "The timing of natural prehension movements," *J. of Motor Behavior*, vol. 16, no. 3, 1984, pp. 235-254.

[10] S.B. Kang and K. Ikeuchi, "Robot task programming by human demonstration: Mapping human grasps to manipulator grasps," *Proc. IEEE/RSJ/GI IROS*, München, Germany, 1994, pp. 97-104.

[11] S.B. Kang and K. Ikeuchi, "Determination of motion breakpoints in a task sequence from human hand motion," *Proc. IEEE ICRA*, 1994, pp. 551-556.

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

[13] S.B. Kang and K. Ikeuchi, "A grasp abstraction hierarchy for recognition of grasping tasks from observation," *Proc. IEEE/RSJ IROS*, July, 1993, pp. 194-201.

[14] S.B. Kang, J. Webb, C.L. Zitnick, and T. Kanade, "An active multibaseline stereo system with real-time image acquisition," *Proc. Image Understanding Workshop*, Monterey, CA, Nov. 1994.

[15] T. Kuniyoshi, M. Inaba, and H. Inoue, "Teaching by showing: Generating robot programs by visual observation of human performance," *Proc. 20th ISIR*, 1989, pp. 119-126.

[16] P. Lammineur and O. Cornillie, *Industrial Robots*, Pergammon Press, 1984, pp. 43-54.

[17] T. Lozano-Perez, "Automatic planning of manipulator transfer movements," *IEEE Trans. on Systems, Man and Cybernetics*, SMC-11(10), 1981, pp. 681-689.

[18] D.M. Lyons, "A simple set of grasps for a dextrous hand," *Proc. IEEE ICRA*, 1985, pp. 588-593.

[19] P. Morasso, "Spatial control of arm movements," *Experimental Brain Research*, vol. 42, no. 1, 1981, pp. 223-227.

[20] M. Okutomi and T. Kanade, "A multiple-baseline stereo," *Proc. IEEE CVPR*, 1991, pp. 63-69.

[21] D.B. Stewart and P.K. Khosla, *Chimera 3., The Real-Time Programming Operating System for Reconfigurable Sensor-Based Control Systems*, Carnegie Mellon University, 1993.

[22] T. Takahashi and H. Ogata, "Robotic assembly operation based on task-level teaching in virtual reality," *Proc. IEEE ICRA*, 1992, pp. 1083-1088.

[23] J.A. Webb, T. Warfel, and S.B. Kang, *A Scalable Video Rate Camera Interface*, Tech. Rep. CMU-CS-94-192, Carnegie Mellon University, 1994.

[24] M.D. Wheeler and K. Ikeuchi, "Sensor modeling, probabilistic hypothesis generation for object recognition," *Proc. 2nd CAD-Based Vision Workshop*, 1994, pp. 46-53.