Sponsored by the IEEE Council on Robotics and Automation

General Chairman: **T. Pavlidis,** SUNY at Stony Brook, NY
Program Chairman: **R.B. Kelley,** Rensselaer Polytechnic Inst.
Treasurer and Coordinator: **Harry Hayman**
Local Arrangements: **R.P. Paul,** University of Pennsylvania

April 25-29, 1988
Franklin Plaza Hotel
Philadelphia, PA

(1)

Workshop on Dextrous Robot Hands

Organizers:
Dr. Thea Iberall
University of Southern California
&
S.T. Venkataraman
University of Massachusetts

IEEE THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, INC.

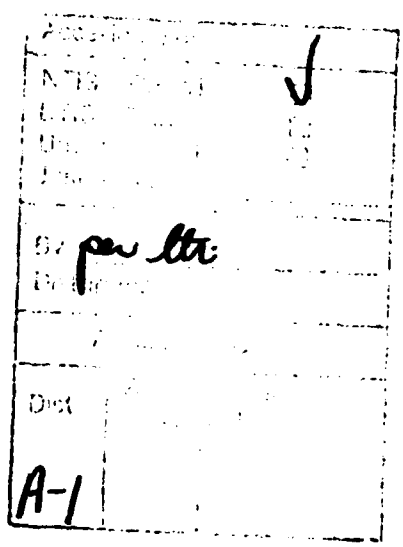
WORKSHOP ON DEXTEROUS ROBOT HANDS

Dr. Thea Iberall
University of Southern California

&

S.T. Venkataraman
University of Massachusetts

April 5, 1988



A

Table of Contents

Chapters

(KR)

CHAPTER 1

SESSION I: HUMAN HAND STUDIES

IEEE WORKSHOP PAPER DRAFT
3/88

INTELLIGENT EXPLORATION BY THE HUMAN HAND

Roberta L. Klatzky
Dept. of Psychology
University of Calif.
Santa Barbara, CA

Susan Lederman
Dept. of Psychology
Queen's University
Kingston, Ontario

BACKGROUND

For the past several years, we have been engaged in a research program to study the competencies of the human haptic system for object recognition. First, a definition: Haptics includes cutaneous sensors in the skin, providing pressure vibratory, and thermal information, and mechanoreceptors in muscles, tendons, and joints, providing position information. Thus the system comprises both touch and movement. Following Gibson (1962), we stress the function of haptics as a purposive exploratory system rather than a passive receiver.

Our research program began with two seminal findings. First, we documented that the haptic system is extremely effective at object recognition (Klatzky, Lederman, & Metzger, 1985). In a study in which subjects were given 100 common manipulable objects and asked to name them, the success rate was 96%, and if near misses were accepted, it was 99%. The modal response latency was 1-2 sec, indicating that not only was recognition accurate, it was fast. This first finding, then, led us to ask, what enables the haptic system to recognize objects so well?

Our second finding tells a more complex story, but one with an equally simple point. The point is that the haptic system capitalizes on the motor capabilities of the hand, in order to enhance its sensing capabilities. The result is a set of specialized "exploratory procedures," or stereotyped movements for apprehending properties of objects (Lederman & Klatzky, 1987).

Much of our work is based on the documentation of exploratory procedures, and we will therefore go into them in some detail. Figure 1 indicates the procedures we have studied. Six of them are directed to encoding basic properties of the substance or structure of objects: texture, hardness, apparent temperature, weight, size, and global and exact shape. We consider these six properties to virtually exhaust the directly sensed, fundamental properties of objects, although others are arguable (for example, instead of "hardness," one might distinguish among rigidity, brittleness, and elasticity). Two other properties we considered are high-order attributes -- the nature of part motion, and the function of the object as inferred from its structure.

The figure indicates that each property can be paired with an exploratory procedure that elicits it. An exploratory procedure is a way of interacting with an object that has certain invariant and typical features. Lateral motion, for example, is the procedure used to extract texture. Its invariant property is lateral movement between the skin and object. Its typical properties are rapidity, repetition, and occurrence on a homogeneous portion of the object surface rather than an edge or junction of different materials. To go on with our list of procedures, there is pressure for encoding hardness; static contact for thermal sensing; unsupported holding for weight; enclosing for volume and gross contour information; and contour following, which is used to extract precise contour information as well as global shape. Specialized performative procedures are also found for encoding our two higher-level object properties, function and part motion. These procedures essentially execute the function or motion.

How do we know these procedures are used? We asked a group of subjects to take part in a match-to-sample task. On each trial, they were given a sample object, followed by three candidates for a match. They were to pick the candidate that best matched the sample. Most importantly, on each trial some object property was targeted for matching. For example, subjects were asked to match the objects on their envelope shape -- roughly spherical, rectangular, or whatever -- regardless of texture, weight, size, and so on. The results showed clearly that the distribution of exploratory procedures shifted with the targeted property. Specifically, use of the procedure(s) relevant to a given property increased sharply when that property was targeted.

INFLUENCES ON EXPLORATION: A CONCEPTUAL MODEL

Up to this point, we have presented the background to the main point of today's talk. Our concern here is with the factors that control the course of haptic object exploration. We can express this concern in the context of a conceptual model of haptic object processing, as described in Figure 2. The model identifies three distinct levels of information representation. The top level represents information about objects -- their parts, their surfaces, their component materials, and so on. The middle level represents information about perceptual attributes such as roughness, divorced from particular objects. The bottom level represents information about exploratory procedures. A recognition model making these distinctions was implemented successfully by Sharon Stansfield (1987) with a robotic arm and end effector.

We view these three levels as modules within a parallel interactive system, in which information is activated top-down and bottom-up. The object level "calls" down for object attributes, based on hypotheses; it may also call directly for a test of higher-level attributes such as function. The attributes level works bottom-up to activate object information; it works downward to call specific procedures. Each attribute sends a

call for the exploratory procedure that is optimally associated with it; for example, the texture attribute would call the lateral motion procedure. Conversely, each exploratory procedure, when executed, sends information about the values on the corresponding attribute dimension, acting bottom-up. The system acts in parallel over these various pathways, but exploratory procedures cannot all be executed in tandem. Thus there is competition among the various sources for control of exploration. Some mechanism is necessary for ultimate decisions about what procedure to execute next.

A number of factors influence this decision process. Each can be associated with a level in our system. These influences include the top-down calls that activate exploratory procedures more or less directly. There are calls from the objects level for attributes and calls from the attributes level for corresponding patterns of exploration. However, there are some less obvious influences and constraints on how exploration can proceed, which we will next consider.

At the top level, where objects are represented, one influence is the modality of representation that is currently being constructed. Within a robotic or human system, an object might be described in terms of visual, tactual, or abstract conceptual primitives. Each of these constitutes a "modality," in present terms. Psychologists distinguish among modalities not only at the perceptual level, but also at higher cognitive levels. There is substantial evidence that the cognitive representation of an object can be abstract and related to its meaning or function, or alternatively, more concrete and analogous to physical media. One much-studied representational medium is visuospatial imagery (e.g., Kosslyn, 1980). Such a representation will convey different attributes than imagery in a haptic modality, or a direct representation of an object's function.

Another influence on exploration arises from the middle or attributes level in our system. It is the distribution of previously perceived values on various attribute dimensions. Consider, for example, the attribute of hardness. In the world of common objects, most are rigid. A very soft object represents an extreme on the distribution of hardness values. Its softness is unusual, which might initiate further exploration. We must consider not only the distributions of values on single dimensions but also conjoint distributions, which also might have extremes. An object that is <u>small</u> but that seems <u>heavy</u> might induce further exploration for weight, for example.

Several influences on exploration arise from the bottom level, because of the nature of exploratory procedures. There are factors which might impose a general preference ordering on exploratory procedures. These would include the energy required for the motor act. The extent to which a procedure is broadly sufficient for apprehending object properties, as opposed to being specialized, is also a likely influence on exploration,

with general procedures being preferred for early, open-ended encoding.

Other influences at the bottom level occur because procedures are motor acts, operating under motor constraints. Whether a procedure is executed is likely to depend on other ongoing actions. For example, some procedures may be motorically compatible, so that one can be initiated while another is being executed. We will provide evidence that this is the case for lateral motion and pressure, which can be evoked together by a hybrid smearing action that applies lateral and normal force. Another motor constraint is manipulatory -- whether a procedure is executed may depend on its compatibility with manipulatory activities of the exploring or the free hand. The pressure procedure, for example, cannot be executed without stabilization of the object, either by holding it while squeezing or with the nonexploring hand.

To summarize, we have proposed a substantial list of influences and constraints that should play a role in directing haptic exploration at all levels. Our research to date has exposed and illuminated a number of these influences. In the following sections, we will review that research and point out its relevance to the issue of exploratory control.

INFLUENCES ON EXPLORATION: EMPIRICAL FINDINGS

The Object Level

Influence of Representational Modality on Exploration. The first influence on exploration I will describe is at the top level in our system -- it is the modality of the object representation being constructed. Our work has contrasted, in particular, representations in the modalities of visual perception, visual imagery, and haptic perception. We predicted that structural attributes, particularly contour information, would predominate in vision and visual imagery, whereas the haptic modality would be likely to emphasize attributes related to substance, such as texture and hardness.

To address the representation issue, we had people explore objects freely, either haptically, or haptically and visually (Klatzky, Lederman, & Reed, 1987). Different task instructions, along with these perceptual variations, were intended to lead to four different types of mental representation. The cover task for all groups was a similarity judgment. Participants were asked to sort a set of objects into bins, so that objects that were similar were placed into a common bin. Under unbiased instructions, with or without vision, participants were not given any particular definition of similarity. These conditions should lead to representations directly based on haptic, or haptic and visual, perception. To bias a visual imagery representation, we told participants to think of similarity in terms of the object's visual image (remember that exploration was haptic without vision). Finally, to induce a haptically biased representation,

we told them to think of similarity in terms of how the objects "felt."

The objects in this study were created from all combinations of three values on each of four dimensions: hardness, shape, roughness, and size. Thus, for example, each object was rigid, slightly compliant, or soft; it was oval, hourglass-shaped, or clover-shaped; and similarly, it had one of three possible roughness values and three possible sizes. Each object was planar, that is, it had a homogenous surface area bounded by a two-dimensional contour, and its third dimension was thin and invariant.

From the participants' sorting behavior, we created a measure of how vivid, or "salient," each dimension was in the object representation. A dimension was salient to the extent that its levels were segregated in different bins. Thus for example, shape would be maximally salient if all the ovals were in one bin, all the hourglass shapes in another, and all the clovers in a third. In this case, each other dimension would have to have zero salience, because its levels would be mixed within each bin. The bin of oval shapes, for example, would have to include some of each possible size.

Finally, in addition to the salience score of each dimension, we had a measure of how frequently each of four exploratory procedures occurred. Those procedures were lateral motion -- related to texture; pressure -- related to hardness; contour following -- related to shape; and enclosure -- related to shape and size. Our concern was not only with salience, but with its relationship to the pattern of object exploration.

Our results are summarized Figure 3. It shows data for each dimension in a separate panel. Within each panel, we can see how groups with different instructions fared. There are two dependent measures of interest -- the salience of the dimension, and the occurrence of the directly related exploratory procedure(s). The salience scores indicate differences in how the objects were represented, depending on the perceptual condition and instructions. Consider the dimension of shape: This had relatively low salience for groups that were exploring haptically without any particular bias, or that were instructed to think about how objects felt. But these same groups showed relatively high salience for the substance dimensions of texture and hardness. The visual imagery group showed strong salience for the shape dimension. And vision acted like a moderator: Those who could see the objects as well as feel them found shape and substance, particularly texture, somewhat salient. We note that size was not found salient by any group, possibly because of the range of sizes we used. (Also, this design pits dimensions against one another, so that size cannot be found salient when other dimensions dominate.)

Next consider the patterns of exploration, as indicated by the percentages of occurrence of targeted exploratory procedures.

For the groups who were denied vision, there is a direct relationship between the salience of a dimension, and the extent to which relevant exploratory procedures are performed. When shape is salient, there is exploration for shape -- contour following and enclosure. When texture is salient, lateral motion tends to occur; and similarly for hardness and pressure. Not surprisingly, vision strongly reduces the amount of haptic exploration that occurs.

These data show a clear relationship between the salience of an object property to a haptic explorer, and the pattern of exploration. Further, this pattern of results clearly reflects an influence on exploration of the desired modality of representation. Consider, for example, someone who wants to form a visual image. The external envelope of the object is the most important property in such a representation. Accordingly, a call may be sent for shape information, leading to enclosure and contour following. Similarly, instructions to consider what objects feel like appear to lead to calls for texture and hardness, and hence to the appropriate procedures for exploration.

On the other hand, the relationship between salience and exploration may also reflect influences from the bottom level of our system. We have previously found that contour following is relatively slow and subject to considerable error with complex contours. Lateral motion and pressure, in contrast, can be executed very quickly. In the absence of instructions to form a particular representation, people may choose to execute these latter procedures because of their low demands on motor energy. As a result, the representation may emphasize the corresponding dimensions of texture and hardness.

In short, either representational salience may invoke patterns of exploration, or patterns of exploration may determine representational salience. When salience invokes exploration, the influence acts top down. When exploration determines salience, the influence is bottom up.

The Attributes Level

Influence of Attribute Distributions. We next consider what people know about object attributes through experience with haptic perception, and how that knowledge influences exploration. They are likely to know the general range of values that objects take on a dimension such as hardness. They are also likely to know which values on different dimensions tend to co-occur. We call such co-occurring values "natural correlations." For example, we know that large objects tend to be heavy; a large light object such as a balloon is an anomaly. How might such knowledge of distributions affect exploration? One strong possibility is that an anomalous observation leads to further exploration, for purposes of verification. Another possibility is that knowledge of natural correlations can be used to "prune" the tree of potential exploratory movements, eliminating those

that are likely to encode redundant features of objects.

We are currently planning a program of research to determine which attributes of objects tend to be correlated, and how correlations affect exploration. Some preliminary data on this point derive from another study (Lederman & Klatzky, in progress), in which people were asked to rank attributes according to their importance in categorizing objects. For example, weight might be critical in categorizing an object as a cast-iron frying pan. We found that certain attributes tended to be correlated, in that if one was highly ranked, the other was as well. The most strongly related pair, in this sense, was size and shape. Texture and hardness were also highly correlated. Thus there was a general pattern of finding correlations between structural categories, or between substance categories, and less so across these boundaries.

The Exploratory-Procedure Level

Earlier, in conjunction with the study of mental representations, we raised the possibility that the effort or energy involved in an exploratory procedure may determine whether it is used. This influence is associated with the bottom level of our system; that is, it is due to the nature of exploratory procedures themselves. We now consider a number of influences associated with that level.

Influence of the Sufficiency, Optimality, and Generality of Exploratory Procedures. The first factor to be considered is the breadth of attribute information that an exploratory procedure provides. We distinguish among exploratory procedures according to their sufficiency, necessity, optimality, and generality for encoding haptic attributes (Lederman & Klatzky, 1987).

By our definition, an exploratory procedure is sufficient to encode an object attribute if it permits one to discriminate between objects along that particular dimension. For example, enclosing an object is not the best way to encode its texture, but it might be sufficient to do so, because of small-scale lateral movements that occur during the act of enclosing.

An exploratory procedure is more than sufficient to encode some dimension -- it is optimal -- if it provides better discrimination performance, in terms of speed or accuracy, than other exploratory procedures. And if it is the only procedure to be sufficient to encode that dimension, it is termed necessary.

Sufficiency, optimality, and necessity are judged of each procedure relative to each dimension. By looking across haptically encoded dimensions, we can ask whether an exploratory procedure is specialized, or in contrast, general. A procedure is specialized to the extent that it discriminates one attribute well, and the others much less well. It is general to the extent that it encodes multiple attributes at about the same level of

discriminability.

The importance of sufficiency, optimality, and generality for patterns of object exploration should be clear. Why waste time on a nonoptimal procedure, if there is only one attribute of an object that is to be encoded? Alternatively, if many attributes are of interest, a nonspecialized, highly general procedure is of greater value. Thus the task context, together with the known performance of an exploratory procedure, is likely to strongly influence which procedure is executed.

We have tested the sufficiency, optimality, and generality of exploratory procedures in a variant of the match-to-sample task. In this case, participants were constrained to explore in a particular way, and to match objects on a particular attribute. All combinations of exploratory procedures and attributes were tested. For example, the lateral motion procedure was used to match not only texture, but weight, size, and so on.

Figure 4 shows the results of this study. We can see that in general, the procedure that was spontaneously used in the unconstrained match-to-sample task is also the optimal one, in terms of accuracy, speed, or both. Thus, for example, lateral motion gave the highest accuracy for texture matching. In the case of matching precise contour, contour following was not only optimal but necessary. Another interesting outcome of this study was that procedures differed in their generality of application, or conversely, their specialization. Pressure was the most specialized procedure, and enclosure the least. Enclosure was grossly sufficient, in that it produced above-chance accuracy on matching most attributes of objects but was generally not optimal. If enclosure with some pressure were accompanied by lifting (unsupported holding), that is, if a simple grasp were performed, it is clear that we could find out a great deal about an object very rapidly. For this reason, grasping would be ideal for initial contact.

Exploratory Routines. In fact, grasping does appear to be our favored way of contacting unknown objects. In this sense it constitutes a habitual "routine" for exploration. The existence of such general routines may be an important determiner of exploration, at least in its early stages.

We examined exploratory routines as part of an ongoing study of object categorization (Lederman & Klatzky, in progress). Blindfolded subjects were given an object in their upraised palms and asked if it was in a particular category. Of interest here is that they tended to follow highly routinized patterns of exploration for the first few moments. Although there were variations, the most general pattern was an enclosure of the object in one or both hands, followed by unsupported holding. This routine is likely to be very successful for recognizing objects, especially at the "basic level" -- the level of common naming, such as jar, pencil, and so on. Enclosure, we know, is sufficient to grossly discriminate values on many attributes.

Accompanied by lifting and even slight pressure, other attributes are added, and a snapshot of the object can be obtained with minimal exploratory effort.

Influence of Compatibility of Exploration and Task Context. Another bottom-level factor likely to be critical to the selection of an exploratory movement is its compatibility with other aspects of the task environment. Compatibility takes various forms. A procedure may be compatible or incompatible with other exploratory procedures that are also under execution. Or its compatibility may be judged relative to general manipulatory constraints in the task context, for example, whether the object is fixed or moveable or whether one or two hands may be used.

We have found clear evidence of compatibility effects of this latter sort in the bin-sorting task described previously (Klatzky, Lederman, & Reed, 1987). Recall that objects varying in size, shape, hardness, and texture were sorted and the most salient dimension(s) was determined. In one condition, we constrained participants to sort the objects either with one or both hands. This was intended to affect their ability to stabilize the object and to reorient it for exploration in different regions.

If two hands are available, one can serve the stabilizing and orienting function (which we have called "task maintenance"), and the other can serve the exploratory function. Given one-handed exploration, the two functions can still be observed: Part of the hand typically stabilizes and part explores. This limits the ease and extent of exploration, of course. Procedures such as lateral motion and pressure, which are performed in a small region of the object, can easily be performed along with stabilization by the same hand. Contour following is more difficult under one-handed conditions. It tends to occur in a broader region, and it requires not only stabilization but also periodic reorientation to bring new parts of the contour into focus. If contour following is limited by one-handed exploration, we should see an effect on the salience of the corresponding object property -- shape.

The results of this manipulation showed a clear effect. In the condition where shape tended to be salient (visual imagery), it was far more so under two-handed exploration. In fact, the salience score was twice as great as for the one-handed condition. Evidently, this general constraint on the manipulatory context influenced the course of exploration and apprehension of object properties.

Influence of Compatibility Between Exploratory Procedures. As noted above, we can also speak of compatibility between exploratory procedures. Some procedures can effectively be produced together, whereas others may interfere. Hence if one procedure is selected, it will influence what others are chosen.

We have found compatibility of exploratory procedures to affect performance in a classification task, using the same multiattribute objects described previously (Klatzky, Lederman, & Reed, in progress). Participants repeatedly classified 9 stimuli into three categories. The categories could be defined by one dimension only; for example, As are hard, Bs medium hard, and Cs soft. Alternatively, the categories could be defined by two dimensions redundantly: As are hard and oval, Bs medium hard and two-lobed, and Cs soft and three-lobed. We also included a condition where three dimensions -- texture, hardness, and shape -- redundantly classified the objects. (Size was not used as a redundant dimension in this study.) As each object was classified, we measured the response latency, defined as the time between initial contact with the object and the vocal response. Our interest was in redundancy effects -- reductions in response time due to additional redundant cues to classification -- and also in the exploratory procedures executed during classification.

We did find that redundant dimensions speeded classification, but only to a point. Two redundant dimensions led to faster classification than one, but a third redundant dimension did not further reduce response times. Examination of exploratory procedures revealed why: Given the redundant dimensions of texture and hardness, subjects tended to execute both. When contour was provided as a further cue to classification, subjects largely behaved as if it were not present. They tended to produce the same exploration as for texture and hardness, with relatively little contour following.

A variety of compatibility effects may underlie this preference to execute the procedures for texture and hardness. One is motoric. The relevant exploratory procedures can be produced together, in the form of a hybrid movement with both lateral and normal force. In fact, this was frequently observed. Further, the extraction of hardness competes with the extraction of shape information, in that the appropriate degree of normal force is quite different. In fact, applying pressure in order to determine hardness may deform the shape of a compliant object. Another aspect of compatibility concerns the appropriate region of the object for exploration. For thin planar objects, contour following is movement along the object's edge. In contrast, the preferred position for lateral motion and hardness is within a larger homogeneous region; the edge may actually interfere with their encoding.

Several converging operations in these studies supported the idea that texture and hardness are jointly encoded more than either dimension is joined with planar contour. In one study, subjects were instructed to classify objects on the basis of one dimension, such as texture. At the same time, another dimension varied redundantly with the explicitly mentioned one. After 100 or so classification trials under these conditions, the implicit redundant dimension was withdrawn. That is, the stimuli being classified were changed so that the redundant dimension was now

invariant. Of course, the explicit classification rule still applied.

Elimination of the implicit redundant dimension greatly interfered with classification, when the two dimensions were texture and hardness. When either dimension was paired with shape, however, the effects were minimal. It appears that texture and hardness were both considered, even under instructions to attend to just one. These results suggest that exploratory compatibility may ultimately determine whether dimensions are processed together in tasks such as object identification.

FINAL COMMENT

We have now discussed evidence, from our ongoing work on human haptics, for a variety of governing influences on patterns of exploration. To understand patterns of haptic exploration, it is essential not only to discern what factors influence it, but also to determine their weights in various contexts. Given a fuller understanding, our initial conceptual model may be developed to predict exploratory sequences for apprehension and identification.

REFERENCES

Gibson, J, J. (1966). The Senses Considered as Perceptual Systems. Boston: Houghton Mifflin.

Klatzky, R.L., Lederman, S.J., & Metzger, V.A. (1985). Identifying objects by touch: An "expert system". Perception & Psychophysics, 37, 299-302.

Klatzky, R.L., Lederman, S.J., & Reed, C. (1987). There's more to touch than meets the eye: The salience of object attributes for haptics with and without vision. Journal of Experimental Psychology: General, 116, 356-369.

Kosslyn, S. M. (1980). Image and Mind. Cambridge, MA: Harvard University Press.

Lederman, S. J., & Klatzky, R. L. (1987). Hand movements: A window into haptic object recognition. Cognitive Psychology, 19, 342-368.

Stansfield, S. (1987). Visually-guided haptic object recognition. University of Pennsylvania Dept. of Computer and Info. Science, Technical Report MS-CIS-87-93, Grasp Lab 122.

3.2 Opposition Space and Human Prehension

Thea Iberall and Christine L. MacKenzie

IEEE Workshop on Dextrous Robot Hands,
Philadelphia, Penn, April 24, 1988

ABSTRACT

Whether a prehensile task is performed by a dextrous robot manipulator or by a human hand, fundamental rules must be adhered to in order to ensure success. Prehension has been defined as the bringing to bear functionally effective forces to an object within a task, given numerous constraints. One key goal in prehension is the establishment and maintenance of a stable grasp. A second (more optional) goal is to maintain the ability to stably manipulate the object. In the robotics literature, these goals (usually referred to as the zero total freedom problem and the force closure problem) have been analyzed for a variety of robot hands. However, in trying to understand control mechanisms within the central nervous system, experimental psychologists study other problems besides the critical issue of maintaining a stable grasp. As the arm reaches out to grasp an object (the transport component), the hand preshapes into some suitable posture (the grasping component). Questions that psychologists ask involve the emergence of the posture during the preshaping of the hand, the timing of the various elements of the movement, the coordination of and limitations on the approximately thirty degrees of freedom of the hand and arm, the use of sensory information from various sensory modalities, the effect of object properties on the movement, and the effect of human motivations.

In this paper, we show how, by experimental manipulations, constraints on human prehensile movement can be observed. One example is Fitts' law, which states that movement time is directly proportional to the precision required in the task. Mackenzie et al [1987a] show how Fitts' law is related to the kinematics of aiming tasks. Marteniuk et al [1987] show how object characteristics, intention and context affect the trajectories and timing parameters of prehensile movements. Another, unanswered, question, is whether the transport component is controlled separately from the grasping component. Jeannerod [1981] has argued for separate control of transport and grasping, although there is a temporal coupling. Wing et al [1986] have demonstrated an interaction between them by increasing movement speed and removing vision. In Mackenzie et al [1987b], object mo-

tion selectively affected the transport component, while object size affected only the grasping, not transport component.

While it is useful to identify the types of constraints acting on prehension, it is not enough for developing a common framework for looking at robotic and living systems. Following the lead of Marr, who suggested three levels for looking at vision (task level, representation level, and implementation level), we argue for levels of analysis for looking at prehension. For a prehensile task (as defined above), we call the highest level the opposition space level, in that high level kinematic and force related prehensile parameters are used to describe movements. A posture in opposition space is re-represented, or mapped, into the biomechanical level, showing the forces and torques acting at the joints (in robotics, joint space). This is re-represented at the lowest level, the anatomical level, in terms of the activation level of the muscles acting on the fingers (or in robotics, motor commands). Prehensile movement can then be described in goal directed terms: in terms of how many forces are needed in the task, in what direction they are to be applied, and at what strength they must be applied. The actual movement occurs obeying the constraints imposed on the lower levels, whether the implementation is a human hand or dextrous robot one. Beyond just separating implementation details from a task description, this proposed view of prehension allows the separation of constraints, separating hard, physical constraints at the lowest level from softer, functional constraints at the highest level.

References

1. MacKenzie et al QJEP, 1987.

2. Marteniuk et al, Can J Psych, 1987.

3. MacKenzie et al Soc Neuro Abstr, 1987.

4. Jeannerod, Attention and Performance IX, 1981.

5. Wing et al, J Motor Behavior, 1986.

PROFICIENT PROSTHETIC PREHENSION

Alan M Wing

MRC Applied Psychology Unit
15 Chaucer Rd, Cambridge CB2 2EF
England, [alanw@uk.ac.cam.mrc-apu]

April 5, 1988

1. Thirsty?

If you want to take a tumbler of water in order to drink from it you will probably reach for it from the side with the forearm midway between pronation and supination (so that the palm is turned towards body midline). What does the reaching movement comprise? At the end of an initial, rapid, distance-covering phase of movement that leaves your hand close to its target, thumb and fingers will have opened sufficiently to allow the tumbler to be encompassed. The second phase is made more slowly and leads upto contact of the hand with the tumbler. In this phase, coordination between transport of the hand by the arm and changes in grasp size becomes critical if the impact of the collision on contact is to be minimised - especially if the tumbler is full to the brim!

Since transport and grasp are subserved by anatomically distinct elements that are capable of being moved separately under voluntary control, the nature of coordination in reaching is of some interest. How does the movement control architecture of the brain specify the functional linkage between transport and grasp? In this paper I will review empirical work that my colleagues and I in Cambridge, England have been carrying out to investigate this functional linkage. My main focus will be the performance of a proficient user of an artificial hand because it raises the interesting issue of how movements of the digits are represented in relation both to movements of the arm and to the external object.

2. Reaching for an unstable object.

In studying reaching behaviour it is useful to have a standardised task that is capable of yielding results that are reliable yet valid. For our research we have developed a task which, like many real-world reaching tasks, emphasises accuracy. A thin cylinder made, for example, of wood dowel is placed on end some distance in front of the subject (see Figure 1).

INSERT FIG 1 HERE

Starting with the hand close to the body, the subject is required to pick up the cylinder from the side, then keep the cylinder upright while moving it some distance before putting it down. Because the cylinders used are relatively lightweight and their diameters are much smaller than their length, errors in positioning the hand or in adjusting the hand's aperture when reaching are of consequence in that the cylinder is easily knocked over. Accurate coordination of transport and grasp components is therefore important, particularly if the location of the cylinder is slightly varied from trial to trial.

The task may be given a further dimension by using cylinders of various diameters. With objects of varying size it is found that the maximum aperture attained by the hand, which occurs around the end of the initial phase of reaching, is adjusted in proportion to the size (Jeannerod, 1981). This is probably in order to preserve a constant margin for position error as the hand moves in to encompass the object.

3. Strategic adaptation of grasp size to movement conditions.

To illustrate this reaching task I have taken data from a study of a group of normal subjects by Wing, Turton & Fraser (1986). Of interest were the effects of the availability of visual information about reaching on grasp aperture. It was found that maximum aperture of the hand was greater with the eyes closed than with the eyes open (see right column in Figure 2).

INSERT FIGURE 2 HERE

This increase in aperture probably served to provide greater tolerance for errors in hand transport which, under these conditions, cannot be corrected on the basis of visual checks of the position of the hand in relation to the object. A widening of maximum hand aperture was also observed when vision was not completely removed but just limited in usability. If a reaching movement is carried out very rapidly, there is little time for visual feedback to be processed in order to improve the end-point accuracy of the movement. When the subjects were asked to make rapid reaching movements, hand apertures were observed to be larger than when reaching at normal speed (as may be seen from the middle column of Figure 2).

Thus maximum hand aperture is related, not only to the 'external' constraint provided by the size of the object, but also to an 'internal' factor, namely, the manner of hand transport. If a visuomotor channel links the perception of intrinsic object attributes, such as shape, to a grasp control process and a separate channel links the extrinsic attribute of position to a transport controller (cf. Arbib, 1981), the effects of the experimental manipulations suggest the existence of a cross-link between the two channels.

A direct demonstration of such cross-linking is provided by a new paradigm described in Athenes & Wing (1988). When reaching for an object a perturbation of the spatial relation between hand and target object was introduced on randomly selected trials, either by deflecting the arm or by moving the object. On those trials when a perturbation could have occurred, but did not actually take place, a widening of maximum hand aperture was observed. This strategic adaptation to the perturbation conditions would have conferred a strategic benefit in improving the chances that the hand would encompass the target had a perturbation actually occurred. Thus, in functional terms, control of grasp and transport are interrelated.

4. Profile of prosthetic prehension.

Our earliest analysis of reaching behaviour involved a single-case studyof a young girl with an artificial hand (Fraser and Wing, 1981). We were interested in finding out whether the aperture adjustments that may be observed during reaching with the natural hand would also obtain in the artificial hand. Since the artificial hand movements were controlled by very different muscles, such similarities would be evidence for a level of movement representation above that of a motor programme specific to a given muscle group.

Thirteen years old at the time of the study, CY had, from the age of two, used a below-elbow prosthesis that provided a form of precision grip with a strong spring keeping the hand normally closed. Opening the hand involved tensioning a cable which ran to the elbow and across the back to a harness around the contralateral, right shoulder. Forward flexion of the right shoulder girdle (or movement of the left elbow away from the body) increased the tension and opened the hand against the action of the spring. The extended period of wearing this appliance from an early age gave CY a notable degree of confidence in its use in carrying out everyday activities.

INSERT FIGURE 3 HERE

Despite CY's proficiency in using her artificial hand it5will be appreciated appreciated that it lacked the attributes of compliance and tactile sensibility which normally contribute to the final contact and grip phase of picking up an object. Thus, for CY, vision was particularly critical in successfully picking up our unstable cylinder.

Figure 4 shows the transport and grasp components of natural and

INSERT FIGURE 4 HERE

artificial hand movements as a function of time. It will be noted that themo dulation of aperture with cylinder diameter seen in the natural hand is also evidenced in the artificial hand. However, there is a delay in closure of the artificial hand. Consequently, its closing takes place with less change in the distance between the hand (or the wrist to be precise) and the target and this is illustrated in Figure 5.

INSERT FIGURE 5 HERE

Although CY's artifical hand was quite natural in appearance, when it was opened or closed the design introduced approximately equal and opposite degrees of rotation of the thumb and finger. Despite this, the recordings of reaching movements showed a stability of the thumb relative to the line taken by the 'hand' in approaching the target object (see Figure 6). This stability was achieved by forearm rotation

INSERT FIGURE 6 HERE

in a horizontal plane around an axis near the wrist. The rotation tended to cancel movement of the thumb relative to the object while amplifying finger movement. (It also caused the elbow to move away from the body imposing extra tension on the control cable which had to be offset by further shoulder movement if the hand was to continue closing).

5. Internal representation of the hand.

Why had this girl evolved a strategy of stabilising thumb position in this way? It seems reasonable to suppose that the forearm rotation geared to contralateral shoulder movement had been developed as a result of experience to allow the separation of visual feedback about hand position and aperture. Maintainance of stability in one of a pair of opposition surfaces as the hand approaches a target object presumably simplifies the processing of visual feedback about positional accuracy.

This observation about prosthetic prehension also seems relevant to the relative stability of the natural thumb or of the finger in the contrasting reaching movements to pick up a small object from the surface of a table depicted in Figure 7.

INSERT FIGURE 7 HERE

With the forearm pronated (so that the palm of the hand faces down) most of the movement occurs in the thumb. When the forearm is semi-pronated (such that the palm of the hand becomes visible) most of the change in the aperture of the hand is achieved with the finger.

The contrasting contributions of thumb and finger movements to the development of aperture in the natural hand could be explained on anatomical grounds. As the adductor muscle pulls the thumb round towards the palm of the hand the range of movement conferred by the joint at the base of the thumb is progressively lost. The greatest freedom for thumb movement is when it is abducted and moves in the plane of the palm. And in that position it moves against the side of the index finger which has relatively little side to side movement, particularly when the metacarpophalangeal joint at its base is flexed. But while these anatomical constraints on movement certainly exist, the organisation of grasp seen in the case of prosthetic prehension further suggests the form of movement has implications for the functional representation of movement control of the hand. The intriguing question that remains to be answered in CY's case is whether thumb stability in the artificial hand arose as a matter of trial and error. Alternatively it could have developed through transfer of the control structure used with the natural hand.

6. References.

Arbib, M. (1981) Perceptual structures and distributed motor control. In Brooks, V.B. (Ed) Handbook of Physiology, Section 1: The Nervous System, Volume 2: Motor Control. Bethesda, Md: American Physiological Society.

Athenes, S. & Wing, A.M. (1988) Knowledge directed coordination in reaching for objects in the environment. In Wallace, S. (Ed) Perspectives on the Coordination of Movement. Elsevier: Amsterdam, Netherlands.

Fraser, C. & Wing A.M. (1981) A case study of reaching by a user of a manually-operated artificial hand. Prosthetics And Orthotics International, 5, 151-156.

Jeannerod, M. (1981) Intersegmental coordination during reaching at natural visual objects. In Long, J. & Baddeley, A. (Eds) Attention and Performance IX. Hillsdale, NJ: Erlbaum.

Wing, A.M. & Fraser, C. (1983) The contribution of the thumb to reaching movements. Quarterly Journal of Experimental Psychology, 35A, 297-309.

Wing, A.M., Turton, A. & Fraser, C. (1986) Grasp size and accuracy of approach in reaching. Journal ofMotor Behavior, 18, 245-260.

7. Figures.

1. Prototypical pick-up paradigm.

2. Illustrative trajectories for subject C with group averages for maximum aperture and pre-contact SD (in mms). Mean data from 10 Ss, 6 trials (reproduced from Wing, Turton & Fraser, 1986).

trajectory normal fast eyes closed .... max aper 80 93 116 pre-cntct SD 0.3 0.5 0.9

3. CY performing activites of daily living (VHS video available).

4 Hand position and aperture as a function of time when using artificial (left) and natural (right) hands to pick up wide and narrow objects.

5. Artificial and natural hand aperture plotted against hand position.

6. Illustrative trajectories from two single reaches by CY; one with the artificial hand, the other with the natural hand showing change of thumb and finger positions relative to wrist-dowel axis (in mms). Mean data for both narrow and wide objects over 4 trials are shown below.(reproduced from Wing & Fraser, 1983).

trajectory left (artificial) right .... thumb +4 -3 finger +26 +15

7. Normal hand positions of thumb, finger and wrist as a function of time illustrating single-sided opening and closing on two trials, one (a) with palm facing down (thumb moves) the other (b) with palm to side facing midline (finger movement predominates).

7. Figures.

1. Prototypical pick-up paradigm: the task is to pick up the unstable cylinder between the pads of the thumb and index finger keeping with forearm midway between pronation and supination.

2. Illustrative trajectories for subject C with group averages for maximum aperture and pre-contact SD (in mms). Mean data from 10 Ss, 6 trials (reproduced from Wing, Turton & Fraser, 1986).

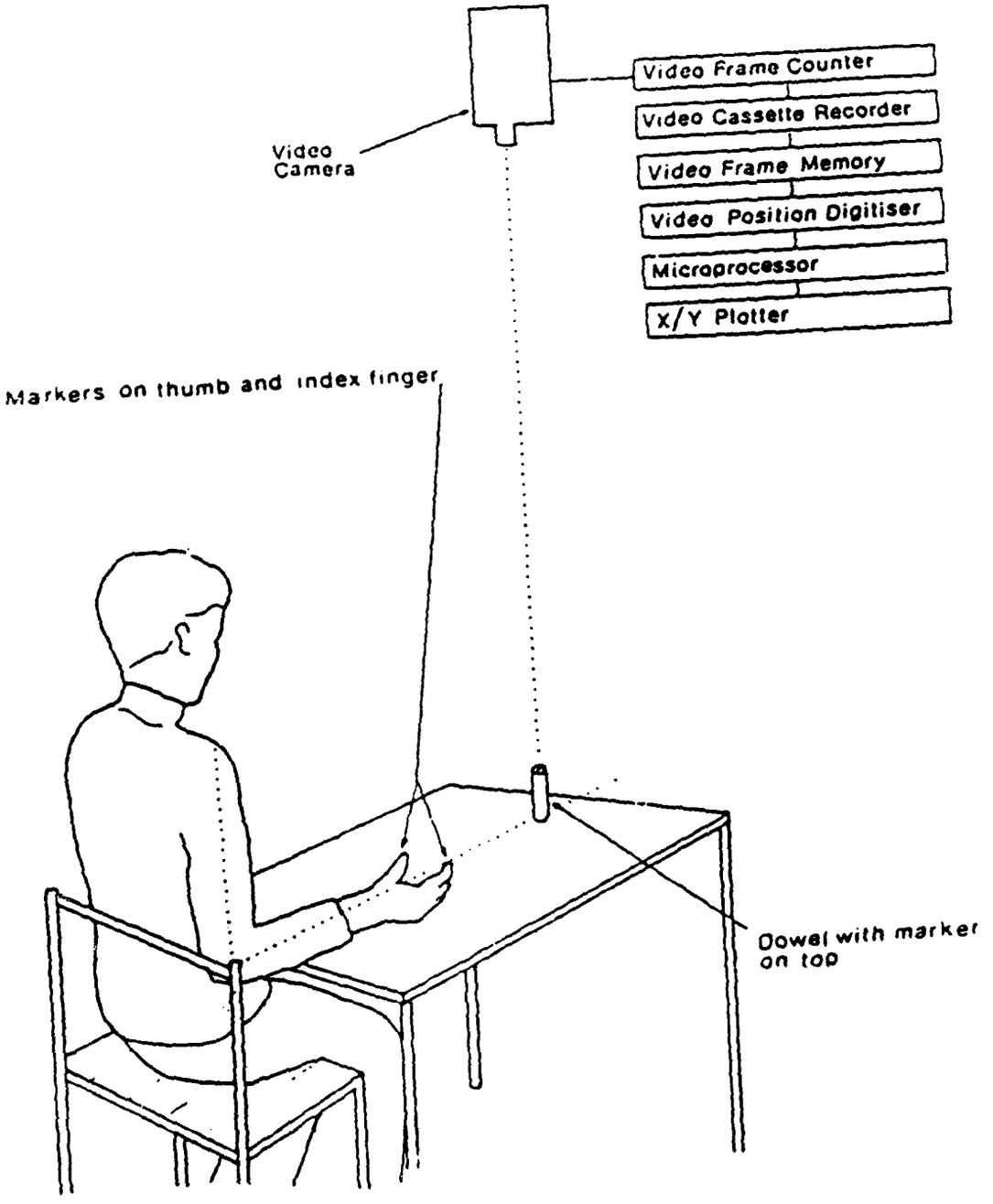
3. (Left) CY transferring pegs using her natural hand, leaving the artificial hand op cord (which runs from the right shoulder, across the back to the left elbow) untensioned so that the prosthesis is held closed by its internal spring. (Right) The camera viewpoint of the reach and grasp task. Note the 20 mm calibration grid (not present during data collection) and the markers on thumb, finger and wrist of CY's artificial left hand (used to aid subsequent digitisation of position).

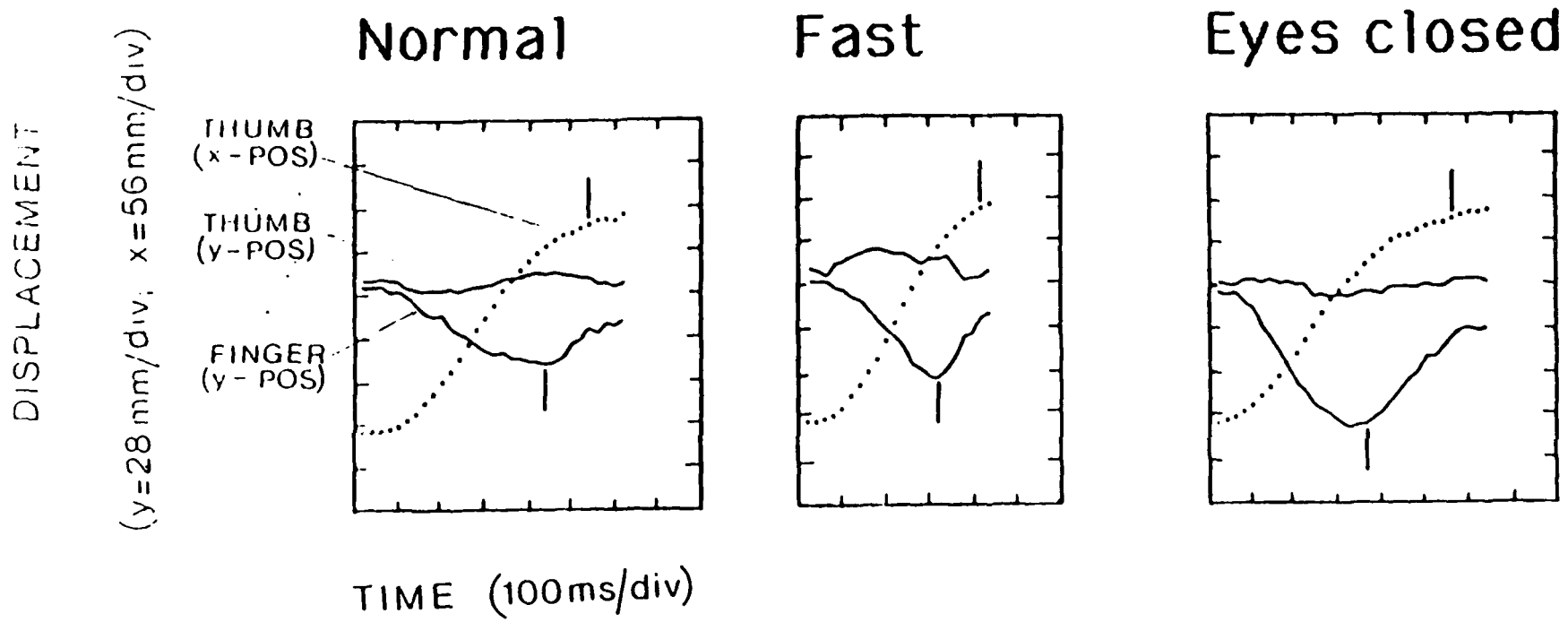
4 Illustrative trajectories for CY reaching and grasping 12 (dashed) and 22 mm (solid line) cylinders. Hand position (above) and aperture (below) are plotted as a function of time when using artificial (left) and natural (right) hands.

5. Illustrative trajectories for CY showing artificial and natural hand aperture as a function of distance of the wrist from the cylinder.

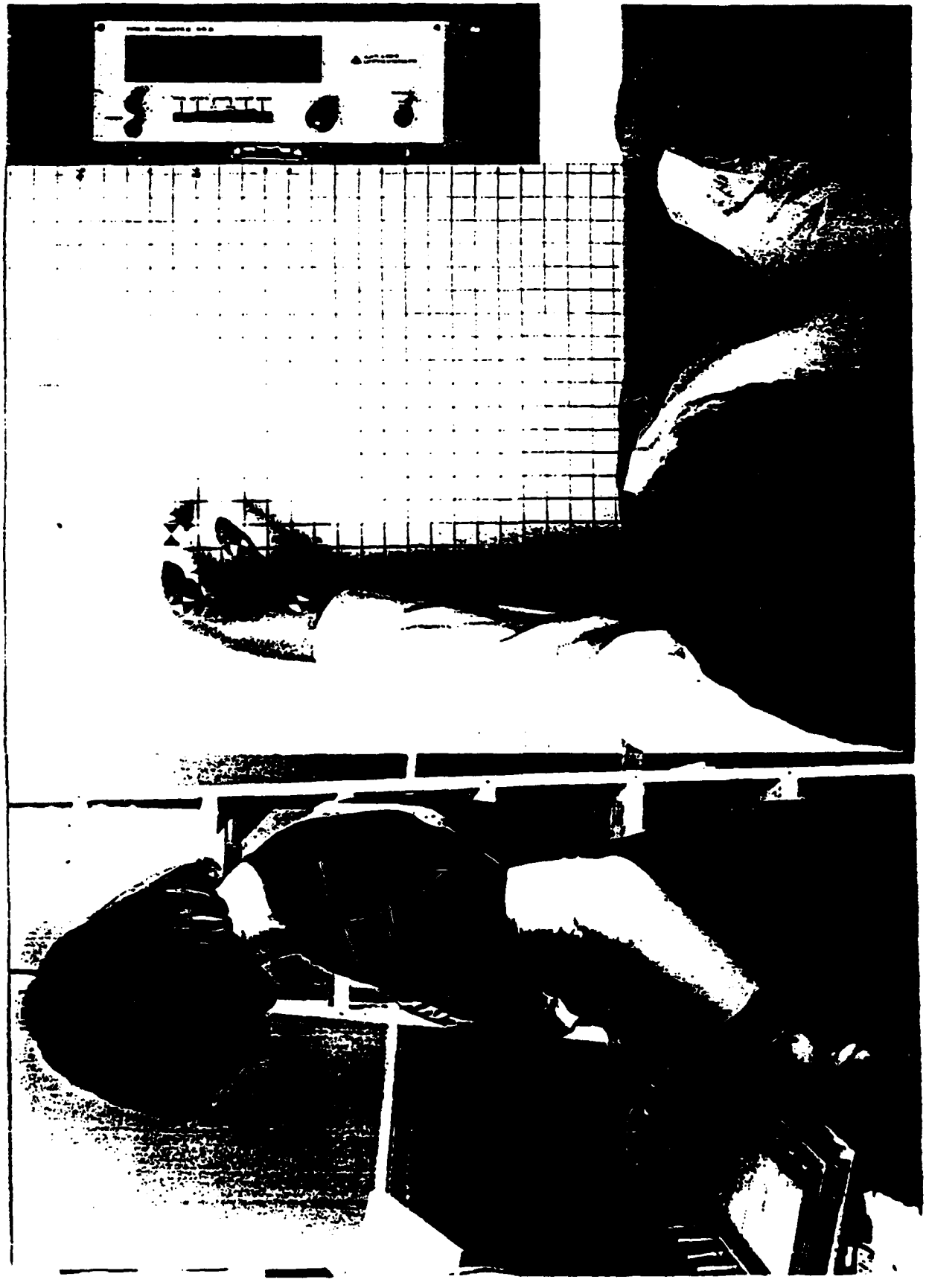
6. Illustrative trajectories from two single reaches by CY; one with the artificial hand, the other with the natural hand showing change of thumb and finger positions relative to wrist-dowel axis (in mms). Mean data for both narrow and wide objects over 4 trials are shown below.

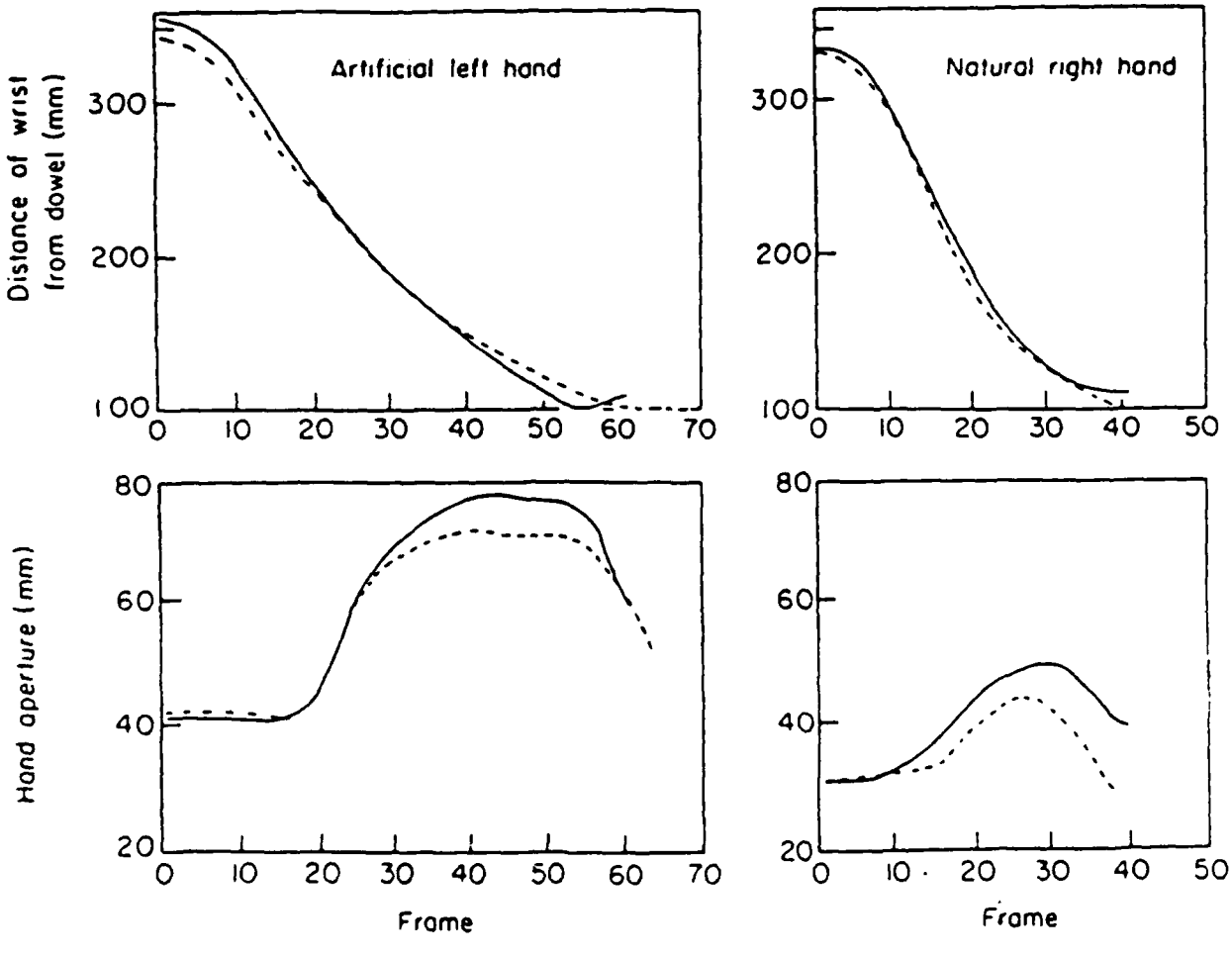
7. Two illustrative normal hand trajectories illustrating single-sided opening with palm to side facing midline so there is pad-to-pad opposition of finger and thumb (left) or with palm facing down so the thumb meets the lateral aspect of the finger (right). The upper 3 curves show (from the top) the home position, movement of the wrist, and movement of the target once it is picked up (after approx 1.5 seconds). The lower two curves show, relative to the wrist marker, the x-position of the thumb (above) and, either the tip (left) or the proximal IP joint (right) of the index finger.

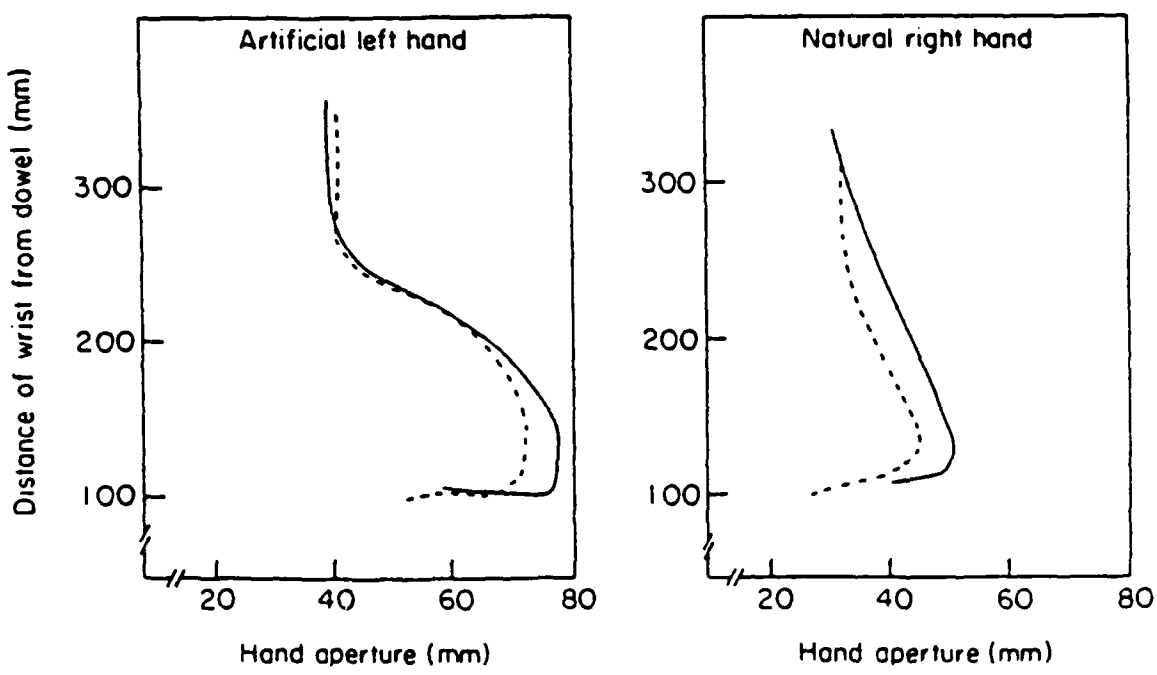

Video Frame Counter
Video Cassette Recorder
Video Frame Memory
Video Position Digitiser
Microprocessor
X/Y Plotter
Video Camera
Markers on thumb and index finger
Dowel with marker on top



Max aperture	80	93	116
Pre-contact SD	0.3	0.5	0.9

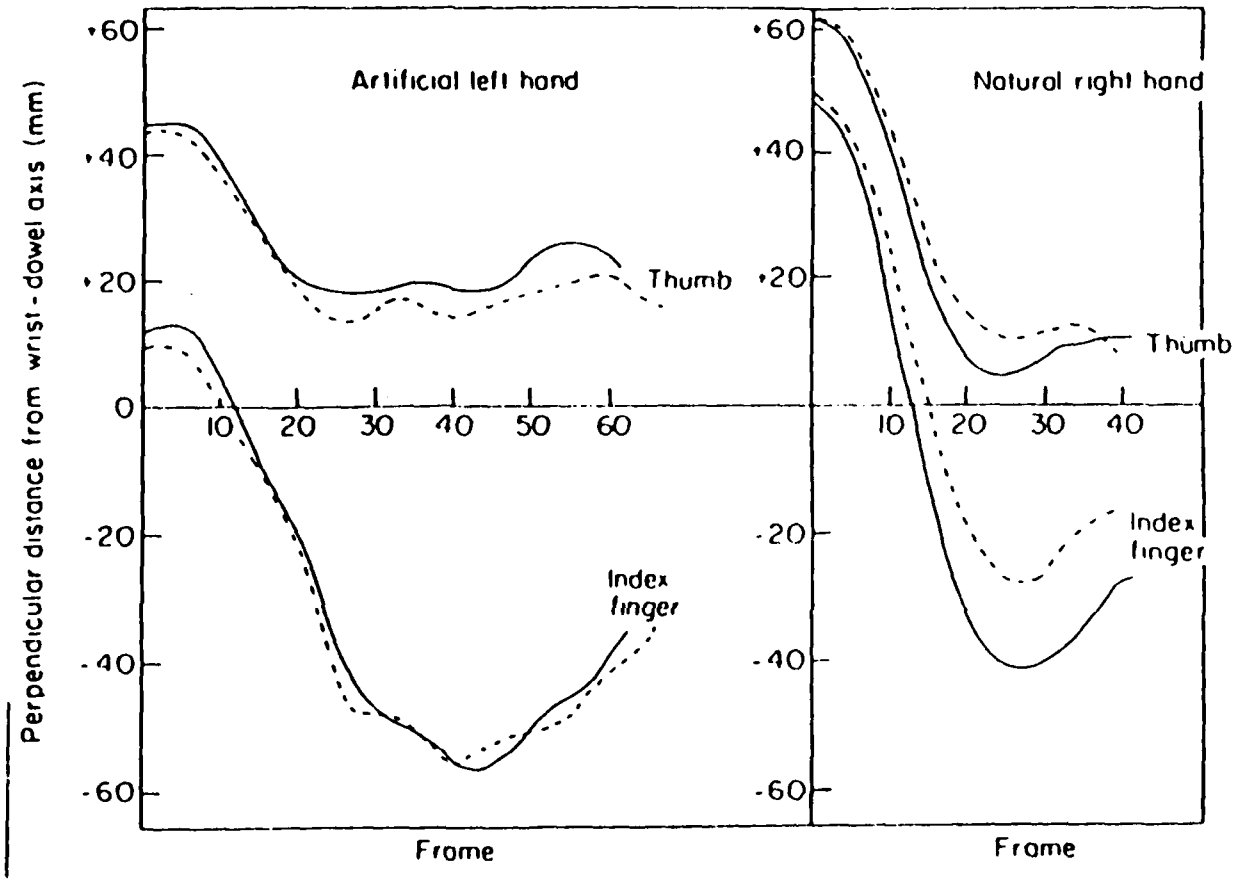



Artificial left hand
Natural right hand
Distance of wrist from dowel (mm)
Hand aperture (mm)
300
200
100
80
60
40
20
0
10
20
30
40
50
60
70
Frame

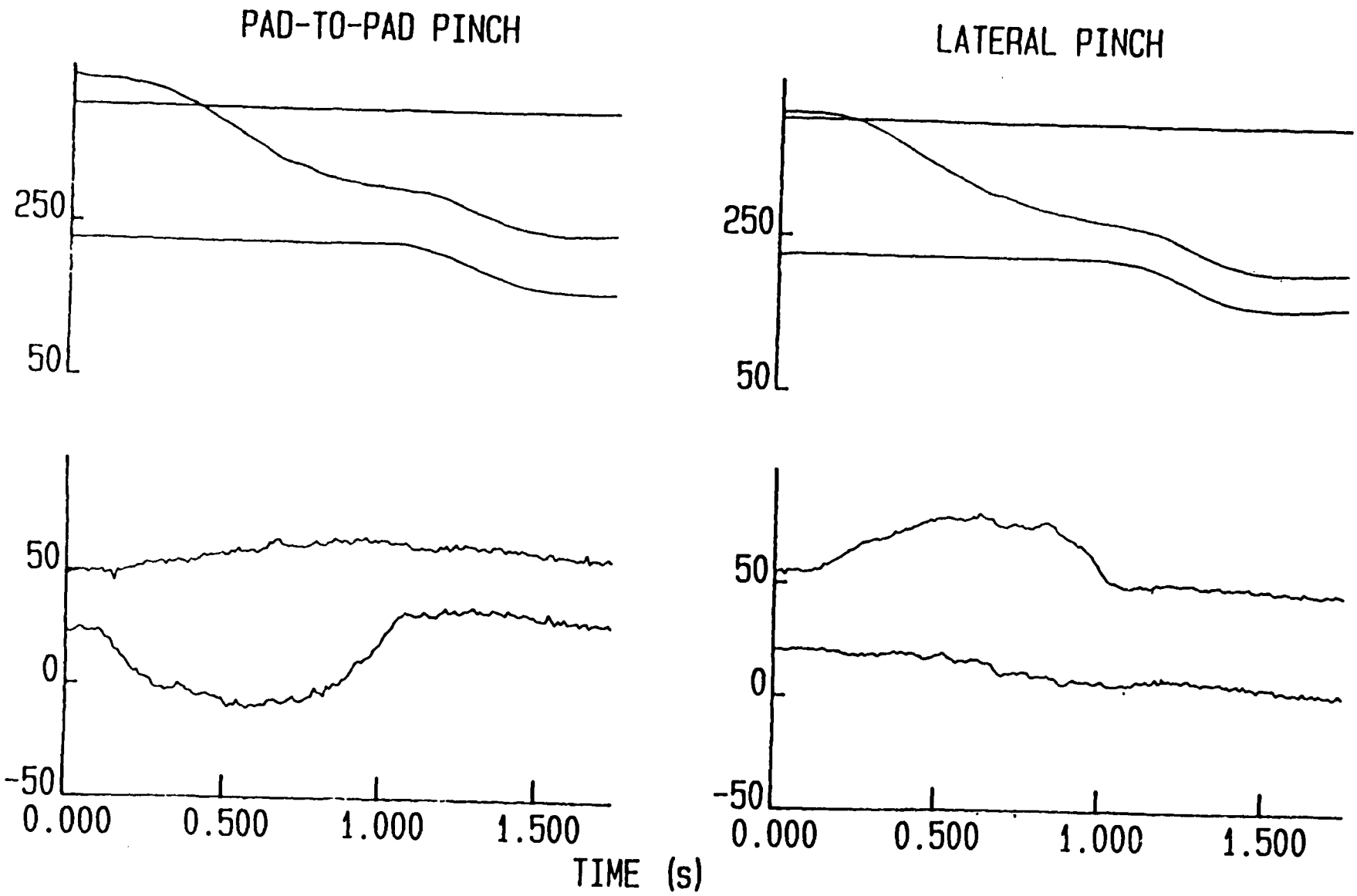

Artificial left hand
Natural right hand
Distance of wrist from dowel (mm)
300
200
100
20
40
60
80
Hand aperture (mm)
Hand aperture (mm)

Left

Right



Thumb	+4	-3
Finger	+26	+15


PAD-TO-PAD PINCH
LATERAL PINCH
250
50
250
50
50
0
-50
50
0
-50
0.000
0.500
1.000
1.500
0.000
0.500
1.000
1.500
TIME (s)

CHAPTER 2

SESSION II: HAND CONTROL ARCHITECTURES

A Task-Oriented Architecture for Dextrous Manipulation

S. T. Venkataraman
Robotics Lab.
University of Massachusetts
Amherst, MA 01003

D.M. Lyons
Dept. of Robotics & Flexibile Automation
Philips Labs
Briarcliff Manor, NY 10589

In order to embed a dextrous hand into a robotic assembly system, it is necessary to develop models of task plans that describe task resource demands on the hand, and models of their execution to meet with the task demands in reality. In this paper, we deal with an important issue in the realization of such models: the development of a task-oriented architecture for the decomposition of abstract task commands into primitive action units.

Our architecture is based on a hierarchy of *functional requirements for robot resources* and an appropriate set of *dextrous hand control strategies* that implement these requirements (based on the action primitives of [16,15]). We develop task plan models for dexterous hand usage and describe how robot resources (ie arms, wrists, hands) can be allocated to a task plan, and develop a set of *hand action primitives* that can be used to command a dexterous hand directly for task execution.

A Task-Oriented Architecture for Dextrous Manipulation

S. T. Venkataraman
Robotics Lab.
University of Massachusetts
Amherst, MA 01003

D.M. Lyons
Dept. of Robotics & Flexibile Automation
Philips Labs
Briarcliff Manor, NY 10589

1. Introduction

We define a *robot system* to consist of an arm and a *dexterous hand*, in addition to the controllers for these manipulators. A dextrous hand is commonly defined as an end-effector capable of imparting arbitrary *fine manipulations* to a grasped object. Clearly, such robot systems have tremendous applicability, especially in assembly automation. However, the research necessary to integrate them into the assembly process has shown little progress so far. In particular, dextrous hand research has shown a strong bottom-up trend, with little emphasis on developing appropriate abstract models of dextrous grasping and manipulation, and without these, a dextrous hand is far too complex a mechanism to integrate into a robotic assembly system.

In this paper, we develop an architecture for task-oriented dexterous manipulation. The architecture accepts object-level assembly operations (by which we mean, the operations are phrased in terms of how the objects move in response to the environment; this is also called *task-level*[5]), decomposes them, and outputs commands that are explicit to a particular robot system. Such an architecture allows the user to program in terms of object-level manipulation commands; a simpler task than programming hand and arm actions directly. In addition, it handles the allocation of task plans to *appropriate* robot resources (particular arms, wrists and hands), based on a statement of what the task plan requires from a robot system in order to achieve its goal.

In [17], we discussed how a task plan could be represented formally: "A *task plan* describes a well defined domain of interaction between the robot and its environment. A task plan contains a *goal* condition, a *control strategy* to achieve this goal, and an *applicability condition* specifying whether or not the control strategy remains appropriate for achieving the goal. The control strategy represents one possible way to achieve the goal, and is what is primarily characteristic about any given task." In that paper, our emphasis was on developing a strong notion of *task context* (the use of task-specific information) and on formally analysing the behavior of a task plan. In this paper, we push these definitions one step further by considering the *task criterion* associated with a task plan; the requirements of a task plan for specific robot resources necessary to achieve its goal.

We suggest a useful format for task critera, and construct an architecture in which task critera are used to allocate robots to executing task plans. We find that some problems, such as regrasping an object, can be naturally expressed by the concept of *reallocating resources* for a task plan when determined by continuous evaluation of the task criterion as the task plan is executing. In addition to considering the allocation of a physical robot to a task plan, we consider the allocation of an appropriate control model with which to use that robot. In particular, we describe a set of appropriate control models for a dextrous hand.

In section 2 we will describe the general structure of our architecture. We will then focus on dexterous hand usage and define hand task plan representation and hand models in greater detail. Section 3 contains discussions on the task criterion and execution of low-level hand task plans. (? Section 4 contains an example of the use of our architecture in carrying out a particular assembly operation. ?) We conclude in section 5 by outlining the current state of implementation of this system.

2. Task-Oriented Architecture

Our domain of application is robotic assembly. By task plan, we now mean one or more assembly operations, and we define an *assembly operation* to be the whole process starting from the acquisition of a component, its transport to a parts-mating site, and the application of an appropriate *parts-mating operation*. These three phases are strongly linked: for example, a component should be grasped to enable the subsequent parts-mating operation operation [12,2,6] and facilitate finding a path to the vicinity of the parts-mating operation [4]; and the transportation phase must bring the component into a suitable position from which to start the parts-mating operation. In turn, different strategies for the parts-mating operation may generate different sets of feasible grasps and transportation paths. Thus, although the phases of the assembly operation are distinct, they can never be decoupled from one another.

We model the architecture on the structure of assembly operations, rather than impose an ad-hoc form on the solution: The input to the architecture is a description of the parts-mating strategy in terms of how the component moves with respect to the environment. If a number of robotic arm-hand systems are available for the assembly operation, then, broadly, the choice of an arm is made for the reaching/transportation phase, while choice of a dexterous hand and appropriate control model, is made for the acquisition/manipulation phase.

The hierarchial control concept has been pioneered by Albus [3] and Saradis [14]. Our architecture is hierarchial, but differs from previous work in two aspects. The first is the strong link with the structure of the assembly operation — our architecture has components which map on each of the phases of the assembly operation. The second is our use of a functional hierarchy — each level in the hierarchy is a functional model of a robot (i.e., the model says only what the robot can do), which can be mapped (possibly dynamically) onto a specific robot system. The mapping constraint is that the robot system be able to provide that level of functionality. The architecture outputs

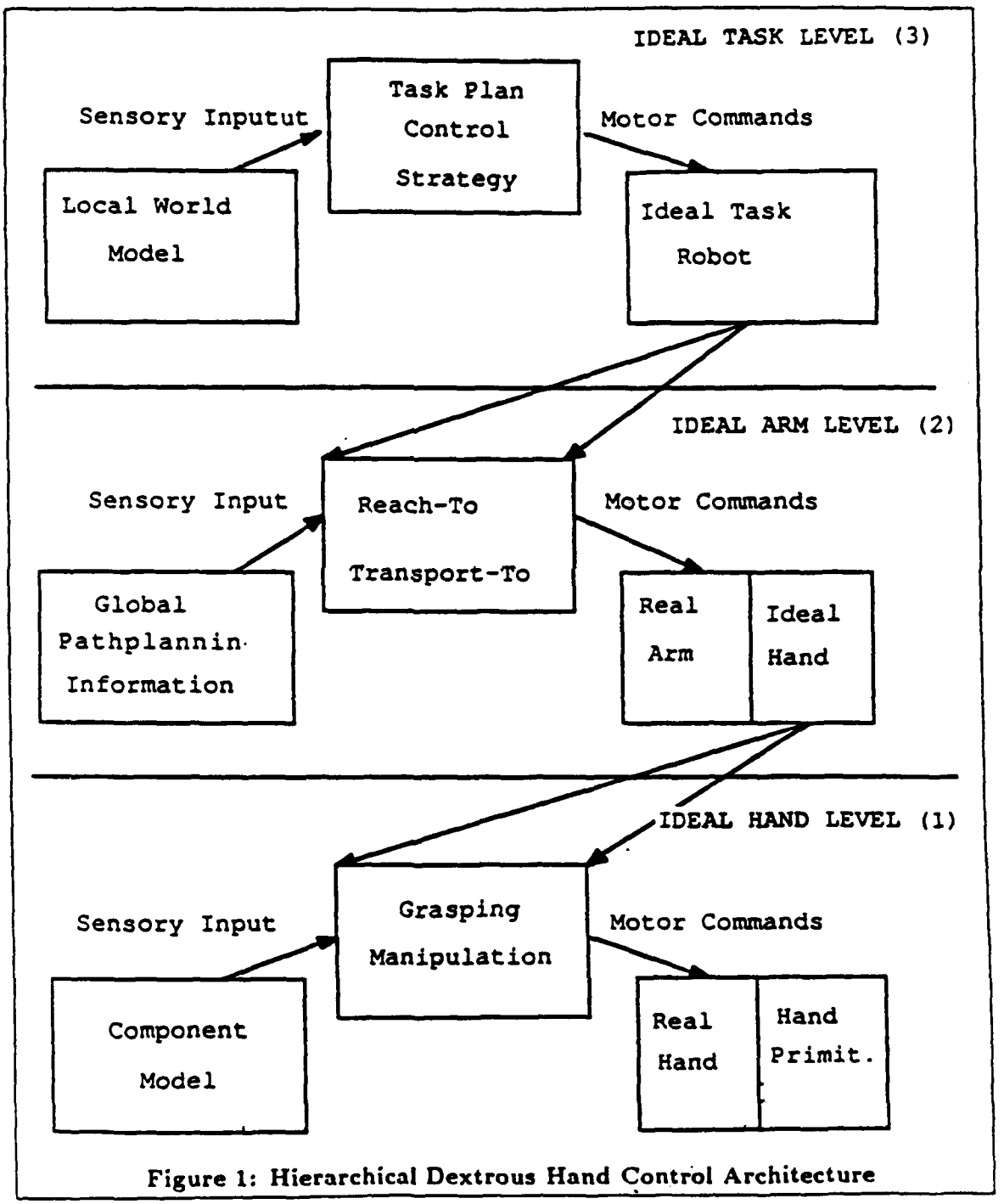


Figure 1: Hierarchical Dextrous Hand Control Architecture

low-level task commands particular to the robot system in use. The outputs are called *task primitives*. We concentrate on the outputs to the dextrous hand: Based on the control models in [16], the vocabulary of hand primitives considered are *preshape, grasping, acquisition* and *manipulation*.

Figure 1 describes the three levels of our architecture. At the topmost level, an object-level control strategy is the input. The control strategy describes one *particular* way to achieve the goal of the assembly operation. For example, the goal of a peg-in-hole-insertion might be to have the end of the peg against the end of the hole and a control strategy to achieve this goal might be to let the peg comply to reaction forces from the environment on all axis except the insertion axis. Note that a poor choice of control strategy may result in its failure to achieve its goal; in the peg-in-hole-insertion through compliance example, the nature of the local geometry may be one major determinant of potential success or failure [5].

The task criterion is constructed from the description of the assembly strategy (we discuss what should go into a task criterion in the next section), and is used to select an appropriate robot system on which to execute the task plan. The selection is made so that the arm may perform the transportation and the hand may perform object acquisition and manipulation. These two steps may be linked. We discuss some of these links within the context of implementing the architecture.

- Ideal Task Level:

  An task plan control strategy is developed for the assembly operation in terms of object motion in response to *its* local environment. The control strategy can be thought of as issuing commands to an *ideal task robot*, that is capable of moving the object directly; i.e. its configuration space is the space of control commands issued by the control strategy.

- Ideal Arm Level:

  At this level, movement of the object is realized as movement of a particular robot holding the object. Firstly, an arm[1] that has an appropriate *range* and *payload* to transport the hand (hand and object together) to the location of acquisition (parts-mating) is chosen. Once a robot arm has been chosen, the commands from the ideal task level are decomposed into robot arm commands for reaching and transportation and acquisition/manipulation commands to an *ideal hand*.

- Ideal Hand Level

  Based on the type of manipulation necessary, we choose a hand that has sufficient degrees of freedom, and additionally choose a control model for the hand. The object-level hand commands (to the ideal hand) are translated into motions of the physical hand's fingers while it maintains contact with the object that it grasps/manipulates. Hand-level task constraints arise from considerations such as grasp stability, manipulatibility and so on. We explain this in some more detail in the following section.

[1] We place a precedence on the choice of the arm because even if the hand (on the arm) is incapable of performing the required manipulations, the object could be transfered to the parts mating site, placed down and regrasped with another hand arm system (with the appropriate hand) before parts-mating.

We allow the choice of models at each levels to change as the task proceeds, subject to certain *consistency* constraints. Such changes may be called for by the robot arm-hand system in use becoming *non-optimal* in meeting with the task criterion constraints at some level. Thus, for example, if an arm-hand system is not available for the complete assembly operation, one may be chosen for the transportation phase. Upon completion of transportation, the choice of another arm-hand system may be optimal for the actual parts-mating operation. In this case, consistency demands either that the object be placed down for regrasping by the more optimal arm-hand system, or the exchange happen through some dual-arm-hand coordination scheme.

As explained in [16] and [15], each task primitive presents an action that needs to be performed by the robot arm/hand system. The actual execution of the command entails, first building (from the command) an internal representation of the task primitive for *task dependent control* of the arm/hand and developing appropriate dynamic models of the arm/hand and *its* environment together. [16] presents control architectures for the execution of hand task primitives on a dexterous hand and describes simple implementation results on the JPL/Stanford Hand. That paper also describes the actual controller(s) design procedure(s) for desired closed loop system behavior for each task primitive.

To conclude, in this section we have introduced the general structure of a task-oriented architecture for dexterous manipulation in which the responsibility of the transportation phase can be delegated to the robot arm and that of the acquisition/manipulation phase to the hand. We have considered *task primitives* as outputs of the architecture, and inputs into an appropriate task dependent control system for its execution on the actual robot arm/hand.

In the following sections, we describe our task plan representation in greater detail by focusing on two issues. We first explain the notion of task criteria and explain how it helps in choosing a robot system for the execution of the task plan. Then we explain in some detail the execution of hand task primitive commands on a dexterous hand. For examples of the development and analysis of task plan control strategies as well as a discussion about the application of task Invariant, see [17].

3. Task Criteria

The *task space* for a task plan is the space of control commands which it issues. We place no constrains on the structure of the task space, but for many assembly operations it will typically be motion and force in object-centered coordinates. Thus for example, a free motion task for a robot arm will be represented in a task space consisting of motion coordinates along available degrees of freedom. The task space for a task plan which issues 1-D position commands and open and close commands to a gripper might be $R \times \{open, close\}$. Alternatively you might consider the task space to be the configuration space of the ideal robot for the task.

A *task criterion* is a specification of what a task plan needs in order to execute on a robot system. Thus, the task criterion is strongly connected to the description of the task space. Nakamura et al. [13] deal with the issue of simultaneously executing several prioritized task plans on a single

redundant robot. In order to do this, they need to represent what each task demands of the robot. Their approach is to consider only the task space itself, R^{m_i} for some task plan i, as a definition of the task criterion.

Li and Sastry [12] describes a more complex task criterion for grasping. They are two (grasped) object-centered task spaces, one with *wrench* and one with *twist* coordinates. The task criterion in each is a *task ellipsoid*. For example, the task ellipsoid in the wrench space (consisting of force and torque coordinates $f_x, f_y, f_z, \tau_x, \tau_y, \tau_z$) is given by the inequality (α is a constant):

$$\frac{f_x}{r_1^2} + \frac{f_y}{r_2^2} + \frac{f_z}{r_3^2} + \frac{\tau_x}{r_4^2} + \frac{\tau_y}{r_5^2} + \frac{\tau_z}{r_6^2} \quad < \quad \alpha^2$$

In their work, the hand-object configruation which maximizes the task-elipsoid is the one most appropriate to execute the task plan on. This method extends the task criterion to include *ranges* on the axes. A limitation is that this method is strongly linked to the twist and wrench space descriptions, and so, generality (i.e., arbitrary nature) of task space is lost.

It is clear that we could contue to 'tack' components onto the task criterion, as necessary. However, the dange is that the task criterion would become unwieldy. A *grasp-based feature space* is used by Lyons [6,7] for grasp selection, in which the task criterion is simply a point. The feature space axes are the amount of precision movement necessary and the amount of grasp security necessary to complete the task (binary condition), and a coarse geometry of the object. The space is segmented into three regions, each corresponding to a steretype hand-object configuration denoted a *grasp* along with an appropriate simplified hand model that includes object *acquisition* and *manipulation* information.

The advantage of using a set of task-oriented models of the robot, is that it simplifies translations from task criterion to dextrous hand by considering the hand to be a set of simple models, each with designated characteristics. A disadvantage is that it forces task criteria to be expressed in task spaces chosen for stereotypical grasps rather than the task itself.

In summary, the task criterion should consist of at least the task space definition, but is made more versatile by considering aspects of the quality of control in the task space. The inherent complexity of the task criterion can be reduced by linking it with appropriate task-oriented models of the robot system.

The task space for task plan i is written R^{m_i}. Let us refer to the task criterion for this task plan as χ_i. Let $\mathcal{R}$ be the set of all available robot systems. We postulate a function $f_{i,r}$ which maps from the task space to the joint space of some robot system r. It is possible that $f_{i,r}$ is not defined for all $r \in \mathcal{R}$; e.g., if the task space contains a force axis, then f is undefined for a robot without control force, or if the task space contains 3 position axes, then f is not defined on a 2-D robot. Clearly, f should be part of χ_i.

Given that the task space can be mapped in some subset of $\mathcal{R}$, then the next questions concern the quality of control of the task space. We propose the following additional components for χ_i:

1. *Range*. The range on each axis of the task spaces is given by:

$$\rho : Z \longrightarrow R \times R \tag{1}$$

where $\rho(n) = [p, m]$ $n \in Z$ where $Z = 1, ..., m_i$ and $p, m \in R$. In words, this means that the map ρ takes an integer for each task axis and gives the corresponding axis' allowable interval. This allows us to set unilateral conditions (pushing an object) through either $p = 0$ or $m = 0$.

2. *Accuracy.* The accuracy required on each task space axis is given by:

$$\alpha : Z \longrightarrow R \tag{2}$$

A command of $r \in R^{m_i}$ to the robot, will result in the robot carrying out $r = (r_1 \pm \alpha(1), r_2 \pm \alpha(2), \ldots, r_m \pm \alpha_i(m))$.

Is is interesting to note that these components can provide information to decide if it's necessary to grasp the object (for some unilateral ρ requests, acquisition may not be necessary). Note that the outputs of a grasp planner like the one described in [?] are usually desired wrenches and twists. In the task space, accuracy measures may be combined with the task ellipsoid concept through specification of error ellipsoids in the wrench and twist spaces rather that ellipsoids of the actual wrench and twist values.

The other other pieces of information in a task criterion are:

1. The position of the component to be acquired, specified as a homogeneous transformation matrix, T_c, with respect to some base coordinate frame.

2. The locality of the subassembly or parts-mating site, T_s, again as a homogeneous transformation matrix.

These pieces of information will be used to determine the minimum workspace envelope necessary for an arm to carry out the reach and transportation phases of the assembly operation. The concept of the required payload capability should also enter into the task criterion. But we have not yet developed a good representation for it (other than simply adding it in as an extra component).

In summary, the task criterion for a single task plan i is defined as

$$\chi_i = (f, \rho, \alpha, T_c, T_s) \tag{3}$$

3.1 Task criteria in the Architecture

At the top level (ideal task), the task criterion is developed from our knowledge of how the object interacts with the sub-assembly. That is,

$$\chi = (f, \rho, \alpha, T_c, T_s) \tag{4}$$

where, ρ specifies the range required in the ideal task space from the ideal robot, α specifes the accuracy required and, T_c and T_s specify the transformation matrices for the object centered coordinate system and the sub-assembly centered coordinate system respectively.

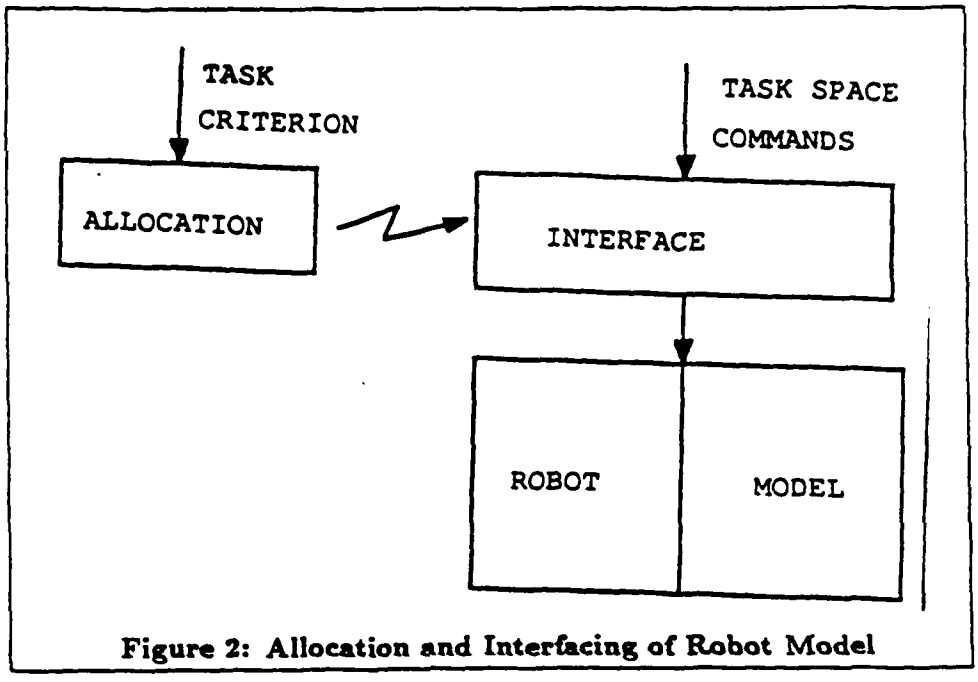


Figure 2: Allocation and Interfacing of Robot Model

At the next level (ideal arm), this criterion is broken up as follows: The task space is considered to be Cartesian space R^3, and an appropriate f_a, embodying the inverese kinematics for this robot, is assumed available[2]. The workspace envelope for an appropriate robot arm is constructed as some function of the range from the current position of the robot to the component, and from there to the sub-assembly

$$\rho_a = Func(\ RANGE(T_r, T_c) + RANGE(T_c, T_s)\ ) \tag{5}$$

where T_r is the current position of the robot.

At the next level (ideal hand), the choice of a parituclar hand model is subject to $\rho_h = \rho$ and $\alpha_h = \alpha$ being satisfied. Since we will consider a specific grasp on a dextrous hand is an example of a hand model, this allocation step *subsumes* grasp selection. Once a hand model has been chosen, commands issued to the ideal hand level are translated into the hand primitives (Section 4) for a particular physical (possibly dextrous) hand; this mapping is f_h.

The basic building block in implementing this architecture is shown in Figure 2. At each level, the motor actions are issued to what might be called a *virtual robot* (by analogy with [1]); that is, a combination of a physical robot plus some reference model.

For the three levels of abstraction in our task-oriented architecture, we have:

[2] If f_a does not exist for the arm under consideration, then clearly that arm cannot be a candidate for carrying out the transport and reaching phases.

Level 3: Ideal Task Level

- Allocation: $\chi = (\rho, \alpha, T_c, T_s)$
- Interface: R^m
- Virtual Robot: Ideal Task Robot

Level 2: Ideal Arm Level

- Allocation: $\chi_a = (f_a, \rho_a, \alpha_a)$
- Interface: reach-to, transport-to, R^{m_2}
- Virtual Robot: Actual Arm, Ideal Hand

Level 1: Ideal Hand Level

- Allocation: $\chi_h = (f_h, \rho_h, \alpha_h)$
- Interface: R^{m_h}
- Virtual Robot: Actual Hand

In the allocation of resources, we choose always a virtual robot model that *minimally* meets the task requirements. Secondly, we must allow *dynamic reallocation* of resources to account for the possibility of the present alllocation becoming *non-optimal*. Such re-allocations must be subject to *consistency* constraints of the type:

- Ideal Robot: Object must be placed down in workspace of new robot, which must now meet the full χ_i specification again.
- Ideal Hand: Object may need to be placed down, but only χ_h needs to be satisfied again.

4. Hand Control Models

We discuss the execution of task plans at level 1 (figure 1) on a dexterous hand. Shown in figure 3 is an architecture for dexterous hand control. It accepts low-level task plans and drives the dexterous hand directly. It consists of three levels: topmost denoted object-level, the middle, finger-level and lower level, control-level. Object-level task plans are decomposed into finger-level task plans, consisting of appropriate finger-level control strategies. Then, dynamic descriptions of fingers and the object (if applicable) together are selected. At lower levels, the actual design of controllers is performed for desired closed loop behaviour (subject to finger-level task criteria) and executed on the dexterous hand, [16].

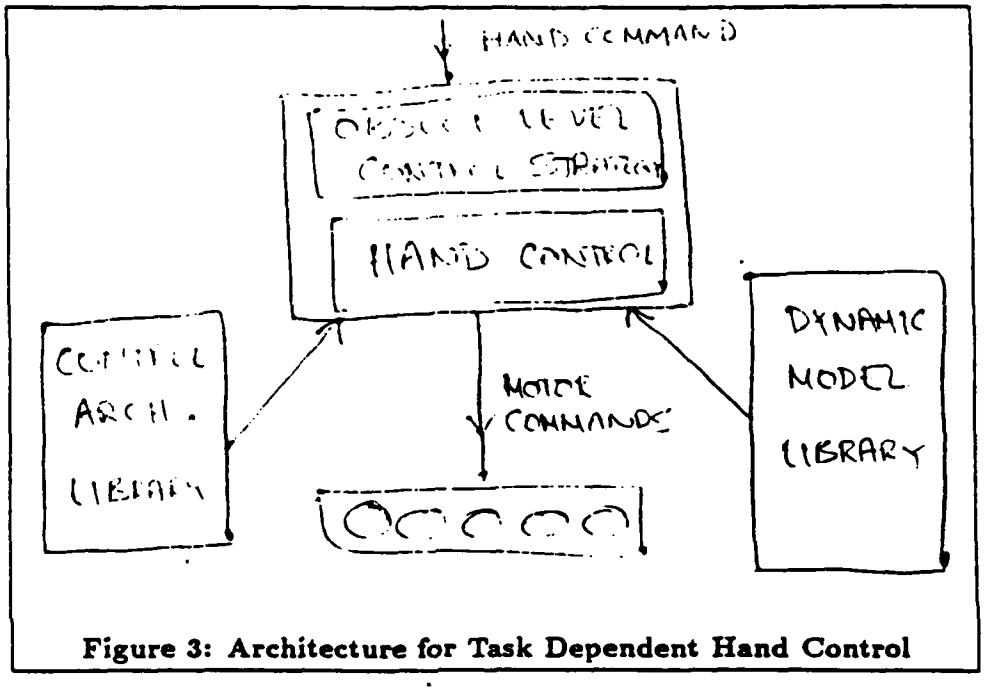


Figure 3: Architecture for Task Dependent Hand Control

In this section, we discuss two important issues in the context of task dependent dexterous hand control. First is the issue of a standard vocabulary of hand task primitives. Through such a vocabulary we standardize communications between the architectures in figure 1 and figure 3. Also, standard task primitive dependent control algorithms can be developed, stored and invoked. The second issue is a brief description of the actual task dependent control at the object and the finger levels. For details on hand control architectures, refer to [16].

Task plans output from level 1 of the task-oriented architecture in figure 1 describe desired low-level primitive *actions* of the hand, or hand and object together. Actions are specified through desired interactions between the hand and its environent, or the object (manipulated by the hand) and its environment respectively. Interactions, in general, occur as a result of mechanical contacts and can be described once the nature of contact is known. As shown in [15], interactions between any body and its environment can be classified into six *modes* (depending on the environment and contact); denoted *free, guarded* and *fine motion*, and, *free, guarded* and *fine force application*. Thus any low-level task plan must describe one of these six interaction modes.

A task space for each task primitive can be easily defined for the task primitives. For example,

1. Free Motion: consists of motion coordinates along allowable degrees of motion freedom.
2. Guarded Motion: consists of motion coordinates along allowable degrees of motion freedom, and the allowable degrees of force freedom that arise from contacts (established due to uncertainities in environmental models).

3. Fine Motion: consists of motion and force coordinates along allowable degrees of motion and force application respectively.

All interactions between the object and its environment can be specified in the vocabulary of the six task primitives. In addition, regardless of whether the hand holds an object within it or not, each finger's interaction with *its* environment can also be described in the task primitive vocabulary. Denote the whole hand's finger-level task plan as a ***hand task primitive***. Any hand task primitive, in general, is more than the collection of the corresponding finger task primitives. Hand actions can be classified based on ***functionality*** into five hand task primitives, [16]. We describe these below:

1. Motions of the hand in free space to configure the fingers around the object. We refer to this as a *Preshape*.
2. Motions of the configured fingers to close in towards the object until some pre-specified finger-object contact forces are established. We refer to this as a *Grasp*.
3. Upon actually making contact, certain contact forces must be applied to overcome the weight of the object. We refer to this as an *acquisition*.
4. Once stably acquired, changes in the states of the object can be imparted with the hand. If object actions refer to free, guarded or fine object motions, then we refer to the hand action as a *motion manipulation*. If the object actions call for free, guarded or fine force application, then we refer to the hand action as *force application manipulation*. This whole sub-class of hand actions is refered to as *manipulation*.

It is evident from the descriptions of hand task primitives that they are more than just a collection of finger task primitives in that they require, in addition, the representation of interactions between the fingers themselves. We consider these interactions in the context of two definitions. *Coordinated* finger interactions occur if all the fingers *work* towards a common hand level goal, [16]. *Synchronous* finger interactions occur if all fingers begin interacting simultaneously and finish interacting simultaneously, [16]. With this, we define preshape and grasp to consist of synchronous free and guarded finger motions respectively and, acquisition, motion manipulation and force application manipulation of coordinated guarded finger force application, coordinated finger fine motion and coordinated finger fine force application respectively, [16].

The hand task primitives, as before, will consist of four components: goal component, task strategy, task criterion and task Invariant. Let R_o^m denote the task space of the object and R_f^m, of all the fingers of the hand together. Let U_o and U_f represent *pseudo* control signals at the object and finger levels respectively, [16]. Let S_o, $\dot{W}_o$, S_f and $\dot{W}_f$ represent the object and finger level motions and forces respectively. The table below summarizes some of important points in the execution of some of the hand task primitives, [16].

	Preshape	Grasp	Acquisition	Free Obj. Mot. Mani
Object Level			$\dot{W}_o$	$S_o(t)$ $U_o = C_{s_o} \Delta S_o$
Hand Level	$S_f(t)$ $U_f = C_{s_f} \Delta S_f$	$S_{f_g}(t) = S_f(t) + C_{\dot{W}_f} \Delta \dot{W}_f$ $U_f = C_{s_f} \left(\Delta S_f + C_{\dot{W}_f} \Delta \dot{W}_f \right)$	$\dot{W}_{f_g}(t) = \dot{W}_f(t) + C_{s_f} \Delta S_f$ $U_f = C_{\dot{W}_f} \left(\Delta \dot{W}_f + C_{s_f} \Delta S_f \right)$	$\begin{bmatrix} S_f \\ \dot{W}_f \end{bmatrix}$ $U_f = \begin{bmatrix} C_{s_f} & \times \\ \times & C_{\dot{W}_f} \end{bmatrix}$

In the table above S_{f_g} and $\dot{W}_{f_g}$ represent mathematically guarded fingers' motions and fingers' force applications, $C_{(.)}\Delta(.)$ represents appropriate control signals. $C_{(.)}$ represents the controller chosen for the purpose and $\Delta(.)$ represents the corresponding error signal. Note also that the following relationships must be satisfied:

$$\begin{aligned} U_o &= \Omega \dot{W}_f \\ \dot{W}_{i_n} &> 0; \quad i = 1, 3 \end{aligned}$$

where, $\dot{W}_{i_n}$ represents the normal component of the contact forces applied by the i^{th} finger upon the object.

to be completed.

5. Example

The following example shows how our architecture can be used. The actual choice of assembly operation is not that important, since the majority of our architectural components are similar no matter what operation is involved. Our example is a compliant strategy for the place operation, defined as follows (see [17] for similar examples):

- Goal, G_p: Object surface D against surface W.
- Strategy, S_p: Move coarsely into position over the destination surface W (Figure 4). Move towards W complying to torques on x and y (in object coordinates), until some threshold force f_z is reached.
- Invariant, N_p: There should be no intervening objects between D and W.

An element of the task space $(R \cup \#)^6$ is $(\tau_x, \tau_y, f_z, p_x, p_y, p_z)$ where the τs are torques in object coordinates (see Figure 4), f_z is the z component of force and the ps are position components, again in object coordinates. The # element will denote we don't care what happens on that degree of freedom; allowing us to specify commands in a hybred position force paradigm.

Based on the task description, we can build χ_p as follows:

- α_p: The strategy hinges on good force and torque control, but the position accuracy can be coarse.

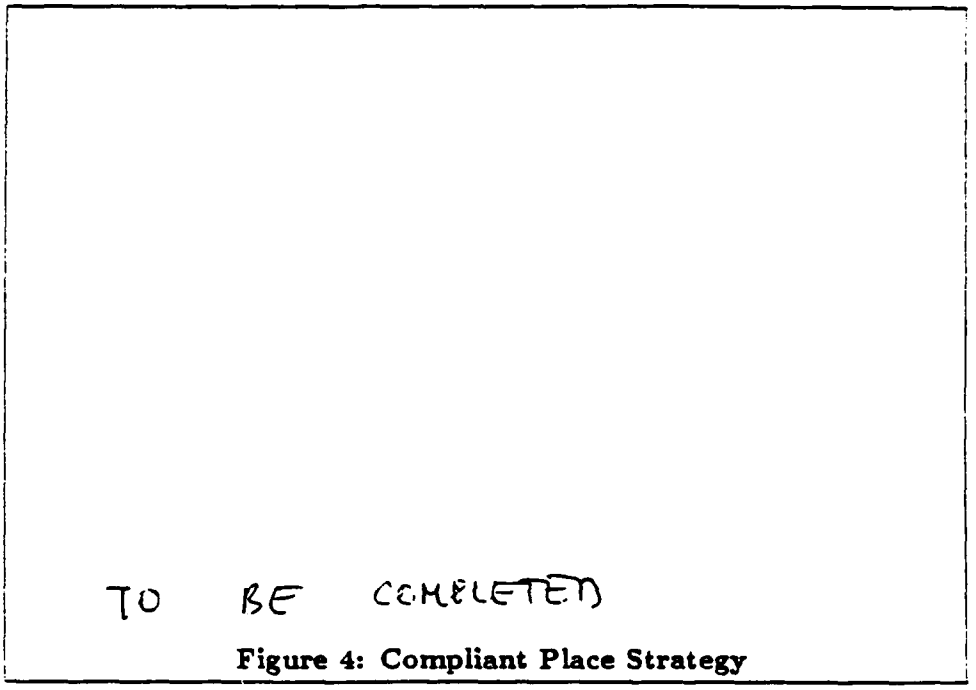


Figure 4: Compliant Place Strategy

- ρ_p: All our position ranges will be bidirectional, and of the order of the component size. The torque and force ranges should be small, and the z force range could be unidirectional.
- T_c and T_s are given by the particular situation.

Ideal Robot

The purpose of this level is to carry out the acquisition transportation phases of the assembly operation; to 'set the scene' for the place strategy. We have already: $\chi_a = (Range(T_r, T_c), Range(T_c, T_s))$. The first range is cruicial; you could ignore the second if you're willing to choose a new arm once the component has been acquired.

Ideal Hand

to be completed

6. Conclusion

In this paper, we have developed an architecture for dextrous hands that is based on the concept of using constraints from the description of the task to be executed to select an appropriate abstract model of the robot hand-arm system with which to execute the task plan. We see this work as a natural extension of our work on formal ways to represent the notion of *task context* (the use of task-specific information). We argue that our view of the task planning problem as

an allocation/interfacing of resources problem is a novel and useful way to use task information in dealing with complex robots, and to have a criterion for using robot resources well.

Implementation has been proceeeding in both of the main components of this paper: the execution of task plans with task criterion, and the construction of useful hand models, contain task-specific robot and environmental dynamics. The task plan representation work has been carried out using the $\mathcal{RS}$ model of computation as both a formal tool[8,9] and a multiprocessor implementation language[9]. The computational complexities of the architecture we have described are very high: especially the issue of remapping robot models. $\mathcal{RS}$ allows us to structure our plan representation as a network of concurrent cooperating computing agents (small grain processes) [11].

REFERENCES

[1] Arbib, M.A., Iberall, A., and Lyons, D., "Coordinated Control Programs for Movements of the Hand," *Exp. Brain Res. Suppl.*, 10, 1985, pp.111-129.

[2] Ahmad, S., Feddema, J.T., "Static Grip Selection for Robot-Based Automated Assembly Systems" *Journal of Robotics Systems* 4 (6), 1987, pp.687–717.

[3] Albus, J., MacLean, C., Barbera, A., and Fitzgerald, M., "Hierarchical Control for Robots in an Automated Factory," *Proceedings, 13th ISIR*, Chicago, Ill., Apr., 1983, pp.13.29–13.43.

[4] Lozano-Perez, T., "Robot Programming," *AI Lab Memo 698*, MIT, Cambridge, MA, Dec., 1982.

[5] Lozano-Perez, T., and Brooks, R., "An Approach to Automatic Robot Programming," *AI Memo 842*, MIT, Cambridge, MA, Apr., 1985.

[6] Lyons, D.M., "A Simple Set of Grasps for a Dextrous Hand," *Proceedings of the 1985 International Conference on Robotics and Automation*, St. Louis, MO, Mar. 25–28, 1985, pp.588–593.

[7] Lyons, D.M., "A Generalization of: A Simple Set of Grasps for A Dextrous Hand" *COINS Tech. Rep. 085-37* University of Massachusetts at Amherst, 1986.

[8] Lyons, D.M., "*RS*: A Formal Model of Distributed Computation for Sensory-Based Robot Control" *Ph.D. Dissertation* and *COINS Technical Report* # 86-43, University of Massachusetts at Amherst, Amherst, MA 01003, 1986.

[9] Lyons, D.M., "A Novel Approach to High-Level Robot Programming" *Proceedings, Workshop on Languages for Automation,* Vienna Austria, 1987.

[10] Lyons, D.M. "Implementing a Distributed Programming Environment for Task-Oriented Robot COntrol" *Philips Technical Note* TN-87-054, April 1987.

[11] Lyons, D.M., "The Task Criterion in Grasping and Manipulation" *Philips Technical Note TN-87-163* December, 1987.

[12] Li, Z., Hsu, P., Sastry, S., "On Grasping and Dynamic Coordination of MultiFingered Robot Hands" ERL Memo UCB/ERL M87/63, College of Engineering, University of California, Berkeley, CA 94720, September, 1987.

[13] Nakamura, Y., Hanafusa, H., Yoshikawa, T., "Task-Priority Based Redundancy Control of Robot Manipulators" *IJRR 6,2* Summer 1987, pp.3–15

[14] Saradis, G.N., "Intelligent Robotic Control" *IEEE Trans. on Automatic Control* Vol AC-28, No.5, May 1983, pp547-557.

[15] Venkataraman, S., "Dextrous Hand Control Through Impedence" *COINS Ph.D. Dissertation* Univ. of Massachusetts, Amherst, MA 01003.

[16] Venkataraman, S., "Task Representation in Robot Control: Part I – Theory" *Proceedings, IEEE SMC*, 1988.

[17] Vijaykumar, R., Venkataraman, S., Dakin, G., and Lyons, D., "A Task-Grammer Approach to the Structure and Analysis of Robot Programs" *IEEE Workshop on Languages for Automation* Aug. 24-27th, Vienna, Austria, 1987.

4.1 Control Architecture for the Belgrade II Hand

George Bekey
Computer Sci Dept
Univ of So Calif
Los Angeles, CA

Rajko Tomovic
Elect Eng Dept
Univ of Belgrade
Belgrade, Yugoslavia

Ilija Zeljkovic
Computer Sci Dept.
Univ of So Calif
Los Angeles, CA

ABSTRACT

The Belgrade-USC dextrous hand is an anthropomorphic manipulator, whose architecture is based on the principles of non-numerical or reflex control. Hence, the design emphasizes built-in synergies between motions and local autonomy, rather than maximizing flexibility. The resulting system is capable of performing a variety of grasping tasks, but is not well-suited for other applications which require a larger number of externally controllable degrees of freedom. The controller is knowledge-based, selecting a preshape for the hand on the basis of visual information. The finger motions leading to a stable grasp of objects of arbitrary shape (within allowable constraints on size and weight) are determined under local sensor control. The hand is mounted on the wrist of PUMA 560 robot.

Physical design Model I of the Belgrade-USC hand is approximately human-sized. It has five fingers. The thumb is rigid and rotates about an axis normal to the palm. The remaining four fingers are identical in size and possess 3 joints each. The motion of the finger segments is not individually controllable; they are connected by means of linkages in such a way as to display motions similar to those of human fingers during grasping; we term this relationship "internal synergy". Three motors mounted in the wrist structure provide the external degrees of freedom. One motor is used to position the thumb while the other move two fingers each. The finger drive is applied through a rocker arm designed in such a way that if the motion of one finger of the driven pair is inhibited, the second finger continues to move, thus achieving some shape adaptation without external control.

Sensors The basic pre-shape decisions are made using an external vision system (described elsewhere in this conference). The hand itself is equipped with three sets of sensors:

(1) Position sensing, to indicate the rotation of the finger base with respect to the palm

(2) Touch-pressure sensing in the fingertips, to detect contact with an object and the force being exerted

(3) Slippage sensing

Position and pressure sensing are currently being performed using pressure sensitive resistor materials. Slippage sensing has been implemented using thermistors to detect temperature changes; it is not completely satisfactory at the present time. A cooperative program with Lord Corporation is expected to lead to a significantly better sensor.

Preshape control Control of hand shape prior to grasping is obtained from a knowledge-based system. A data structure received from the vision system contains information on the location, orientation and geometry of the target object. The geometric information is combined with task information to produce the most desirable grasp mode configuration using a program implemented on a TI Explorer Lisp machine. The configuration is transferred to the hand control computer (an IBM PC/XT) using a serial line.

Finger controllers The d.c. finger drive motors are controlled using a PC/XT and a single-board controller. Position and force sensing signals are acquired using 16 A/D converters and fed back to the PC which implements a control algorithm to insure stability consistent with performance. The complete paper presents the frequency response of the finger-drive system and other performance characteristics.

Future developments Among the continuing developments are the following:

a. Redesign of the hand to allow for a jointed thumb and spreading of the fingers. It is possible that Model II of the Belgrade-USC hand will have only 4 fingers, rather than 5, each with its own motor.

b. Increasing intelligence in the hand shape control software, to allow the system to learn from unsuccessful grasp attempts.

c. Improved slippage sensing, to be incorporated in the finger pads as well as the palm.

d. Addition of proximity sensing, to give the hand even greater autonomy as it approaches target objects.

e. Implementation of the control hardware in a single chip, thus allowing for both motors and control electronics to be contained within the wrist structure.

Computational Architectures for Robot Hands

Sundar Narasimhan, David M. Siegel, John M. Hollerbach

MIT Artificial Intellegence Laboratory
545 Technology Sq., Cambridge, MA 02139

Abstract: This paper presents an overview of architectures used for controlling the Utah-MIT hand and other such complex robots. These robots are characterized by a number of joints and consequently demand powerful computer architectures to be controlled and utilized effectively. Emerging from our experience with such robots and control architectures are a few principles that we hope will guide other hand researchers.

1 Introduction

Computer architectures for robotics and indeed for other real-time systems have not received the attention they should. This is usually attributed to a number of reasons:

1. Present-day robots are simple, and hence do NOT require powerful computers to control them.
2. In industry, the constraints of economics often dictate the amount of dollars one has to spend on computers.

The first of these is not true for the kinds of robots that this workshop addresses. Robot hands that usually have nine to twenty-five degrees of freedom, are complex mechanical devices. For example, the Utah-MIT hand has 16 joints, each with 2 actuators. There are 32 tendon tension sensors, and 16 joint position encoders. Typically, a servo rate on the order of 400 hertz is required. That necessitates reading 19,200 sensor values per second and outputting 12,800 actuator values per second. If the operating system introduces anything but the most minimal overhead for servicing these rapid events, the desired update rates will not be met.

These robot hands, in their present stage of development, are to be found mostly in research laboratories. While attention must be paid to the economic soundness of an architecture, the need for computational horsepower must not be overlooked.

Coupled with the need for performance is the need for flexible software development environments in robotics. Most robots are simply too tedious to program – robot hands are even more so. While a number of efforts are under way to automate the task of a robot programmer, researchers need a good solution in the interim, when no such *task-level* programming languages are generally available.

Our earlier attempt at providing a standardized architecture for robotics was named the Muse (Narasimhan, et al. [1986]). Our second revised architecture has been termed the *CONDOR* (Narasimhan, et al. [1988]). In this paper, we present the lessons we have learned from our two attempts, which we hope will be of use to other researchers embarking on a new project with robot hands.

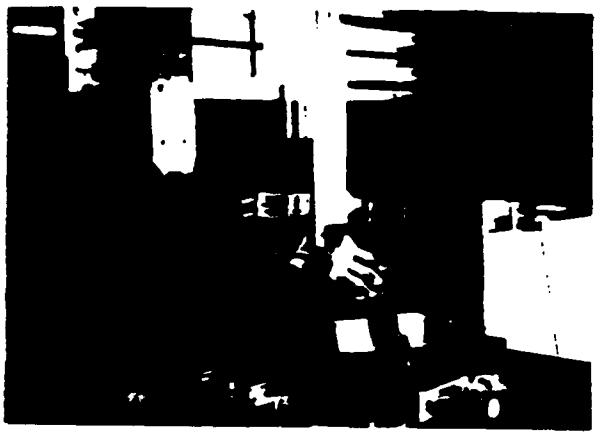


Figure 1: The Utah-MIT Hand Arm system

2 The Utah-MIT Hand Project - A brief update

The Center for Engineering Design at the University of Utah and the Artificial Intelligence Laboratory at the Massachusetts Institute of Technology have designed and developed a multi-fingered robot hand to pursue the achievement of advanced robot dexterity (Jacobsen et al. [1984]). There have been a series of papers published on different aspects of this project: on design and construction of the hand (Jacobsen et al. [1984] and [1986]) on low-level control issues (Biggers et al. [1986]), on algorithms for force control (Hollerbach et al. [1986], on tactile sensor design and construction (Siegel et al. [1987]), and on the computational architecture used to control the hand (Narasimhan et al [1986]).

The hand has now been mounted on a four degree of freedom x-y-z table (see Figure 1). Both the hand and the arm systems are now being controlled by a new control architecture labeled as the *CONDOR* real-time control system.

We have completed re-implementing the position control schemes that had originally been implemented on the MUSE on the new *CONDOR* system. We have also implemented the force controller based on the algorithm described in Hollerbach et al. [1986] on this system. The results from the initial experiments we have performed using the force controller indicate that the computational architecture presented in this paper is adequate to address problems of this nature.

3 Hardware

This section is organized as a set of simple guidelines. They are presented in a decreasing order of importance as we view them. Our guidelines may seem more like common sense on hindsight, but they are nevertheless a distillation of our experience with three versions

of our hardware and software configurations.

• *Do not build custom hardware unless you really want to.*

The reasons for this guideline are obvious. The amount of effort invested in building and maintaining custom hardware is often disproportionate compared with the increased performance they provide. Most robotics research laboratories are also staffed not by hardware designers, but by researchers whose primary focus is robotics. Hardware development is therefore best left to companies that specialize at it.

As applied to robot hands, which are characterized by a number of different joints (and consequently actuators), any custom hardware solution must be replicated to actually work. While modular designs are indeed possible, the increased number of components not only increases the cost of design and maintenance, it also increases the chances of failure.

• *Use simple, standardized components.*

If you do not use custom hardware, then you are left with playing the role of a system integrator. This narrows down the task, and one is left with merely choosing from amongst different commercially viable options. The array of board level products that are available today however is quite diverse, and making judicious choices to attain the performance you need at a price you can afford may not be entirely easy.

The need for *standardized* components cannot be overemphasized. Both versions of our controller for the Utah-MIT hand were built based on industry standard busses (Narasimhan et al. [1986], Narasimhan et al. [1988]). If we had not done this, it would have been difficult or impossible to find commercial vendors for the other peripherals we needed.

• *Use multi micro-processors but be conservative.*

The use of micro-processors in real time control systems is certainly not something new. There have been a few attempts to use a number of these powerful micro-procesors (Chen [1986], Gauthier et al. [1987], Kim et al. [1987]). While the Muse was based on the Motorola 68000 processor, the Condor is based on the Motorola 68020 processor coupled with the fast Motorola 68881 floating point unit. As we mentioned earlier, the computational task involved in controlling a robot like the Utah-MIT hand is quite burdensome for a single processor. Rather than having a monolithic single processor and program to control robot hands, we advocate using tightly-coupled systems.

The power of the individual microprocessor that is to be chosen is of some importance. Listed in Table. 1 are a few cpu's that we looked at before deciding on our current architecture. This list of processors is representative. One of the important considerations in choosing the hardware is the software support available for it. A commercially available fast processor is not really useful without a good, optimizing high level language compiler and development environment.

There are a couple of architectural options that we ruled out at the outset when we were putting together the second version of our system. We ruled out RISC architectures, partly because most commercially available cpu's still do not deliver fast floating point performance. Such architectures are also highly non-standard and usually require specialized compilers and development environments. We also ruled out processors that looked promising but were very much in the alpha or beta stages of their releases. This was mainly because in our first version of our system we ended up spending a lot of time debugging the

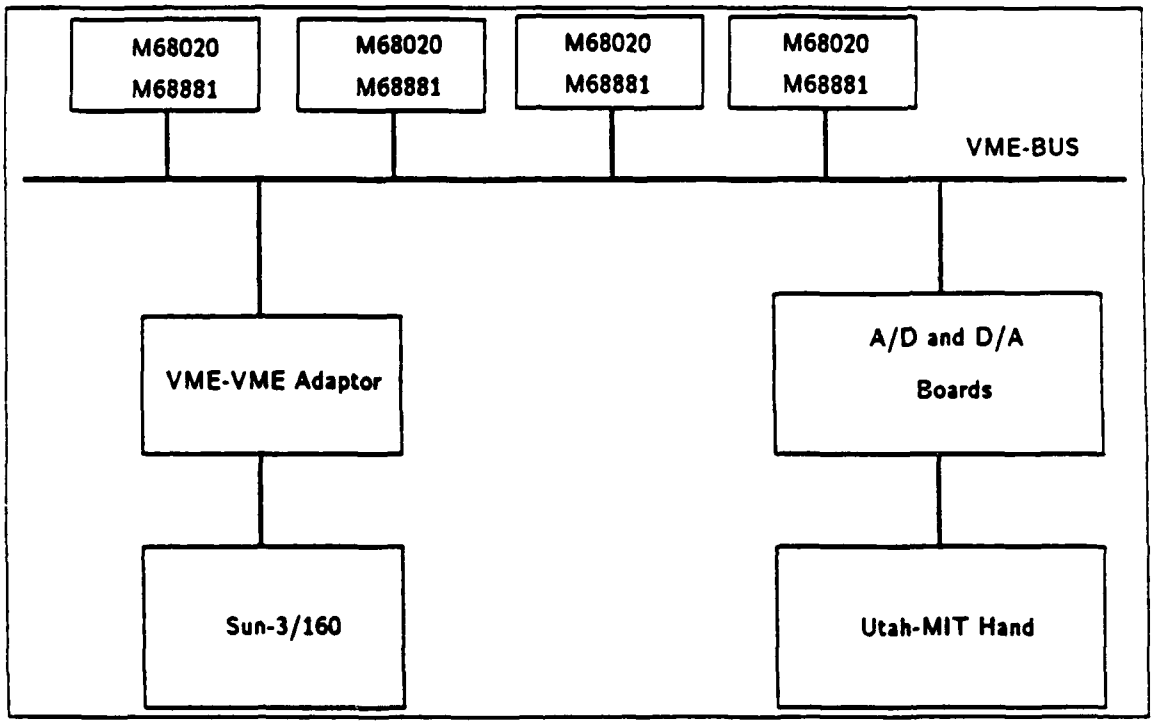


Figure 2: Version II hardware block diagram

Table 1: Comparisons of processing power available from alternative hardware configurations (July 87).

Processor Type	Speed	Cost	Comments
Microvax II	1 MIP	mod	interconnect problems
Vax 11/750	1 MIP	high	interconnect problems
Symbolics 3600	1 MIPS	high	lacks real time support
National 32032	1 MIPS	low	
Motorola 68000	1 MIPS	low	lacks floating point
Motorola 68020	2.5 MIPS	moderate	has fp co-processor
Motorola 68030	4 MIPS	high	unavailable

cpu board manufacturer's product rather than on robotics or hand control. Consequently, we learned to be conservative in our second attempt.

Tightly coupled systems facilitate easy software development. With dual ported memory, multi-processor programs can be written using the shared memory model. Robot hand control involves *coarse-grain* parallelism and this can be exploited by partitioning the tasks appropriately. When computational requirements increase, additional power can be obtained by simply adding more processor boards to the system and repartitioning the control algorithms.

• *Get floating point support if you can.*

Another realization that has become more obvious to us is that hardware support for floating point is useful in typical hand control applications. In the Muse, there was no such support. All arithmetic was performed in scaled integers. Although such implementations did demonstrate the feasibility of such schemes, the maintainability of the hand control software could be rated only as poor. Consequently, in the Condor, we chose the Motorola 68020 mainly to use its co-processor interface to the fast floating point Motorola 68881 chip.

• *Use a standard bus.*

The interconnect used in the first version of our system was the then standard Multibus I. Our second version is based on the VME-Bus. It must be mentioned that these industry-standard interconnects provide a very high bandwidth connection between the individual microprocessors, and with dual-ported memory shared memory communication becomes available for free. Peripheral boards like serial and parallel interfaces, a/d and d/a converters, additional memory boards are all readily available for such standard busses from a number of vendors.

It is true that such tightly-coupled schemes do not generalize beyond around ten microprocessors. Such schemes also do not work very well with cached systems, necessitating special solutions. However, we feel that other schemes for communication do not offer the bandwidth needed for the real-time computations that a robot hand typically requires.

The first version of our solution used a DMA connection between a VAX 11/750, which was our development host to the control microprocessors. This we found was too slow to do high bandwidth data transfers because of the setup overhead associated with each transfer, and because of the incompatible byte formats on the two machines.

Serial and parallel ports were not considered because of their low bandwidth, while we overruled devices like ethernet interconnections because of their complicated software requirements.

• *Choose your development host carefully.*

The choice of a development host ought to be dictated primarily by the software development environment that you wish to have. One important feature of the CONDOR is that it separates out, rather cleanly, the real time component, from the development environment. Obviously, if these two environments are compatible to a large extent then much is to be gained, since programs can be written and debugged on the development host and then run on the control microprocessors with small modifications.

Capabilities that a good development host ought to provide include high level language compilers for the real-time system, a good debugger, and bit-mapped graphics.

• *Be prepared to get your hands dirty.*
It would be nice if our guideline for using standardized, board level products resulted in a scenario wherein end users would need to have only a minimum knowledge of computer hardware. They could then concentrate on what they presumably want to do with the minimum of wasted effort. Unfortunately however, this is only ALMOST the case. Even the CONDOR requires the user to have some knowledge of the multiprocessor that he is writing his programs for. Debugging and maintaining the hardware has, on the other hand, become a board level operation for us. With the standardized components we use, trouble shooting is relatively easy.

To conclude this section on hardware, we would like to mention that the particular choice of components often seems to be dictated by intangibles or by variables that are hard to measure like availability of peripheral products, software compatibility, price etc. Our system whose block diagram is shown in Fig. 2, has worked surprisingly well, and is now being used to control the MIT Serial Link Direct Drive Asada Arm. There are a number of copies of this system being used at the A. I. Lab and in other research laboratories around the country to control an assorted variety of robots, which we hope will help our goal of standardizing robot control hardware.

4 Software

An important piece in any computational architecture is the software that is available to run on the hardware. The previous comments regarding flexibility and efficiency of robot control hardware apply equally as well to the software components. In this section we will provide an overview of what we feel are the innovative aspects of this part of the system.

Our approach to robot programming has been `library` oriented: at the lowest level are the system libraries that change relatively infrequently. Built on top of this is a set of libraries that are common across different robots; for example, routines to generate trajectories, low level kinematics calculations etc. Finally, come the robot specific libraries or user programs that we expect researchers to write for particular robots.

The *CONDOR* 's software development environment is almost entirely written in C.

The design goals of the software system were to:

(a) provide a flexible environment in which control programs can be written, debugged, and run,

(b) provide efficient and low overhead means of doing often repeated tasks,

(c) provide easy to use programmer libraries for dealing with data transfer hardware, and real-time interaction with robotic devices, and

(d) provide a graphics based user interface.

To achieve these goals, the system is structured around a few relatively simple organizing principles. In a typical program development scenario, the user is expected to write

and compile a program on the development Sun-3 host. Since the Sun runs the Unix operating system[1], the programmer has access to all the standard Unix software development tools. Once the program has been compiled, it is linked with the real-time library, and then downloaded onto the slave microprocessors for execution. The run-time environment provides the user with access to a number of program libraries for performing common tasks in a portable manner. For example, libraries are provided for control loop scheduling, file serving, and plotting data, to name just a few.

Thus, the *CONDOR* 's environment is really *two* different environments. First, it is an environment on the Sun known as the *development* environment, and then it is a *run-time* environment for both the Sun and the slave microprocessors. Much effort has been put into making the Sun and the slave microprocessor run-time environments as compatible as possible. In fact, most programs that run on the real-time controllers will run on the Sun simply by relinking them with the appropriate library.

The hardware architecture is tightly-coupled, and is a true MIMD machine. As such, it requires a program to be partitioned into segments that can run on multiple processors. Our approach to managing this multiplicity problem in terms of software engineering has been to largely ignore it. The number of processors in a *CONDOR* system rarely exceeds five or six and we do not expect this architecture to be applied to problems requiring more than a dozen processors. In essence, the course-grain parallelism applied to control programs is managed by the programmer directly.

4.1 Devices

Interacting with robots usually requires interfacing a controller to various input and output devices. Most robots have idiosyncratic front-end controllers, and the array of sensors connected to them is diverse. A computational architecture must support both easy hardware interconnections and easy software interface to these external devices. Since the *CONDOR* is based on the VME bus, hardware interfacing is straightforward. Software integration utilizes a device switch structure modeled after the system used by the Unix kernel.

The design of the device system was motivated partly by experience. In Version I of the system there was no systematic way of accessing devices. What existed was an ad-hoc interface between the Unix-style `read` and `write` calls and various device specific routines for the controller interface hardware. To use a parallel port board a user had to know it's particular initialization routine, and often needed to know such details as the device's control register address. In Version II of the *CONDOR* we replaced this with a clean design from the lowest level.

The *CONDOR* real-time environment was designed to provide an extensible way of writing device drivers. The low level details of a device, including its register formats and interrupt mechanisms, are abstracted from user code and hidden within the device driver. The trick was to keep the device drivers highly efficient, while providing the necessary level of abstraction to free user code of low level details. The added expense incurred by a

[1] Most of the system runs on Sun OS 3.2, which is closely compatible with and is based on Berkeley 4.2 BSD Unix.

standard operating system's device driver mechanism would not be acceptable in real-time controller environments for which the *CONDOR* is designed.

The *CONDOR* system's device mechanism is designed to be extremely fast and portable. The mechanism is static. No support for dynamic loading of device drivers is provided, which necessitates recompiling the system libraries each time a new hardware device is installed. The overhead associated with recompilation is significantly lower than the overhead and complexity associated with any dynamic loading scheme.

These mechanisms provide functions that:

(a) automatically initialize devices,

(b) handle interrupting devices,

(c) handle shared interrupt vectors,

(d) emulate system calls like **open**, **read** and **write**, and

(e) handle devices that may require more than the standard read and write style accesses.

Though the mechanism is modeled after the style found in early versions of Unix, there are a few significant differences. Each device is essentially modeled as an abstract data type, on which a few standard operations can be performed. When such a device is opened, an integer object called a **file descriptor** is returned whose semantics are close to that of the conventional Unix file descriptor. This object can be used as an argument to other operations that are performed on the device.

The standard file descriptor maps to a device structure of the following format:

```
struct devsw {
    char *dvname;
    int (*dvopen)();          /* open routine */
    int (*dvclose)();         /* close */
    int (*dvread)();          /* read */
    int (*dvwrite)();         /* write */
    int (*dvcntl)();          /* ioctl */
    int (*dvinit)();          /* init - probe */
    int (*dvputc)();          /* put a char */
    int (*dvgetc)();          /* get a char */
    int (*dvseek)();          /* seek */
    int (*dviint)();          /* input interrupt routine */
    int (*dvoint)();          /* output interrupt routine */
    char *dvdata;             /* device specific data */
    char *dvbuf;              /* device's buffer */
    int dvno;                 /* device's no */
    int *dvcsrs;              /* device csr array */
    int *dvvectors;           /* device's vector array */
    int dvnumvectors;         /* number vectors for a single device*/
```

This code fragment shows how a parallel port is opened and configured for further operation:

```
int
parallel_port_startup(board_number)
int board_number;
{
    int fd;
    if((fd = open(''':mpp'', board_number, 2)) < 0){
        printf(''Couldn't open device?\r\n'');
        exit(0);
    }

    /* Reset the board */
    mpp_reset(fd);

    /* Configure the board to be in raw 16-bit mode */
    mpp_config_16bit_raw(fd);

    /* return the fd, so that the user can use it later */
    return(fd);
}
```

As the above example illustrates, each device has an **open** routine and a **close** routine. There is one system-specific configuration file that tells the run-time system the types of devices that may exist. The *CONDOR* run-time system will, upon startup of each slave microprocessors determine which of those possible devices are actually present, and initialize them using their device specific **init** routines. This is analogous to the **probe** routine used by Unix systems.

Device independent operations like **open**, **read**, **write**, and **close** are mapped to device specific routines using the supplied file descriptor and the device switch mapping table. In addition, a device-specific buffer is allocated for each opened device, and contains data that is used internally by the device system. The lower level interrupt routines, which will be discussed later, operate on these device-specific buffers.

There are a number of operations performed on devices that do not easily fall into the Unix read and write paradigm. The standard Unix way of handling such unusual devices is to overload the **ioctl** system call. In the *CONDOR* system we chose to use the following conventions to deal with this problem, and in practice, this has proved effective:

1. Device specific routines are uniquely named by prefixing the routine with the name of the device (for example, a routine for configuring the **mpp** device will be called **mppconfigure**).

2. Routines that are peculiar to a device will take the file descriptor as the first argument and map it to a device specific structure. Once the mapping has been made the driver

can perform any necessary operations. This essentially provides entry points into the device driver through a back door, and bypasses the standard device structure.

The device driver interface also provides the low-level glue for the interrupt mechanisms, the file server interface, and the buffered input and output libraries.

4.1.1 Interrupts

Another capability that control system architectures require is servicing interrupts in real-time. These interrupts usually correspond to events that require attention, or periodic interrupts from timers.

The VME bus provides support for eight levels of prioritized vectored interrupts, and the Motorola 68020 processor has the capability to support 256 different interrupt vectors. Interrupts on the VME Bus are vectored, in contrast to the Multibus which typically supports non-vectored interrupts. The Multibus-II supports a different notion of interrupts which essentially increases the number of different levels of interrupt available on the bus, and is in some ways more desirable than the scheme supported by the VME bus. However, VME bus compatibility with the Sun-3 hardware was considered to be more important.

In real-time control, interrupts can come from a variety of sources: interval timers, analog to digital and digital to analog converters, parallel and serial devices, etc. A uniform way in which all interrupts are handled is indispensable in such a system. In Version I interrupts were handled in a rather ad-hoc fashion. In the *CONDOR* , all interrupt vectors map to the same higher level routine. The system has a software data structure that maps vector numbers to interrupt servicing routines. This data structure is used to map incoming interrupts to their appropriate servicing routines. This scheme has resulted in a single assembler routine that services all interrupts.

There are a few complications that the system must handle. A raw interrupt event must be mapped to a file descriptor corresponding to a device. Since the *CONDOR* is not multi-tasking, no distinction exists between system space and user space. When a serial port interrupts the system, the device generic interrupt handler must determine which serial driver's input buffer should receive the character.

The problem is further complicated by shared interrupt vectors, where multiple devices can interrupt the system with the same interrupt vector. The *CONDOR* system solves this complication by maintaining a list of all devices that receive interrupts on a particular vector. When more than one device uses the same interrupt vector, the *CONDOR* system maps this interrupt vector to a generalized *device-level interrupt* vector. This routine searches a list of interested devices and invokes the interrupt routine of each of those devices one by one, polling the possible choices.

4.2 Message Passing

While device drivers and other utilities provide support for bootstrapping and running a program on a single processor, the *CONDOR* message passing system addresses the issue of multiple processors and communication between them.

The message passing system provides a simple and low-overhead manner in which communication of data can occur between multiple processors, and between processes on the Sun. Since robotic control is always compute bound, a system for communication between such tasks has to be extremely time-efficient. The primary design goal of the *CONDOR* message passing system was therefore efficiency.

Interprocess communication for robotics, as mentioned by Gauthier et al. [1987], has been tackled in a variety of ways. Architectures based on RS-232 serial lines, rely on slow and primitive forms of communication; More recently, schemes based on parallel ports and LAN's have begun to appear. Shared memory based schemes are popular in tightly coupled systems like those described by Chen et al. [1986], while combinations of both shared memory and message passing have been used in others.

The present system is to a large extent, a redesign of the system described in Narasimhan et al. [1986]. Although the functionality provided by that system was far greater than that provided by the present version, the new scheme is much more efficient.

4.2.1 Messages

Since the Ironics processors and the Sun host computer are all bus masters on a common VME bus, each machine has access to each other's dual-ported memory. Interprocessor communication occurs over the bus and directly uses shared memory. This allows any processor to directly access data in another processor's memory. The most basic form of interprocessor communication possible would be direct memory reads and writes. Unfortunately, while this unrestricted access is highly efficient, it is hard to control.

To overcome the problems of unrestricted memory access, a mailbox-based message passing system is supported. Mailbox interrupts can be thought of as a software extension to the processor's hardware level interrupts. Another way of thinking about them conceptually is to regard mailbox numbers as port numbers that map to specific remote procedure calls.

A mailbox interrupt has a vector number and a handler routine. When a particular mailbox vector arrives, its appropriate handler is invoked. The handler is passed the processor number that initiated the mailbox interrupt and a one integer data value. This integer data value is the message's data[2].

The Version I message passing system was substantially more complex. Messages could be of arbitrary size, and they were addressed to virtual devices that corresponded to the mailbox handler routines in Version II. These handler routines were assigned to processors by a preprocessor that took as input an assignment file, that configured the routines available on each processor. The preprocessor then generated a routing table that had to be linked in with each program. The routing table mapped a virtual device number to a processor that could handle the function. The *CONDOR* redesign was done because the complexity and the overhead of the earlier system prevented most control programs from using message passing.

An important capability that the Version I implementation lacked was the ability to reply to messages. A program could not determine if a particular message succeeded or

[2]Integers are currently any 32 bit data quantity.

failed. Implementation of messages with replies could be done in the Version I system through an ad-hoc process, that was very inflexible. The *CONDOR* provides support for messages with or without replies. The implementation uses a reverse send from the recipient processor to the sending processor to acknowledge the receipt of a message. The implementation has been written carefully to be reentrant so that nested sends will indeed work correctly.

The following examples illustrate the operation of the message passing system. In the example, a message will be sent to a handler to read the value of a memory location. The location to be read is passed to the handler as the data portion of the message. The reply from the handler is the contents of that memory location. The code fragment for the handler would be:

```
simple_decoder(proc, data)
int proc;
int data;
{
    return(*(unsigned int *)data);
}
```

The handler is associated with a vector number using **mboxvectorset**:

```
mbox_set_vector(12, simple_decoder, ''A test handler'');
```

Now, any message sent to this processor for vector 12 will be handled by the **simpledecoder** handler.

Another processor can invoke the decoder by using the **mboxsend** routine. If the **simpledecoder** routine is available on processor 0 one can execute the following piece of code on any of the processors (including 0 itself) to invoke the service.

```
value = mbox_send_with_reply(0, 12, address);
```

This will cause the handler that corresponds to the number 12 to be invoked on processor 0 with the second argument being **address**. The call will not return until the other processor has responded with the value found at the given address. This call can be used to provide synchronization. For services that do not require synchronization, and hence do not return a value, the **mboxsend** call can be used.

In summary, the following are the key features of the message passing system:

(a) Since message sending happens asynchronously, the execution of a handler resembles an interrupt. All caveats that apply to interrupts and interrupt handlers also apply to message handlers.

(b) The base system is extensible in the sense that more complicated protocols can be built on top of it. For example, the underlying system does not support queueing of messages, although one can easily build one for mailboxes that require this.

(c) Since the message system is based on shared memory, sending long messages is usually handled by sending a pointer to the beginning of a long piece of data.

(d) Where efficiency is important, the message handling system can be used to set up pointers from one processor into another's memory. The processors can read and write this shared memory, without the minimal overhead of message passing.

Message sending and the invoking of message handlers is implemented using a *mailbox interrupt* which is a hardware interrupt that is invoked by writing into a particular memory location in a processor's memory. This hardware support is critical for the implementation's efficiency. To protect the integrity of certain critical data structures the *test-and-set* instruction is used. It is important that this instruction be supported truly indivisible by the hardware across the bus.

4.3 Support for Message Passing on the Sun

The Sun development host supports the same primitives for message passing available on the control microprocessors. Any number of processes on the Sun may communicate with each of the slave processors using the message passing protocol.

A message that arrives on the Sun must be mapped to a particular Unix process that can handle it. While each microprocessor is thought of as a single message-receiving processor, the Sun supports the notion of *virtual processors*. Each Sun process that receives interrupts is assigned a unique hardware interrupt vector. Each process on the Sun which participates in message passing must register itself with the Sun kernel, indicating which hardware interrupt vector corresponds to its messages. When a slave processor interrupts the Sun it does so using this vector. The Unix kernel traps on this interrupt vector and signals all processes that have expressed an interest in receiving the interrupt.

From the Sun, the *CONDOR* system maps the entire VME 24D32 space into the user address space of the control process (using the **mmap** system call). Memory references to any of the control processor's memory, or to the additional one megabyte memory board allocated in the VME backplane for Sun use, become simple array references. The PROCRAM macro returns a pointer to the beginning of memory for the particular processor. For example, to write a value to location 100 in processor 3's memory one would use the following code:

```
int *processor3_ram = (int *)(PROC_RAM(3));
processor3_ram[100] = value;
```

The PROCRAM macro is also used for programs running on controller processors to access memory of other Ironics processors. The code above would work, in fact, on any processor in the system.

Table 2 summarizes the performance of the message passing system as benchmarked by a variety of routines. As can be seen from this table, the performance of the message passing system is extremely fast between two control processors. The slower speed for messages sent from the control processors to the Sun host is caused by overhead present in the Unix timesharing operating system.

It should be noted that the message rates are not as high as servo rates. Consequently, messages are used only as signals to start and stop processes or control the flow of com-

Table 2: Performance of the Message Passing System

3

Type of Operation	Msecs/Message	Variation (Msecs)
Ironics to Sun	34	5
Ironics to Sun(R)	38	10
Sun to Ironics	3.9	0
Sun to Ironics(R)	4	0
Ironics to Ironics	0.2	0
Ironics to Ironics(R)	0.25	0

putation and are not used as timing pulses. Servo communication is done using shared memory structures that are initialized by messages.

4.4 Message Passing and its Implication for Control

Using the facilities provided by the message passing system it is possible to treat a hierarchical controller as an object-oriented system that responds to control messages. For example, a low-level joint PID controller could be described as an object that responds to two different kinds of messages: messages that alter internal parameters like gains and position set points, and messages that control the execution of the servo loops. Using this scheme, any processor, including the Sun host, can control the execution of any servo loop in the system.

4.5 Higher Level Protocols

The *CONDOR* system requires the concurrent operation of several control microprocessors to perform its tasks. Programming such a complex MIMD machine would certainly be a nightmare were it not for the numerous services that were built on top of the base system using the message passing facilities. These services are flexible interaction with a number of slave microprocessors at a time, provide file server capabilities on the development host, and symbolic debugging.

4.5.1 Debugging

One of the most important utilities in the system is the symbolic debugger. To implement a debugger in a flexible manner it was decided to emulate the Unix ptrace system call. Ptrace is used by Unix debuggers to examine registers, set breakpoints, and control program execution of a slave process. Our emulated ptrace is linked into all control programs as part of the standard library.

The emulated ptrace communicates with the Sun-3 development host using the message passing system. For example, if the Sun-3 host wishes to examine a control program's registers, it would send a message down to the processor's ptrace handler. The ptrace

handler would reply with the desired information. So, the debugging program runs on the host Sun computer. Only the low level ptrace routine runs on the microprocessors.

By emulating the low level ptrace call, almost all Unix based debuggers can be adapted, with little modification, to debug programs on the controller microprocessors. Initially, the Gnu Debugger (GDB) is being used. It provides fully symbolic, C source code level debugging tools. The debugger not only helps debug higher level application programs, but has also been used to find low level system bugs.

The debugger can be used in 3 basic modes. The first allows the user to start executing a program directly under the debugger's control. This mode would be used when a bug is actively being tracked down, and setting initial break points might be necessesary. The second mode allows the debugger to be attached to a processor after execution has begun. For example, if a program were in an infinite loop, the debugger could be attached to the running program to determine what went wrong. Finally, if a program receives a fatal exception, for example an addressing error occurred, the debugger can be attached after the error occurred to help analyze the problem.

Once the debugger has attached to a process on the slave control processor, all the capabilities of the debugger can be used to debug the program running on the remote host, just as if it were running on the development host.

4.5.2 File Serving

Control programs may need access to data stored in files, for example. To allow these types of access, a file server protocol, built using the low level message passing system, is provided. From a control program point of view, the standard Unix file operations are available, both buffered and unbuffered. Typically, each file operation results in a message being sent to the Sun, where a server process performs the equivalent Unix file operation. The result is then sent back to the microprocessor as a reply.

To make the file server efficient, shared memory is extensively used to avoid copying data. When a microprocessor performs a read or a write, the file server process running on the Sun reads or writes the data directly to or from the microprocessor's memory. No intermediate data copy is required. To perform a read, for example, the microprocessor would send a message to the Sun giving the address to read the data into, and the number of bytes to read. The data is directly transferred into the processors buffer.

The file server is designed to operate in a stateless manner. All data necessary is stored on the microprocessor. The Sun server does cache some information, but if necessary, it can request the information from the microprocessor. So, if the Sun file server process is terminated and restarted, the microprocessor can continue to file serve without problems.

4.5.3 Virtual Terminals

Many microprocessors are used in a *CONDOR* system. The Sun-3 host computer provides a window-system based interface to access these computers. One window for terminal input and output to each computer is provided. A virtual terminal protocol, built using

the message passing system, routes data between the Sun and the microprocessors. This avoids the bank of terminals that would otherwise be connected to the real-time controller.

The virtual terminal protocol works in both directions. The Sun can send a message to a microprocessor that contains an input character. The character is added to the terminal input queue by the virtual terminal message handler. Each character is sent in one message, since transfers in this direction are limited by the rate at which a person can type.

Terminal output from a microprocessor to a Sun virtual terminal window is sent in blocks, not one character at a time. This is done purely for efficiency purposes. The output message contains a pointer into the terminal driver's memory, and the number of characters to output.

4.6 The Condor User Interface

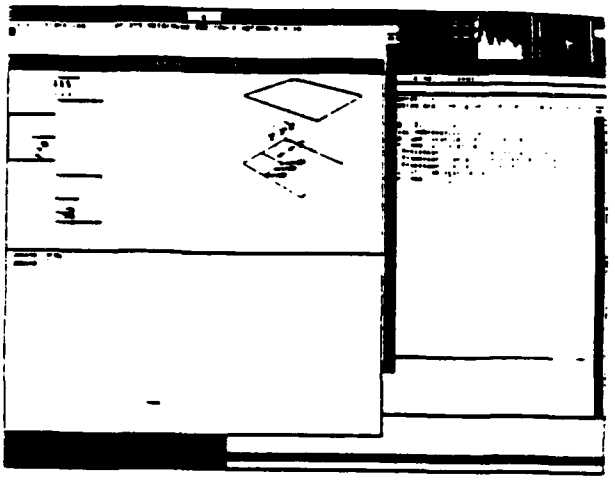


Figure 3: The Condor running under the X Window System

The file server, debugger and virtual terminals combined together form the *CONDOR* user interface. This program is an X-window system based application that programmers utilize to interact with the slave microprocessors. The user interface provides one virtual terminal for each slave processor, it runs a file server, and it can start debuggers. Figure 3 shows the screen of a typical *CONDOR* user interface session.

5 Conclusion

In this paper, we have presented a multi-microprocessor system architecture that has been designed explicitly for research in robotics. The hardware and software systems have been designed to provide flexibility and ease of maintenance as well as performance. Our

goals were achieved by decoupling the real-time component of our system from the development environment and making sure that each system was highly optimized for its task. We believe that the *CONDOR* will form the basis for a new type of standardized robot controller that will become common in research laboratories. Such a common architecture will then enable a sharing of work and duplication of results that has not been possible until now.

6 References

1. **Biggers, K. B., Gerpheide, G. E., Jacobsen, S. C.,** "Low-level control of the Utah-MIT dexterous hand," *IEEE Conference on Robotics and Automation*, pp. 61-66, San Francisco, April 1986.

2. **Chen, J. B., Fearing, R. S., Armstrong, B. S. and Burdick, J. W.,** "NYMPH: A Multiprocessor for Manipulation Applications," *Proc. IEEE International Conference on Robotics and Automation*, pp. 1731-1736, San Francisco, April 1986.

3. **Gauthier, D., Freedman, P., Carayannis, G., and Malowany, A. S.,** "Interprocess Communication for Distributed Robotics," *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 6, pp. 493-504, 1987.

4. **Hollerbach, J. M., Narasimhan, S., Wood, J. E.,** "Finger force computation without the Grip Jacobian," *IEEE Conference on Robotics and Automation*, pp. 871-875, San Francisco, April 1986.

5. **Jacobsen, S. C., Wood, J. E., Knutti, D. F., Biggers, K. B.,** "The Utah/MIT Dextrous Hand: Work in Progress," *Robotics Research*, MIT Press, pp. 601-653, Cambridge, MA, 1984.

6. **Narasimhan, S., Siegel, D.M., Hollerbach, J.M., Biggers, K.B., Gerpheide, G.E.,** "Implementation of Control Methodologies on the Computational Architecture for the Utah/MIT Hand," *IEEE Conference on Robotics and Automation*, pp. 1884-1889, San Francisco, April 1986.

7. **Kim, J. J., Blythe, D. R., Penny, D. A., Goldenberg, A. A.,** "Computer Architecture and Low Level Control of the Puma/RAL Hand System: Work in Progress," pp. 1590-1594, *IEEE Conference on Robotics and Automation*, Raleigh, NC, April 1987.

8. **Lozano-Perez, T., Brooks, R.** "An approach to Automatic Robot Programming," *Artificial Intelligence Laboratory Memo AIM 842*, MIT Artificial Intelligence Laboratory, April 1985.

9. **Salisbury, K.,** "Kinematic and force analysis of Articulated Hands," *Ph. D. Thesis*, Department of Mechanical Engineering, Stanford University, July 1982.

10. **Siegel, D.M., Narasimhan, S., Hollerbach, J.M., Kriegman, D., Gerpheide, G.**, "Computational Architecture for the Utah-MIT Hand," *IEEE Conference on Robotics and Automation*, pp. 918-925, St. Louis, March 1985.

11. **Siegel, D.M., Garabieta, I., Hollerbach, J.M.**, "A capacitive based tactile sensor," *Proc. of the SPIE Conference on Intelligence Robots and Computer Vision*, pp. 153-161, Cambridge, MA, September, 1985.

CHAPTER 3

SESSION III: ROBOTIC HANDS & THEIR USAGE

Dextrous Robot Hands: Several Important Issues

Zexiang Li and Shankar Sastry

Electronics Research Laboratory
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley. Ca.94720

Abstract

In this paper, we disccuss several important issues related to the study of dextrous robot hands: (1) grasp planning, (2) the determination of coordinated control laws with point contact models, and (3) redundancy resolution. We develop dual notions of grasp stability and manipulability and use these notions to formulate grasp quality measures. Based on the point contact models, we develop a "computed-torque" like control algorithm for the coordinating manipulation by a multifingered robot hand. The control algorithm which takes into account both the dynamics of the object and the fingers is shown to be exponentially stable. Finally, we define the notions of redundant grasps and redundant hands, and formulate the corresponding redundancy resolution problems.

Table of Contents

1. Introduction:

A new avenue of progress in the area of robotics is the use of a multifingered robot hand for fine motion manipulation. The versatility of robot hands accrues from the fact that fine motion manipulation can be accomplished through relatively fast and small motions of the fingers and from the fact that they can be used on a wide variety of different objects (obviating the need for a large stockpile of custom end effectors). Several articulated hands such as the JPL/Stanford hand [10], the Utah/MIT hand [19] have recently been developed to explore problems relating to grasping and manipulation of objects. It is of interest to note that the coordinated action of multiple robots in a single manufacturing cell may be treated in the same framework as a multifingered hand.

Grasping and manipulation of objects by a multifingered robot hand is more complicated than the manipulation of an object rigidly attached to the end of a six-axis robotic arm for two reasons: the kinematic relations between the finger joint motion and the object motion are complicated, and the hand has to firmly grasp the object during its motion.

The majority of the literature in multifingered hands has dealt with kinematic design of hands and the automatic generation of stable grasping configurations as also with the use of task requirement as a criterion for choosing grasps (see for example the references [1 ~ 4], [6 ~ 8], [10], [13 ~ 18]). Some of these references ([2,3,6,12,14,16]) have suggested the use of a task specification as a criterion for choosing a grasp, albeit in a some what preliminary form. A few control schemes for the coordination of a multifingered robot hand or a multiple robot system have been proposed in ([8, 23 ~ 26]). The most developed scheme is the master-slave methodology ([23,24]) for a two-manipulator system. The schemes developed so far all suffer from the drawback that they either assume rigid attachment of the fingertips to the object or are open loop. The schemes do not account for an appropriate contact model between the fingertips and the object.

This paper treats three fundamental problems in the kinematics and control of multifingered hands: *grasp planning, the determination of coordinated control laws with point contact models and redundancy resolution.* We develop dual notions of grasp stability and grasp manipulability and propose a simple procedure for task modeling. Using the task we then define the structured (or task-oriented) grasp quality measures, which are subsequently used for devising a grasp planning algorithm. We give a basic control law for the coordinated control of a multifingered robot hand manipulating an object, which takes into account both the dynamics of the object and the fingers and assumes a point contact model. Furthermore, we show that the basic control law can be easily extended to cases when the fingertips roll on the object (i.e., rolling motion). Finally, when a robot hand has more degrees of freedom than needed to acomplish the

main goal, we show how to use these redundancy to acheve additional objectives, such as collision avoidance, maximizing manipulability measures, or stability measures.

A brief outline of the paper is as follows:

In section 2, we define the grasp map and its associated effective force domain, and the hand Jacobian. We develop dual generalized force and velocity transformation formulae relating the finger joint torques and velocities to the generalized force on and generalized velocity of the body being manipulated. Using these relations we define stability and manipulability of a grasp. In section 3, we extend our previous work in [2] to define task oriented measures for grasp stability and manipulability. In section 4, we use the machinery in sections 2 and 3 to develop a new "computed torque-like" control scheme for the dynamic coordination of the multifingered robot hand, along with a proof of its convergence. In section 5, we formulate the problem of redundancy resolution for a redundant robot hand.

2. Kinematics of Multifingered Robot Hands.

In this section, we discuss a few concepts concerning rigid body motion and the kinematics of a multifingered robot hand. These subjects are discussed in greater depth in, for example, [1, 2, 4, 5, 21] and the texts [20] and [22].

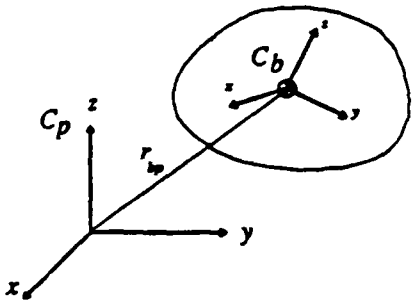


Figure 1. A rigid body in R .[3]

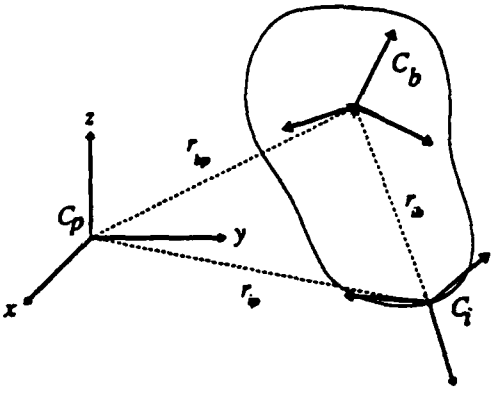


Figure 2. A rigid body with two body frames.

Let C_p, C_b be the two coordinate frames of R^3 as shown in Figure 1. Let $r_{bp} \in R^3$ and $A_{bp} \in SO(3)$ denote the position and orientation of C_b relative to C_p, and express the motion of C_b relative to C_p in homogeneous representation by

$$g_{bp} = \begin{bmatrix} A_{bp} & r_{bp} \\ 0 & 1 \end{bmatrix} \in SE(3) \tag{2-1}$$

We define the translational velocity and the rotational velocity of C_b relative to C_p by

$$v_{bp} = A_{bp}^t \dot{r}_{bp} \tag{2-2a}$$

and

$$\omega_{bp} = S^{-1}(A_{bp}^t \cdot \dot{A}_{bp}) \tag{2-2b}$$

respectively, where the operator $S: R^3 \to T_e SO(3)$ is defined by

$$S: R^3 \to T_e SO(3), S\left(\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix}\right) = \begin{bmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{bmatrix} \tag{2-3}$$

which has the property that

$$S(\omega) \cdot f = \omega \times f \quad \text{for all} \quad \omega, f \in R^3 \tag{2-4}$$

and

$$A\, S(\omega)A^t = S(A\,\omega) \quad \text{for all } A \in SO(3). \tag{2-5}$$

The generalized velocity of C_b relative to C_p is of the form $(v^s_{bp}, \omega^s_{bp})$ and is obtained via

$$\begin{bmatrix} S(\omega_{bp}) & v_{bp} \\ 0 & 0 \end{bmatrix} = g^{-1}_{bp}\cdot \dot{g}_{bp}. \tag{2-6}$$

Consider now that three coordinate frames C_p, C_b and C_i of R^3, as shown in Figure 2, where C_i is fixed relative to C_b, i.e.,

$$g_{ib} = \begin{bmatrix} A_{ib} & r_{ib} \\ 0 & 1 \end{bmatrix} \text{ and } \dot{g}_{ib} = 0 \tag{2-7}$$

then the velocity of C_i relative to C_p is related to the velocity of C_b relative to C_p by the following transformation

$$\begin{bmatrix} v_{ip} \\ \omega_{ip} \end{bmatrix} = \begin{bmatrix} A^t_{ib} & -A^t_{ib}S(r_{ib}) \\ 0 & A^t_{ib} \end{bmatrix} \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix} \tag{2-8}$$

To see this, we observe that

$$g_{ip} = g_{bp}\cdot g_{ib} = \begin{bmatrix} A_{bp}A_{ib} & A_{bp}\, r_{ib} + r_{bp} \\ 0 & 1 \end{bmatrix} \tag{2-9}$$

from which we obtain that

$$S(\omega_{ip}) = (A_{bp}A_{ib})^t(\dot{A}_{bp}A_{ib}) = S(A^t_{ib}\omega_{bp}) \tag{2-10a}$$

and

$$v_{ip} = (A_{bp}A_{ib})^t(\dot{A}_{bp}r_{ib} + \dot{r}_{bp}) = -A^t_{ib}\, S(r_{ib})\omega_{bp} + A^t_{ib}v_{bp}. \tag{2-10b}$$

Combining (2-10a) and (2-10b) gives (2-9).

To describe the motion of a rigid body in R^3, a coordinate frame (C_b), called the body frame, is attached to its mass center and the motion of the body relative to an initial frame C_p is identified with the motion of C_b relative to C_p. If a second body coordinate frame C_i is introduced the velocities of the rigid body described using the two body frames are related by the transformation (2-9).

In terms of the body frame C_b, we denote the space of (generalized) velocities of the rigid body at the identity configuration e by $T_eSE(3)$. The velocity of the rigid body at an arbitrary configuration $g \in SE(3)$ is given by $g^{-1}\dot{g}$ as in (2-7). Dual to $T_eSE(3)$ is the space of generalized forces (or wrenches) that can be exerted on the rigid body, and we denote it by $T^*_eSE(3)$. We can write a generalized force (or a wrench) $\eta \in T^*_eSE(3)$ as

$$\eta = \begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} \tag{2-11}$$

where $f_{bp}, m_{bp} \in R^3$ are respectively the force and the moment exerted on the body. The work done per unit time of η on a generalized velocity $(v_{bp}^t, \omega_{bp}^t)^t$ is given by

$$[f_{bp}^t, m_{bp}^t] \cdot \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix} = f_{bp}^t v_{bp} + m_{bp}^t \omega_{bp} \tag{2-12}$$

Similarly, when a second body coordinate frame C_i is used to describe the motion of the rigid body, we denote the set of generalized forces expressed in the C_i frame by $\eta_{ip} = (f_{ip}^t, m_{ip}^t)^t$. The wrench transformation between $\eta_{bp} = (f_{bp}^t, m_{bp}^t)^t$ and η_{ip} are given by the dual relation of (2-9), using the principle of virtual work (2-12), as

$$\begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} = \begin{bmatrix} A_{ib} & 0 \\ S(r_{ib})A_{ib} & A_{ib} \end{bmatrix} \begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} \tag{2-13}$$

(2-9) and its dual (2-13) are the basic transformation relations to be used in this paper.

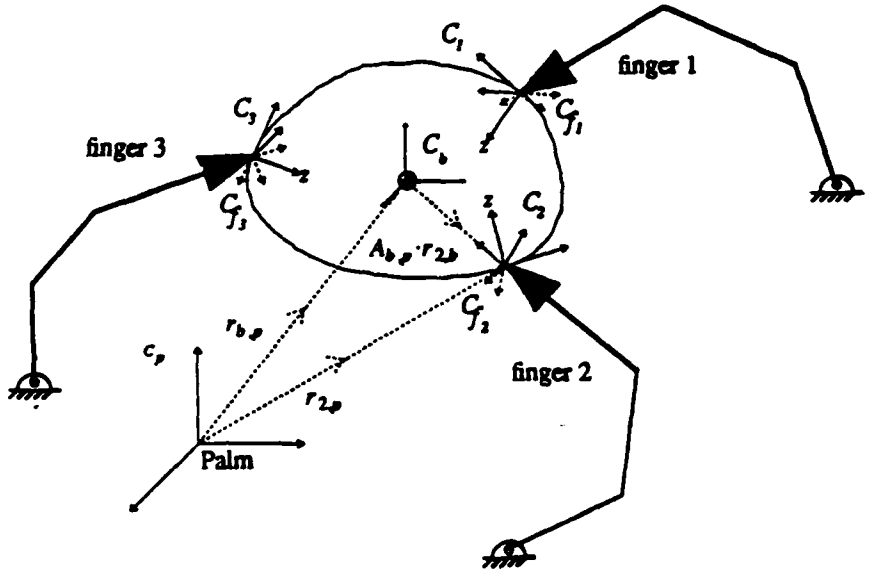


Figure 3 A three-fingered hand grasping an object.

Figure 3 shows a manipulation system with a three fingered hand, where the coordinate frame C_p is fixed to the hand palm and the coordinate frame C_b is fixed to the mass center and oriented so that the moment of inertia matrix of the object is diagonal. Associated with each contact point is a set of contact frame C_i, i = 1,... 3, which is choosen so that the z-axis coincides with the inward pointing normal to the body at the point of contact. The fingertip frame is denoted by C_{fi}, i = 1,...3.

In this paper, we use three commonly accepted contact models to model the contact between the fingertips and the object: (a) a point contact without friction, (b) a point contact with friction, and (c) a soft finger contact. It is well known that the number of independent finger wrenches that can be applied to the object through the contact is one for a point contact without friction (a force in the normal direction), three for a point contact with friction (a normal force and two frictional components in the tangential directions), and four for a soft finger contact (the three independent directions for a point contact with friction along with a torque in the normal direction) [1, 10].

Let n_i be the number of independent contact wrenches that can be applied to the body through the ith contact and $T_e^*SE(3)$ the wrench space of the object. Consider the following definition.

Definition 2.1 (contact): *A contact on a rigid body is a map* $\psi_i : \mathrm{R}^{n_i} \to T_e^*SE(3)$ *given by*

$$\psi_i : \begin{bmatrix} x_{i1} \\ \cdot \\ \cdot \\ \cdot \\ x_{in_i} \end{bmatrix} \to \begin{bmatrix} A_{ib} & 0 \\ S(r_{ib})A_{ib} & A_{ib} \end{bmatrix} B_i \begin{bmatrix} x_{i1} \\ \cdot \\ \cdot \\ \cdot \\ x_{in_i} \end{bmatrix} = T_{fi} B_i \begin{bmatrix} x_{i1} \\ \cdot \\ \cdot \\ \cdot \\ x_{in_i} \end{bmatrix} \tag{2-14}$$

Here T_{fi} is the transformation matrix specified in (2-13), and $B_i \in \mathrm{R}^{6\times n_i}$ is the basis matrix which expresses the unit contact wrenches in the contact frame ([1, 2, 6].

When a multifingered hand consists of k fingers with each finger contacting the object at a point r_{ib} with contact map $\psi_i : \mathrm{R}^{n_i} \to T_e^*SE(3)$ the grasp map for the hand is defined to be

Definition 2.2 (grasp map): *The grasp map for a k-fingered robot hand holding an object is a map* $G : \mathrm{R}^n \to T_e^*SE(3)$, $n = \sum_{i=1}^{k} n_i$ *given by*

$$G(x_{11}, \cdots x_{1n_1}, x_{21}, \cdots, x_{kn_k}) = \psi_1(x_1) + \cdots + \psi_k(x_k) \tag{2-15}$$

$$= \cdot \begin{bmatrix} T_{f1}, \cdots, T_{fk} \end{bmatrix} \begin{bmatrix} B_1 & 0 & \cdots & 0 & 0 \\ 0 & B_2 & \cdots & \cdot & \cdot \\ \cdot & 0 & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & 0 & \cdot \\ \cdot & \cdot & \cdots & B_{k-1} & 0 \\ 0 & 0 & \cdots & 0 & B_k \end{bmatrix} \begin{bmatrix} x_1 \\ \cdot \\ \cdot \\ \cdot \\ x_k \end{bmatrix} = T_f B x$$

Remarks: (1) The grasp map G transforms the applied finger wrenches expressed in the contact frames into the body wrenches in the body frame.

(2) Since a normal contact force can only be exerted unidirectionally and friction forces are finite in magnitude of size less than the normal force times the coefficient of friction, the domain of the grasp map needs to be restricted to a proper subset of R^n. For example, for a soft finger contact

the

effective force domain is

$$K_i = \{ (x_{i1}, \cdots x_{i4}) \in R^4,\ x_{i1} \geq 0, x_{i2}^2 + x_{i3}^2 \leq \mu_t^2 x_{i1}^2 \text{ and } |x_{i4}| \leq \mu_r x_{i1} \}$$

where μ_t, μ_r are the Coulomb, torsional friction coefficients respectively. The effective force domain for a point contact with and without friction are easily defined similarly and are convex cones in R^1 and R^3 respectively. The effective force domain K for the grasp map G is the direct sum of all the force domains of the contacting fingers [2].

(3) The null space ($\eta(G)$) of the grasp map G is called the space of internal grasping forces [1, 4, 8]. Any applied finger forces in $\eta(G)$ do not contribute to the motion of the object. However, during the course of manipulation a set of nonzero internal grasping forces is needed to assure that the grasp is maintained. Usually, the set of desired internal grasping forces is higher for manipulation under an uncertain environment than for manipulation under a known environment. Both [1] & [8] have presented detailed discussions on the optimal choice of internal grasping forces.

Let the ith finger have m_i joints with joint variable denoted by $\theta_i = (\theta_{i1},...,\theta_{im_i})^t$. The velocity $(v_{f_i p}^t, \omega_{f_i p}^t)^t$ of the i th fingertip frame is related to the $\dot{\theta}_i$ through the finger Jacobian by

$$\begin{bmatrix} v_{f_i p} \\ \omega_{p f} \end{bmatrix} = \bar{J}_i(\theta_i)\dot{\theta}_i \tag{2-16}$$

Now the contact frame C_i and the fingertip frame C_{f_i} are located at the same point but may have different orientations. Consequently, the velocity of the ith fingertip frame C_{f_i} seen from the ith contact frame C_i using (2-10) with ($r_{if_i} = 0$) is given by

$$\begin{bmatrix} v_{ip} \\ \omega_{ip} \end{bmatrix} = \begin{bmatrix} A_{if_i}^t & 0 \\ 0 & A_{if_i}^t \end{bmatrix} \bar{J}_i(\theta_i)\dot{\theta}_i \triangleq J_i(\theta_i)\dot{\theta}_i \tag{2-17}$$

In (2-17) above, A_{if_i} expresses the the relative orientation of the ith fingertip frame with respect to the i th contact frame and is given by $A_{if_i} = A_{ib}^{-1} \cdot A_{bp}^{-1} \cdot A_{f_i p}$. On the other hand, the motion $(v_{bp}^t, \omega_{bp}^t)^t$ of the body as seen from the i th contact frame is given by

$$\begin{bmatrix} v_{ip} \\ \omega_{ip} \end{bmatrix} = \begin{bmatrix} A_{ib}^t & -A_{ib}^t S(r_{ib}) \\ 0 & A_{ib}^t \end{bmatrix} \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix} \tag{2-18}$$

Now, the velocities as specified in (2-17) and (2-18) are not identical but agree along the directions

specified by the basis matrix B_i [1, 2, &6]. We take this contact constraints into account by insisting that

$$B_i^f J_i(\theta_i) = B_i^f T_f^i \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix} \tag{2-19}$$

Concatenating equation (2-19) for $i = 1, \cdots, k$, and defining the hand Jacobian $J_h(\theta)$ by

$$J_h(\theta) = B^t J(\theta) \tag{2-20}$$

we obtain

$$J_h(\theta)\dot{\theta} = B^t J(\theta)\dot{\theta} = G^t \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix} \tag{2-21}$$

where

$$J(\theta) = \begin{bmatrix} J_1(\theta_1) & 0 & \cdots & 0 \\ 0 & J_2(\theta_2) & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & J_k(\theta_k) \end{bmatrix}, \quad \theta = \begin{bmatrix} \theta_1 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \theta_k \end{bmatrix} \tag{2-22}$$

and G^t is the transpose of the grasp map defined in (2-15).

Equation (2-21) is the equation relating the joint velocities to the body velocity. The dual of equation (2-21) is an equation relating the joint torques of the fingers to the body wrench. We define $B_i\, x_i \in \mathrm{R}^6$ to be the finger wrench expressed in the ith contact frame with the $x_i \in \mathrm{R}^{m_i}$ representing the vector of applied finger wrenches. By the Principle of Virtual Work the resulting joint torque vector $\tau_i \in \mathrm{R}^{m_i}$ is related to $B_i x_i$ by

$$\tau_i = J_i^t(\theta_i) B_i x_i \tag{2-23}$$

Aggregating this equation for $i = 1, 2, \cdots k$ we get

$$\tau = J^t(\theta) B x = J_h^t(\theta)\, x \quad \text{with } \tau \in \mathrm{R}^m,\ x \in \mathrm{R}^n,\ \textit{and } m = \sum_{i=1}^{k} m_i. \tag{2-24}$$

Also, as we have seen from the definition of the grasp map G in (2-15), that the body wrench (f_{bp}, and m_{bp} respectively) is given by

$$\begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} = Gx \tag{2-25}$$

We claim that the equations (2-24),(2-25) are a dual of the equation (2-21). To make the duality more explicit, we define

$$\lambda = G^t \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix}, \qquad \lambda \in R^n \tag{2-26}$$

Then we may summarize the equations in the following table (see also Figure 4)

	Force Torque Relations	Velocity Relations
Body to Fingertip	$\begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} = Gx$	$\lambda = G^t \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix}$
Fingertip to Joints	$\tau = J_h^t(\theta)\, x$	$J_h(\theta)\dot{\theta} = \lambda$

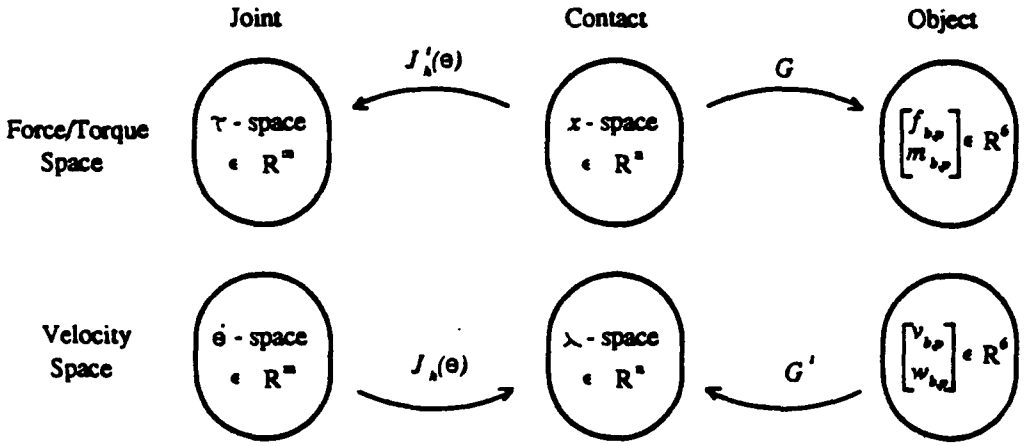


Figure 4. The force/torque and velocity transformation relations.

The following dual definitions are now intuitive.

Definition 2.3 (Stability and Manipulability of a Grasp) *Consider a grasp by a multifingered hand with k fingers each having m_i joints, $i=1,\cdots k$ and with fingertips having contacts with n_i degrees of freedom, $i=1,\cdots k$. Let $\theta \in R^m, \tau \in R^m$ represent the joint angles and torques respectively. Then:*

(i) The grasp is said to be stable if, for every wrench $(f_{bp}^t, m_{bp}^t)^t$ applied to the body, there exists a choice of joint torque τ to balance it.

(ii) The grasp is said to be manipulable if, for every motion of the body, specified by $(v_{bp}^t, \omega_{bp}^t)^t$, there exists a choice of joint velocity $\dot{\theta}$ to impart this motion without breaking contact.

Grasp stability and manipulability are now easily characterized for a given position of the fingers by

Proposition 2.4: *(i) A grasp is stable if and only if G is onto, i.e. the range space of G is the entire R^6. (ii) A grasp is manipulable if and only if $R(J_h(\theta)) \supset R(G')$, where $R(\cdot)$ denotes the range space of.*

Remark: The conditions (i) and (ii) superficially appear to be distinct, but they are related. Let us begin by examining the implications of condition (i) on grasp manipulability.

Consider Figure 4 focusing attention especially on the two orthogonal direct sum decompositions of R^n given by

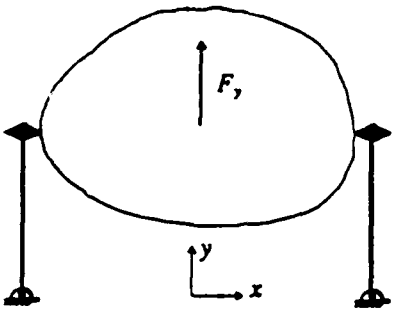
$$R^n = R(G') \oplus \eta(G) \tag{2-27}$$

$$= R(J_h(\theta)) \oplus \eta(J_h^t(\theta))$$

If G is onto, then equation (2-25) has a solution. Furthermore, the solution will be unique in the range space of G^t (the least norm solution of (2-25)). If for some body wrench there exists an x that needs zero joint torque, then $R(G') \cap \eta(J_h^t(\theta)) \neq \phi$ and consequently the condition $R(J_h(\theta)) \supset R(G')$ fails. This implies that the grasp is not manipulable.

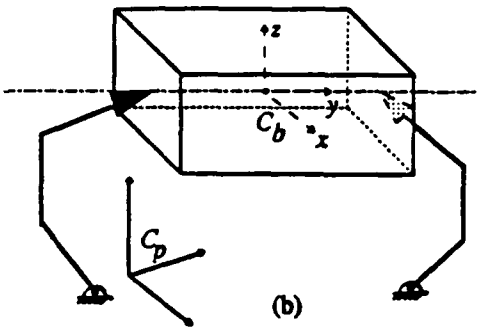
For the converse, consider the implication of condition (ii) on grasp stability. Suppose that $R(J_h) \supset R(G')$ and there exists a body velocity (v_{bp}, ω_{bp}) which produces zero λ and consequently zero $\dot{\theta}$, then $\eta(G') \neq \phi$ and therefore G can not be onto. This implies that the grasp is not stable.

To give simple examples to illustrate the foregoing comments, it is of interest to specialize the definitions to the plane. For grasping in the $x-y$ plane, the only forces and torques that need to be considered are $(f_x, f_y, m_z)^t \in R^3$ and the velocities $(v_x, v_y, \omega_z)^t \in R^3$. Figure 5 now shows a planar two fingered grasp which is stable but not manipulable. The two fingers are one jointed and the contacts are point contacts with friction. A force f_y can be resisted with no joint torque τ_1, τ_2. However the grasp is not manipulable, since a y-direction velocity on the body cannot be accommodated.



(a)

Figure 5: A stable but not manipulable grasp



(b)

Figure 6: A manipulable but not stable grasp

Figure 6 shows a grasp of a body in R^3 by two three jointed fingers. The contacts are point contacts with friction. The grasp is manipulable, since $J_h(\theta)$ has rank 6, though the object can spin around the axis $n-n$ with zero joint velocities $\dot{\theta}$. However the grasp is not stable since a body torque τ_n about the axis $n-n$ cannot be resisted by any combination of joint torques.

In view of the preceeding discussion, we will require the grasp to be both manipulable and stable, i.e.,

$$R(G) = R^6 \quad \text{and} \quad R(J_h(\theta)) \supset R(G') \tag{2-28}$$

Condition (i) suffers from the drawback that the force domain is left completely unconstrained. As we have seen earlier the forces are constrained to lie in a convex cone K, taking into account the unidirectionality of the contact forces, finite friction, etc, in which case the image of $K \cap R(J_h)$ under G should cover all of R^6. If we generalize the previous definitions to formally define a grasp to be $\Omega = (G, K, J_h(\theta))$ we have the modified stability and manipulability conditions of a grasp by

Corollary 2-5: *A grasp under unisense and finite frictional forces is both stable and manipulable if and only if*

$$G(K \cap R(J_h)) = R^6, \text{ and } R(J_h) \supset R(G') \tag{2.2-19}$$

3. Grasp Planning

Typical tasks associated with multifingered robot hands include scribing, inserting a peg into a hole, assembly operations and etc. Common to these tasks are the fact that the robot hand must manipulate an object from one configuration to another, while exerting a set of desired contact forces on the environment. Successful execution of such tasks amounts to having the robot hand perform a sequence of operations: (1) selecting a "good" grasp on the object, and (2) using the cooperative action of the fingers to control the object. As we can see that the first operation is essential to the execution of the task. For example, if a pencil is not grasped at the right position and with the right postures of the fingers, it will be extremely difficult to perform a scribing task. In this section we study how to generate a "good" grasp for a given task and in the next section we study how to manipulate the object with the cooperative action of the fingers.

Existing approaches to characterize a grasp include the "fuzzy" approach, which describes a grasp in linguistic variables such as a "power grasp", a " precision grasp", or a "lateral grasp", and etc., for more details see [29, 11], and the "nonfuzzy" approach, which describes a grasp in terms of some given criteria such as stability and objective functions ([2, 10]). In this section we will study grasp planning based on the second approach only.

In the literature various stability criteria used to characterize a grasp have been proposed and studied extensively [3, 10, 13]. But, in many cases such a criterion is too rough as it may generate a large number of stable grasps to a given object. For example, for a pencil there exist infinitely many choices of stable grasps, and while some are satisfactory for the scribing task some others are not. To solve this problem, additional criteria have been proposed in [2 & 17], for example the minimum singular value of the grasp matrix G, the determinant of GG^t [2], and some objective functions defined in [17].

After investigating human grasps, the author in [6] has suggested using the task requirement as the criterion for evaluating a grasp. Several other researchers also have had studies in incorporating the task requirement into the selection of a grasp. In [3, 15] a task is modeled by a desired compliance matrix and the final grasp is then required to have the desired compliance property. In [16] a task is modeled by a desired inertia matrix about some operating point, and the final grasp is required to have the desired inertia property at the operating point. In [2] a task is modeled by an ellipsoid, called the task ellipsoid, in the wrench space of the object, and the final grasp is required to maximize the task ellipsoid with u.. control effort. While all the above three approaches were concerned with the selection of a task oriented opt.al grasp, the nature of the tasks addressed among them are different. In [3, 15] the tasks are quasi static and the system potential energy is assumed to dominate the kinetic energy. In [16] the tasks are purely dynamic and the inertia property rather than the compliance property is the main concern in the grasp selection

process. On the other hand, the approach via [2] does apply to both classes of tasks, but it lacks the level of generality in the sense that only the wrench space and the grasp map G were considered in the optimization process.

In this section, we extend the work of [2] to consider in the selection process the aggregated behavior of $\Omega = (G, K, J_h)$ in both the wrench space and the twist space. Task modeling by task ellipsoids will take place in both the wrench space and the twist space. As in [2] the methodology of modeling a task is to associate each task one ellipsoid (A_α) in the wrench space and another ellipsoid (B_β) in the twist space. The shape of the ellipsoid A_α (B_β respectively) reflects the relative force requirement (or the motion requirement) of the task. For example, if the relative force requirement in a certain direction, such as the normal direction of the grinding application with a grinding tool, is high the task ellipsoid A_α then is shaped long in that direction. To demonstrate the precise implications of the methodology we study task modeling for the following two tasks.

Example 3-1: Consider the peg insertion task depicted in Figure 7 where the robot grasps the workpiece and inserts it into the hole.

In order to execute the task, a nominal trajectory is planned before grasping. After grasping the hand follows the planned trajectory until some misalignment of the peg causes the object to deviate from the nominal trajectory and collide with the environment.

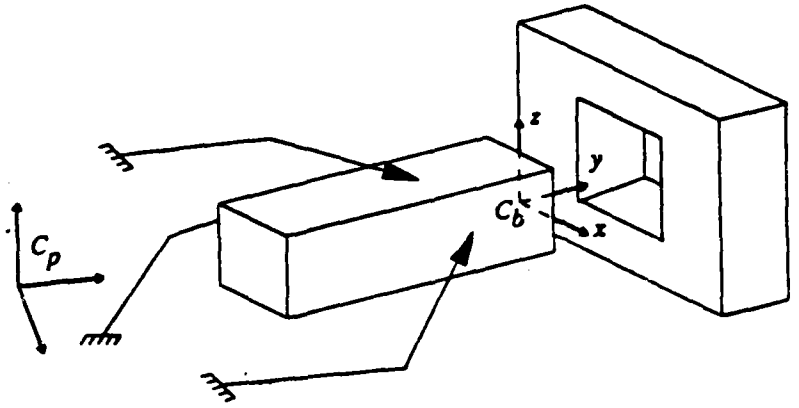


Figure 7 Peg-in-hole task.

With the body coordinate chosen as shown, the likehood of collision forces in each force direction of decreasing order would be $-f_y$, $\pm\tau_z$, $\pm\tau_x$, $\pm f_z$, $\pm f_x$, $\pm\tau_y$ and $+f_y$. If we denote by $(r_i)_{i=1}^6$ the ratio of maximum expected collision forces in each direction, we obtain a set A_α, parametrized by $\alpha \in [0, \infty)$, in the wrench space space of the object by

$$A_\alpha = \left\{ (f_x, \cdots \tau_z) \in R^6, \frac{(f_y + c_1)^2}{r_1^2} + \frac{\tau_x^2}{r_2^2} + \frac{\tau_z^2}{r_3^2} + \frac{(f_z - c_2)^2}{r_4^2} + \frac{f_x^2}{r_5^2} + \frac{\tau_y^2}{r_6^2} \le \alpha^2 \right\} \quad (3.0\text{-}1a)$$

where the constant c_1 reflects the offset of maximum expected collision force between $+f_y$ and $-f_y$ directions, and c_2 reflects the gravitational force on the object. The set A_α is an ellipsoid in the wrench space centered at $(0, c_1, c_2, 0, 0, 0)$, with the principal axes given by the generalized force directions, and axes lengths by the corresponding ratios r_i. The size of the ellipsoid is scaled by the parameter α.

By appropriately assigning a set of values to the constants (r_i, i=1, ... 6) and (c_i, i=1, 2) we can decide on the shape of the ellipsoid so that it reflects the task requirement in the wrench space. In particular, the peg insertion task requires that ($r_i \geq r_j$) whenever $i \geq j$ and c_1 to be large when collision forces in $+f_y$ direction are very unlikely.

On the other hand, since the peg insertion task requires precise positioning the grasp should provide good manipulation capability (or dexterity) in certain directions. First, in the v_y direction relatively large motion is needed. Then, the grasp should be very sensitive in ω_y, v_z and v_x directions. If we model by $(\delta_i)_{i=1}^{6}$ the ratio of relative maximum motion requirement among the six generalized velocity directions we obtain an ellipsoid B_β in the twist space, parametrized by $\beta \in [0, \infty)$, define by

$$B_\beta = \left\{ (v_x, \cdots, \omega_z) \in R^6, \frac{v_x^2}{\delta_1^2} + \frac{v_y^2}{\delta_2^2} + \frac{v_z^2}{\delta_3^2} + \frac{\omega_x^2}{\delta_4^2} + \frac{\omega_y^2}{\delta_5^2} + \frac{\omega_z^2}{\delta_6^2} \leq \beta^2 \right\} \qquad (3.0\text{-}1b)$$

The shape of B_β reflects the task requirement in the twist space. In this case δ_2, δ_3 and δ_4 are relatively larger than the other constants. Precise values of these constants can be obtained from experiments or experience through error-and-trial procedures.

Example 3-3: Consider the task of scribing with a pencil. Human experience tells us that, in order to execute the task efficiently, the grasp should provide, (1) good dexterity at the lead and (2) sufficient normal forces. With the body coordinate shown in the figure, the task requirement can be translated into requirements on the two task ellipsoids by (a) the task ellipsoid B_β in the twist space should be long in ω_y and ω_z directions and flat in the other directions, and (b) the task ellipsoid A_α in the wrench space should be long in f_z direction and then τ_z and τ_y directions. Applying this reasoning we obtain in (3.0-2) two task ellipsoids A_α and B_β that describe the relative force and velocity ratios of the task.

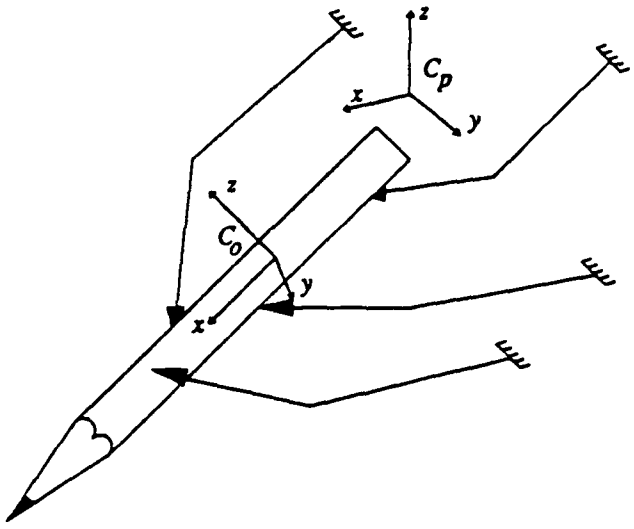


Figure 8. A scribing task.

$$A_\alpha = \left\{ (f_x, \cdots, \tau_z) \in R^6, \frac{(f_x + c)^2}{r_1^2} + \frac{f_y^2}{r_2^2} + \frac{f_z^2}{r_3^2} + \frac{\tau_x^2}{r_4^2} + \frac{\tau_y^2}{r_5^2} + \frac{\tau_z^2}{r_6^2} \le \alpha^2 \right\} \qquad (3.0\text{-}2a)$$

Wrench Space Task Ellipsoid

$$B_\beta = \left\{ (v_x, \cdots, \omega_z) \in R^6, \frac{v_x^2}{\delta_1^2} + \frac{v_y^2}{\delta_2^2} + \frac{v_z^2}{\delta_3^2} + \frac{\omega_x^2}{\delta_4^2} + \frac{\omega_y^2}{\delta_5^2} + \frac{\omega_z^2}{\delta_6^2} \le \beta^2 \right\} \qquad (3.0\text{-}2b)$$

Twist Space Task Ellipsoid

To conclude these examples, we emphasize that to each task we can associate two task ellipsoids, one in the wrench space that represents the relative force requirement and the other in the twist space that represents the relative motion requirement of the task. The constants (r_i, δ_i, c_i) that determine shapes of these ellipsoids can be obtained from experiments or from experience with similar tasks. Hence, we need to store in a library a set of ellipsoid data for a set of interesting tasks, which usually involves considerable modeling effort.

There are also other approaches to develop task ellipsoids. For example, if stiffness control is used for the hand, then the maximum expected positional uncertainities in each of the task directions may be used to scale the axis of B_β. Also, during parts mating, jamming can be avoided if certain constraints on the ratios of the contact forces are satisfied ([Whitney]), using these constraints on the force ratios to scale the ellipsoid A_α is another approach.

Generalizing from these examples we will assume that the task is modeled by generalized ellipsoids A_α (wrench space) and B_β (twist space) of the form

$$A_\alpha = \left\{ \alpha Ax + c, \textit{ such that } x, c \in R^6, \|x\| \le 1, \text{and } A \in R^{6\times 6} \right\} \quad (3.0\text{-}3a)$$

and

$$B_\beta = \left\{ \beta Bx + d, \textit{ such that } x, d \in R^6, \|x\| \le 1, \text{and } B \in R^{6\times 6} \right\} \quad (3.0\text{-}3b)$$

where the structure matrices A, B are given by

$$A = \begin{bmatrix} U_1, \cdots U_6 \end{bmatrix} \begin{bmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & \cdots & \sigma_6 \end{bmatrix} \begin{bmatrix} U_1^t \\ \vdots \\ U_6^t \end{bmatrix} = U \Sigma U^t, \; U_i \in R^6, \sigma_i \ge 0, i = 1, \cdots 6. \quad (3.0\text{-}4a)$$

and

$$B = \begin{bmatrix} U_1, \cdots U_6 \end{bmatrix} \begin{bmatrix} \delta_1 & \cdots & 0 \\ \vdots & \vdots & \vdots \\ 0 & \cdots & \delta_6 \end{bmatrix} \begin{bmatrix} U_1^t \\ \vdots \\ U_6^t \end{bmatrix} = U \Delta U^t, \; U_i \in R^6, \delta_i \ge 0, i = 1, \cdots 6. \quad (3.0\text{-}4b)$$

Here, U_i, (i = 1, ... 6,) is the task direction expressed in the body coordinates (see Section 2) and σ_i (or δ_i) is the relative importance ratio in the wrench space (and the twist space respectively) at direction U_i. The constant c, d $\in R^6$ is callled center of the task ellipsoid A_α , B_β respectively. In the following development, we may assume without loss of generality that c = d = 0, namely the task ellipsoids are all centered at the origin. When the structure matrix, say A, is nonsingular an alternative expression of A_α may be obtained as

$$A_\alpha = \left\{ y \in R^6, \textit{ such that } \langle y - c, \; \beta^{-2}(A\,A^t)^{-1}(y - c) \rangle \le 1 \right\}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product in R^6.

When stiffness control is used for the hand, σ_i is related to δ_i by $\sigma_i = K_i \cdot \delta_i$, where K_i is the desired stiffness in direction U_i. On the other hand, when hybrid position/force control is used for the hand, $\sigma_i = 0$ if direction U_i is position controlled and $\delta_i = 0$ if direction U_i is force controlled. Consequently, we see that our approach to task modeling applies to very general tasks.

3.1. The Structured Quality Measures for Grasp Planning

We have shown that a grasp $\Omega = (G, K, J_h)$ contains information about the locations of the fingertips on the object (G and K) and the postures of the fingers (J_h). Also, we have modeled the task by two

ellipsoids A_α and B_β. We now integrate these concepts to develop two quality measures for a grasp, σ_i and δ_i are related by $\sigma_i = K_i\ \delta_i$ one in the twist space and the other in the wrench space.

Definition 3-5 (The Structuured Twist Space Quality Measure μ_t): *Following the previous notation we let $O_1^m \subset R^m$ denote the unit ball in R^m, the space of finger joint velocities, and define the structured twist space quality measure $\mu_t(\Omega)$ of Ω by*

$$\mu_t(\Omega) = \sup_{\beta \in R.} \left\{ \beta,\ \text{such that } J_h(O_1^m) \supset G^t(B_\beta) \right\} \tag{3.1-1}$$

The geometric meaning of $\mu_t(\Omega)$ is as follows(see Figure 9): the unit ball O_1^m in the finger joint velocity space is mapped into the space of fingertip velocity by J_h. On the other hand, a task ellipsoid B_β in the twist space is mapped back into the fingertip velocity space by G^t; $\mu_t(\Omega)$ is then the largest β such that $G^t(B_\beta)$ is *contained* in $J_h(O_1^m)$. From a theoretical point of view, $\mu_t(\Omega)$ is the ratio of the *"structured" ouput (i.e., the task ellipsoid)* over the input (i.e., the finger joint velocity). We also see from the figure that $\mu_t(\Omega)$ is at its maximum if the inner ellipsoid has the same shape and orientation as the outer ellipsoid.

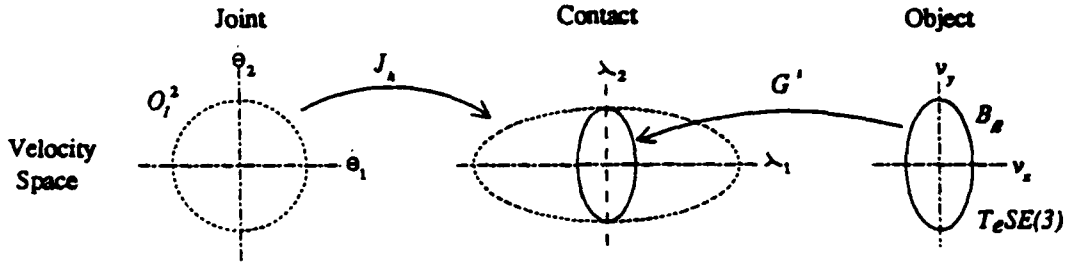


Figure 9 Geometric interpretation of μ_t.

Definition 3-6 (The Structured Wrench Space Quality Measure μ_w): *Let $O_1^n \subset R^n$ be the unit ball in the finger wrench space and $\sigma_{max}(J_h)$ the maximum singular value of J_h. We define the structured wrench space quality measure μ_w of Ω by*

$$\mu_w(\Omega) = \sup_{\alpha \in R.} \left\{ \alpha,\ such\ that\ G\ (O_1^n \cap K) \supset A_\alpha \right\} \cdot \sigma_{max}^{-1}(J_h) \tag{3.1-2}$$

Remark: The geometric meaning of (3.1-2) can be interpreted as follows (see Figure 10):-The first term is the largest α such that A_α can be emdedded in G(O_1^n) (the output), and the second term is the largest input torque required to generate the finger wrench O_1^n (the input). The structured quality measure (3.1-2) is given by the gain factor (output/input).

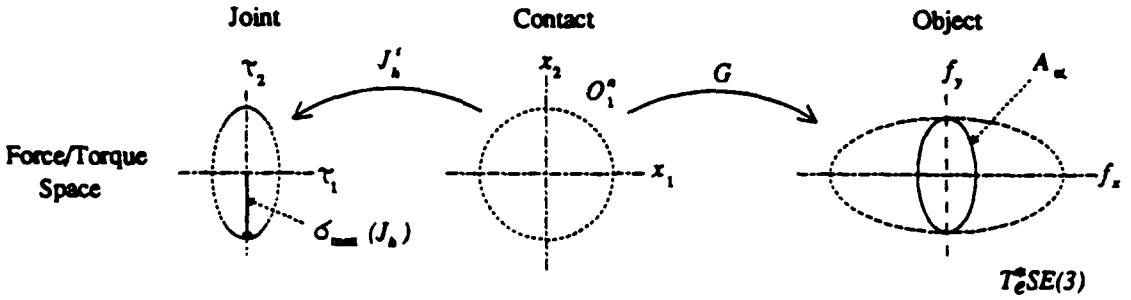


Figure 10 Geometric interpretation of μ_w.

These quality measures defined in (3.1-1) and (3.1-2) provide useful characterization of a grasp. Clearly, we can say that a grasp Ω is a good grasp with respect to a *given task, modeled by* A_α *and* B_β, if it has higher structured quality measures μ_f and μ_w than other candidate grasps. But due to unidirectionality and finite frictional forces as reflected by K, which is usually a proper subset of R^n, it is difficult to evaluate (3.1-2). To simplify this problem we will assume that $K = R^n$ and explore the properties of (3.1-1) and (3.1-2) further.

Proposition 3-7: *Under the assumption that* $K = R^n$ *the structured quality measures (3.1-1) and (3.1-2) are given by*

$$\mu_f(\Omega) = \sigma_{\max}^{-1/2}\left\{ B^t G (J_h J_h^t)^{-1} G^t B \right\} \tag{3.1-3}$$

and

$$\mu_w(\Omega) = \sigma_{\max}^{-1/2}\left\{ A^t (G\ G^t)^{-1} A \right\} \cdot \sigma_{\max}^{-1}(J_h) \tag{3.1-4}$$

Proof of (3.1-3). Using the following expression

$$J_h(O_1^m) = \left\{ y \in R^n, < y,\ (J_h J_h^t)^{-1} y > \le 1 \right\} \tag{3.1-5}$$

and

$$G^t(B_\beta) = \left\{ \beta\, G^t B x,\ x \in R^6,\ \|x\| \le 1 \right\} \tag{3.1-6}$$

in (3.1-1) and notice that $G^t(B_\beta) \subset J_h(O_1^m)$ if and only if

$$< \beta G^t B x,\ (J_h J_h^t)^{-1} \beta\, G^t B x > \le 1,\ \textit{for all}\ \|x\| \le 1 \tag{3.1-7}$$

In particular, (3.1-7) must hold for

$$\beta^2 \sup_{\|x\|=1} < G^t B x,\ (J_h J_h^t)^{-1} G^t B x > = \beta^2 \sup_{\|x^t\|=1} < x,\ (G^t B)^t (J_h J_h^t)^{-1} G^t B x > \le 1 \tag{3.1-8}$$

which is equivalent to

$$\beta \le \sigma_{max}^{-1/2} \left\{ B^t G \, (J_h \, J_h^t)^{-1} \, G^t B \right\} \tag{3.1-9}$$

But by (3.1-1) we want the largest β such that (3.1-7) is true, hence we have (3.1-3).

The proof of (3.1-4) follows immediately.

Remark: The quality measures given here can be easily evaluated using singular value decomposition data of a matrix. If we alsp want to consider the manipulability of a grasp in a certain direction, say U_i (see Figure 9), we may simply apply U_i to (3.1-7) and obtain that

$$\beta_i = < G^t B \, U_i, \, (J_h \, J_h^t)^{-1} G^t \, B \, U_i >^{-1/2} = \delta_i^{-1} \cdot < G^t \, U_i, \, (J_h \, J_h^t)^{-1} \, G^t \, U_i >^{-1/2} \tag{3.1-10}$$

Here, β_i measures teh effectiveness of the grasp in imparting motion at direction U_i, and a similar relation holds for the stability measure.

The reader may also compare these definitions specialized to a single manipulator, where G = Identity, with these of [12]. Notice that (3.1-3) and (3.1-4) exhibit an interesting ***dual relation*** in the following sense: let the task ellipsoids be the unit balls, if we hold G constant but vary J_h then $\mu_f(\Omega)$ is directly propositional to $\sigma_{min}(J_h)$, and $\mu_w(\Omega)$ is inversely propositional to $\sigma_{max}(J_h)$. On the other hand, if we hold J_h constant but vary G then $\mu_w(\Omega)$ is directly propositional to $\sigma_{min}(G)$ and $\mu_f(\Omega)$ is inversely propositional to $\sigma_{max}(G)$. This observation implies that to a ceratin point it is in general not possible to increase the two quality measures simultaneously by varying G and J_h. Namely, increasing one quality measure will sacrifice the other. For instance, if we select a "power grasp " in the scribing task, i.e., a grasp with high quality measure in the wrench space, then the grasp will be very inefficient in imparting motion at the pencil lead. Conversely, if we choose a " precision grasp " with high quality measure in the twist space the grasp will be very poor in rejecting disturbance forces. The objective of grasp planning, therefore, is to search for a grasp which maximizes some performance measure (PM) defined by

$$PM = [\, \mu_f(\Omega) \,]^{\gamma} \cdot [\, \mu_w(\Omega)]^{1-\gamma}, \quad \gamma \in [0, 1] \tag{3.1-11}$$

Here, γ is called the selection parameter. $\gamma > 0.5$ indicates that the task is motion oriented and $\gamma < 0.5$ indicates that the task is force oriented. A grasp that maximizes PM with γ close to 1 will be a " precision grasp" and a grasp that maximizes Pm with γ close to 0 will be a "power grasp". More generally, a grasp that maximizes with γ close to 0.5 will be both stable and manipulable. Depending on the nature of the task we can set the parameter γ so that the final grasp satisfies the task requirement. Consequently, γ should also be included as a modeling variable of the task.

To plan for the final grasp, we can in principle use PM, gemoetry of the object and structures of the hand to formulate the corresponding optimization problem (see [2] for details) and by solving this opimization problem we obtain the final grasp. However, simultaneously placing all k fingers optimally usually involves considerable computation and is thus impractical. For example, for a three fingered hand with three jointed fingers the optimization problem involves 27 variables.

To simplify this problem, we observe that a human usually plans grasp by sequentially placing his fingers around the periphery of the object. Mimicking this procedure we arrive at the following suboptimal algorithm for grasp planning: *palce fingers one after another on the object periphery using the constraint that the objective function PM be maximized.*

For this, we assume that the structured stability measures (3.1-4) is the only objective, i.e., PM with $\gamma = 1$, and the task is defined in (3.0-3a). Define two sequence of matrices by:

$$G_0 = 0 \qquad J_0 = 0$$

$$G_1(r_{1b}) = [\, T_{f_1}(r_{1b})B_1 \,] \qquad J_1 = [\, B_1^t \; J_1(\theta_1) \,]$$

$$\vdots \qquad\qquad \vdots$$

$$G_i(r_{ib}) = [\, T_{f_1}B_1, \ldots, T_{f_{i-1}}B_{i-1}, T_{f_i}(r_{ib})B_i \,] \quad \text{and} \quad J_i = diag\,\{ J_1, \ldots, J_{i-1}, B_i^t J_i(\theta_i) \}$$

$$\vdots \qquad\qquad \vdots$$

$$G_k(r_{kb}) = [\, T_{f_1}B_1, \ldots, T_{f_k}(r_{kb})B_k \,] \qquad J_k = diag\,\{J_1, \ldots, B_k^t J_k(\theta_k)\}$$

where r_{ib} belongs to the periphery, ∂O, of the object. $G_i(r_{ib})$ is seen to be the grasp matrix of the first i fingers and depends on r_{ib} if the previous $i-1$ fingers have been located. J_i is the hand Jacobian of the first i fingers. Clearly, the following holds

$$\sigma_{max}(J_k) = \max_{j \in \{1, k\}} [\, \sigma_{max}(B_j^t J_j(\theta_j))]$$

Thus, the finger joint variable can be chosen separately from the finger locations. Consequently, the only objective here is solve for r_{ib} assuming $r_{1b}, \ldots r_{i-1\,b}$ have been chosen. For this we consider two separate cases: rank (G_{i-1}) <6 and rank (G_{i-1}) = 6.

If rank(G_{i-1}) < 6, then $G_i(r_{ib})$ may not be of full rank and (3.1-4) is not well defined (consider what happens for i=1). Hence, we can not use the objective function (3.1-4) to solve for r_{ib}. Let $r_{ib} \in$ *aprtialO* and we decompose $G_i(r_{ib})$ according to

$$G_i(r_{ib}) = \begin{bmatrix} V_1, \ldots, V_6 \end{bmatrix} \begin{bmatrix} \rho_1 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & \rho_{s_i} & 0 & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} U_1^t \\ \vdots \\ \vdots \end{bmatrix}, \; s_i = \sum_{j=1}^{i} n_j \,, \text{ and } \rho_1 > \cdots > \rho_{s_i} > 0 \quad (3.1\text{-}12)$$

where t_i = rank $(G_i(r_{ib}))$, and ρ_j, j = 1, ... t_i is the nonzero singular value of $G_i(p_i)$. We will refer to V_j, which corresponds to the *j−th* largest singular value of G_i, as the *j−th* principal axis.

G_i maps the unit ball $O_1^{s_i}$ of R^{s_i} to an ellipsoid of R^6. The length of intersection of V_j, j= 1, ... t_i, with the unit task ellipsoid A_1 is given by (see Figure 11)

$$w_j = < V_j \ , \ (A \ A^t)^{-1} \ V_j >^{-1/2} \tag{3.1-13}$$

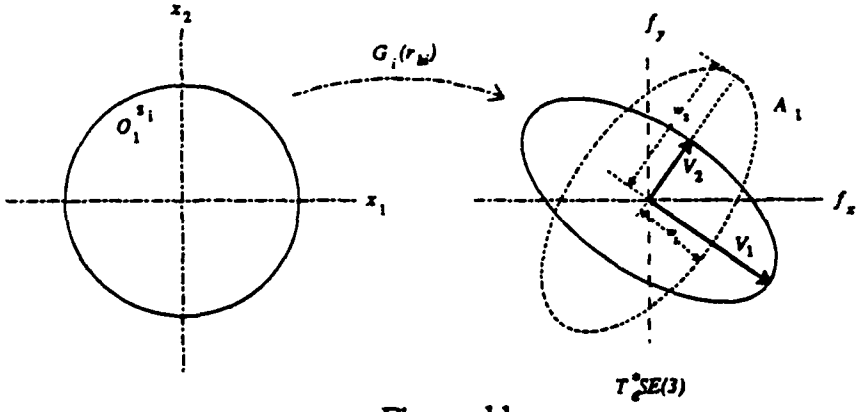


Figure 11

w_j is maximized if V_j coincides with the first principal axis of A_1, i.e., the principal axis with the largest singular value.

Define a function $\eta : G_i(r_{ib}) \rightarrow R_+$ by

$$\eta: G_i(r_{ib}) \rightarrow R_+, \quad \eta(G_i(r_{ib})) = \sum_{j=1}^{t_i} \rho_j \cdot w_j \tag{3.1-14}$$

which measures the "resemblance" of the ellipsoid $G_i(r_{ib})(O_1^{s_i})$ to the ellipsoid A_1, and can be thought of as a generalization of Definition 3-6. Apparently, η is maximum when V_1 coincides with the first principal axis of A_1, and V_2 coincides with the second principal axis of A_1, and so on for V_j, j = 3, ... t_i. For a special case consider i=1 and a model of point contact without friction. η is maximized if the normal force vector applied by the fingertip aligns with the first principal axis of A_1. In other words, the applied finger force is in the direction of worst task requirement. In general, due to the constraints on the structures of G_i we can not achieve the global maximum. But clearly, *the optimal choice of r_{ib} is to maximize η while staying on the periphery of the object.*

If rank $(G_{i-1}) = 6$; we then solve for r_{ib} by maximizing the objective function PM, (3.1-4).

Generalizing the preceeding discussions, we formalize the suboptimal algorithm as follows:

For i = 1, ... k, do

If rank $(G_{i-1}) < 6$, then compute r_{ib} by solving

$$\underset{r_{ib} \in \partial O}{Maximize}\ \eta(G_i(r_{ib})) \tag{3.1-15}$$

If rank (G_{i-1}) = 6, then compute r_{ib} by solving

$$\underset{r_{ib} \in \partial O}{maximize}\ \sigma_{max}^{-1/2}\left\{A^t(G_i(r_{ib})\, G_i(r_{ib})^t)^{-1}A\right\} \tag{3.1-16}$$

Remark: (1) The computation involved in (3.1-15) and (3.1-16) is considerably simplified than as in the previous paper ([2]). Each step the optimization problem is with respect to a single vector $r_{ib} \in R^3 \cap \partial O$ and there are numerically efficient algorithms ([30]) for these problems.

(2) Generalizing this suboptimal algorithm to PM with $\gamma \in (0, 1)$ will enable us selecting both stable and manipulable grasps.

Part of our current effort is to develop and implement this algorithm and extend it to PM with $\gamma \in (0, 1)$. Here, we present a simple example to illustrate the preceeding discussions.

Example 3-8: Consider again the planar Peg-in-Hole task, shown in Figure 12. For simplicity we assume that the body orientation coincides with the orientation of C_p. Hence, the task direction matrix U expressed in C_b is given by $U = I \in R^{3\times3}$.

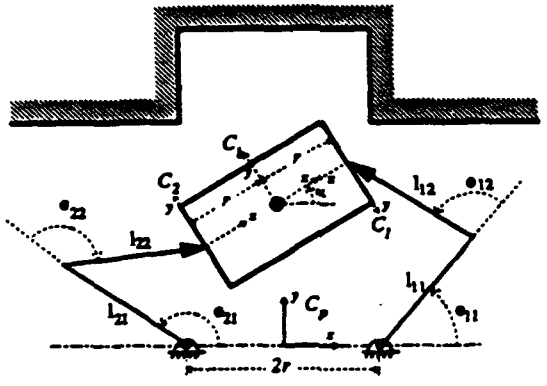


Figure 12. Planar peg-in-hole task.

To execute the task stiffness control is used for the hand ([28]) and we assume that the desired stiffnes matrix is given by

$$K = \begin{bmatrix} K_x & 0 & 0 \\ 0 & K_y & 0 \\ 0 & 0 & K_\theta \end{bmatrix} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 7 & 0 \\ 0 & 0 & 100 \end{bmatrix} \tag{3.1-17}$$

From [28] and Example 3-1 we model the task by

$$U = I \in R^{3\times3},\ \Sigma = diag\left\{4, 5, 2\right\},\ \Delta = diag\left\{0.8, 0.7, 0.02\right\} = K^{-1}\cdot\Sigma \tag{3.1-18}$$

With this task modeling the objective is to search for grasps that maximize quality measures (3.1-3) and (3.1-4).

Let the contact be modeled as a point contact with friction, the block width and the finger spacing be 2. To simplify the problem further we make additional assumptions: G is fixed as in the figure and the object is constrained to move vertically. This leaves the system with a single degree of freedom. Let θ_{11} be *the generalized coordinate of the system* and we study how θ_{11} affecting the structured grasp quality measures.

As shown in the figure, the grasp matrix is given by

$$G = \begin{bmatrix} -1 & 0 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & -1 & 0 & -1 \end{bmatrix} \tag{3.1-19}$$

and the hand Jacobian J_h is

$$J_h = \begin{bmatrix} J_1 & 0 \\ 0 & J_2 \end{bmatrix} \tag{3.1-20}$$

where

$$J_1 = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} -\sin\theta_{11} - \sin(\theta_{11}+\theta_{12}) & -\sin(\theta_{11}+\theta_{12}) \\ \cos\theta_{11} + \cos(\theta_{11}+\theta_{12}) & \cos(\theta_{11}+\theta_{12}) \end{bmatrix} \tag{3.1-21}$$

and

$$J_2 = \begin{bmatrix} -\cos\alpha & \sin\alpha \\ -\sin\alpha & -\cos\alpha \end{bmatrix} \begin{bmatrix} -\sin\theta_{21} - \sin(\theta_{21}-\theta_{22}) & \sin(\theta_{21}-\theta_{22}) \\ \cos\theta_{21} + \cos(\theta_{21}-\theta_{22}) & -\cos(\theta_{21}+\theta_{22}) \end{bmatrix} \tag{3.1-22}$$

Where α is the orientation angle of the object. The previous assumptions impose the following constraints: $\alpha = 0$, $\theta_{12} = \pi - 2\theta_{11}$, $\theta_{21} = \pi - \theta_{11}$, and $\theta_{11} - \theta_{22} = \pi - (\theta_{11} + \theta_{12})$. Figure 12b shows plots of the quality measures and the performance measure ($\gamma = 0.5$) as functions of θ_{11}. The structured measure $\bar{\mu}_s(\Omega)$ and PM attain their maximum at $\theta_{11} = 0.475$ radian (27° degree). Since G is held constant the task structures *have no effect on* μ_w, *which is still inversely propositional to* $\sigma_{max}(J_h)$. Clearly, the optimal posture (or grasp) is $\theta_{11} = 27°$.

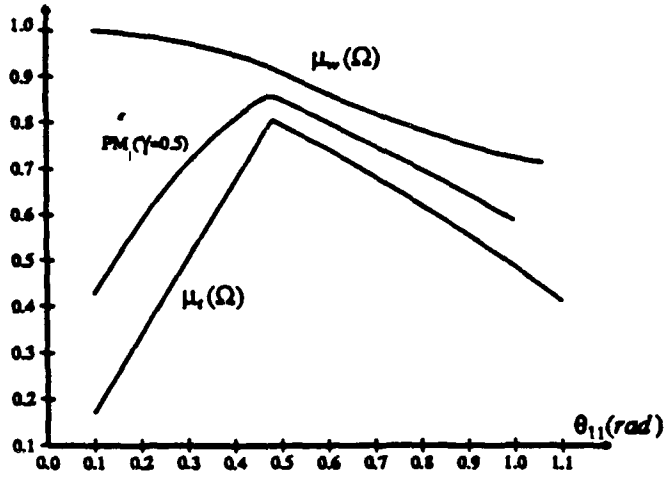


Figure 12b. Quality measures and performance measure versus θ_{11}.

4. Coordinated Control of a Multifingered Hand

In this section, we develop control algorithms for the coordinated control of a multifingered robot hand. The goal of the control scheme is to specify a set of control inputs for the finger motors so that the gripped object undergoes a desired body motion while exerting a set of desired contact forces on the environment.

Previous researchers have suggested the so-called "master-slave" control methodology for two robot manipulators (see, for e.g. [23, 24]). Others have generalized this method to a group of several manipulators (or a multifingered robot hand) [25]. In [8], [26] alternative approaches have also been proposed. The principal drawbacks associated with most existing schemes ([26], [23] & [24]) are the fact that they have all assumed *rigid attachment of the fingertips to the target object.* Consequently, in order to impart motion on the object each finger needs to be six jointed. This contradicts the reality that most developed robot hands have three or less joints for each finger. On the other hand, the scheme proposed as in [8] was open loop and did not take the finger dynamics into account. For many industrial applications associated with a multifingered robot hand, such as micro-manipulation (e.g. chip insertion), hand inertias are often compatible to that of the target object. Neglecting hand inertias in the control law, especially in a coordinated multiple robotic system, may introduce significant errors and possibly instability to the system.

The control scheme we develop in this section, which is based on a generalization of the computed torque methodology, will obviate most problems associated with the previous approaches. Namely, it will have the following desirable features: *(1) it assumes a point contact model, (2) it takes hand dynamics into account, (3) it realizes both the desired position trajectory and the desired internal grasping force trajectory, and (4) it can be easily extended to permitt rolling motion of the fingertips on the object.* The last property can be appreciated when we study fine manipulation of an object within the hand.

Without loss of generality, we may assume that the desired task is: (1) to manipulate the object along the following prespecified trajectory

$$g_{bp\mathcal{A}}(t) = \begin{bmatrix} A_{bp\mathcal{A}}(t) & r_{bp\mathcal{A}}(t) \\ 0 & 1 \end{bmatrix} \in SE(3) \qquad (4.0\text{-}1)$$

and (2) to maintain a set of desired internal grasping forces during the course of manipulation. Even though a grasp chosen based on the preceeding discussions may significantly benefit the control from an energy stand point of view, we only need the following simple assumption about the grasp:

(A1): The grasp is both stable and manipulable (see Corollary 2-6).

A necessary condition for (A1) to hold is that both the grasp map G and the hand Jacobian $J_h(\theta) = B^t J(\theta)$ be of full rank. From Section 2.2, we know that in order to maintain the contact during manipulation the finger joint velocity $\dot{\theta}$ and the object velocity $[v_{bp}^t, \omega_{bp}^t]^t$ must satisfy the following velocity constraint relation:

$$J_h(\theta)\dot{\theta} = G^t \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix} \tag{4.0-2}$$

Differentiating (4.0-2), we obtain the following acceleration constraint equation

$$J_h(\theta)\ddot{\theta} + \dot{J}_h(\theta)\dot{\theta} = G^t \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} \tag{4.0-3}$$

Since $R(J_h(\theta)) \supset R(G^t)$ by assumption (A1), we may express the joint acceleration $\ddot{\theta}$ in terms of the object acceleration $[\dot{v}_{bp}^t, \dot{\omega}_{bp}^t]^t$ by

$$\ddot{\theta} = J_h^+ G^t \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} - J_h^+ \dot{J}_h \dot{\theta} + \ddot{\theta}_o \tag{4.0-4}$$

here $J_j^+ = J_h^t (J_h J_h^t)^{-1}$ is the generalied inverse of J_h, and $\ddot{\theta}_o \in \eta(J_h)$ is the internal motion of redundant joints not affecting the object motion.

Remark: (1) Using (4.0-4) we can develop the control algorithm in the operational space of the body being manipulated. But if we express the object acceleration in terms of $\ddot{\theta}$ by

$$[v_{bp}^t, \omega_{bp}^t]^t = (GG^t)^{-1}(J_h\ddot{\theta} + \dot{J}_h\dot{\theta})$$

we can develop a control algorithm in the joint space of the fingers. In future work we will consider this alternative since it appears to hold some interesting and different possibilities.

(2) When J_h is square, its generalized inverse J_h^+ is just the usual inverse, and $\ddot{\theta}_o$ disappears from (4.0-4). This also implies that the joint motion is determined uniquely by the motion of the object.

The dynamics of the object are given by the Newton-Euler equations

$$\begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} + \begin{bmatrix} \omega_{bp} \times m v_{bp} \\ \omega_{bp} \times I\, \omega_{bp} \end{bmatrix} = \begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} \tag{4.0-5}$$

where $m \in R^{3\times3}$ is the diagonal matrix with the object mass in the diagonal, $I \in R^{3\times3}$ is the object inertia matrix with respect to the body coordinates, and $[f_{bp}^t, m_{bp}^t]^t$ is the applied body wrench in the body coordinates which is also related to the applied finger wrench $x \in R^n$ through

$$Gx = \begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} \tag{4.0-6}$$

Since we have assumed that the grasp is stable, i.e., G is onto, we may solve (4.0-6) as

$$x = G^{+}\begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} + x_o \tag{4.0-7}$$

where $G^{+} = G^{t}(GG^{t})^{-1}$ is the left inverse of G, and $x_o \in \eta(G)$ is the internal grasping force. Part of the control objective is to steer the internal grasping force x_o to a certain desired value $x_{o,d} \in \eta(G)$.

Combining (4.0-5) and (4.0-7) yields

$$x = G^{+}\left\{ \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} + \begin{bmatrix} \omega_{bp} \times mv_{bp} \\ \omega_{bp} \times I\,\omega_{bp} \end{bmatrix} \right\} + x_o \tag{4.0-8}$$

4.1. The Control Algorithm

The dynamics of the i th finger manipulator is given by

$$M_i(\theta_i)\ddot{\theta}_i + N_i(\theta_i,\dot{\theta}_i) = \tau_i - J_i^{t}(\theta_i)B_i x_i \tag{4.1-1}$$

Here, as is common in the literature $M_i(\theta_i) \in R^{n \times n}$ is the moment of inertia matrix of the i th finger manipulator, $N_i(\theta_i,\dot{\theta}_i) \in R^{n}$ the centrifugal, Coriolis and gravitational force terms, τ_i the vector of joint torque inputs and $B_i x_i \in R^6$ the vector of applied finger wrenches. Define

$$M(\theta) = \begin{bmatrix} M_1(\theta_1) & 0 & \cdots & 0 \\ 0 & M_2(\theta_2) & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & M_k(\theta_k) \end{bmatrix}, \quad N(\theta,\dot{\theta}) = \begin{bmatrix} N_1(\theta_1,\dot{\theta}_1) \\ \vdots \\ N_k(\theta_k,\dot{\theta}_k) \end{bmatrix} \quad \text{and} \quad \tau = \begin{bmatrix} \tau_1 \\ \vdots \\ \tau_k \end{bmatrix} \tag{4.1-2}$$

Then, the finger dynamics can be grouped to yield

$$M(\theta)\ddot{\theta} + N(\theta,\dot{\theta}) = \tau - J_h(\theta)^{t}\, x \tag{4.1-3}$$

The control objective is to specify a set of joint torque inputs τ so that both the desired body motion $g_{bp,d}(t)$ and the desired internal grasping force x_{od} are realized.

Since SO(3) is a compact three dimensional manifold, we may locally parametrize it by either the Euler angles, the Pitch-roll-yaw variables [20,22], or the exponential coordinates ([22]). Let $\phi_{bp} = [\phi_1,\phi_2,\phi_3]^{t}$ be a parametrization of SO(3), we can express the body trajectory $g_{bp}(t)$ as

$$g_{bp}(t) = \begin{bmatrix} A_{bp}(\phi_{bp}(t)) & r_{bp}(t) \\ 0 & 1 \end{bmatrix} \in SO(3) \times R^3 \tag{4.1-4}$$

and the body velocity as

$$\begin{bmatrix} v_{bp}(t) \\ \omega_{pa}(t) \end{bmatrix} = U(\phi_{bp}(t), r_{bp}(t)) \begin{bmatrix} \dot{r}_{bp}(t) \\ \dot{\phi}_{bp}(t) \end{bmatrix} \tag{4.1-5}$$

where $U(\phi_{bp}(t), r_{bp}(t)) \in R^{6\times 6}$ is a parametrization dependent matrix that relates the derivatives of the parameterization to the body velocity. Differentiating (4.1-5) yields

$$\begin{bmatrix} \dot{v}_{bp}(t) \\ \dot{\omega}_{bp}(t) \end{bmatrix} = U \begin{bmatrix} \ddot{r}_{bp}(t) \\ \ddot{\phi}_{bp}(t) \end{bmatrix} + \dot{U} \begin{bmatrix} \dot{r}_{bp}(t) \\ \dot{\phi}_{bp}(t) \end{bmatrix} \tag{4.1-6}$$

Theorem 4-1: *Assume that (A1) holds and that the fingers are nonredundant, i.e., $m_i = n_i$, for $i = 1, \cdots k$. Define the position error $e_p \in R^6$ to be*

$$e_p = \begin{bmatrix} r_{bp} \\ \phi_{bp} \end{bmatrix} - \begin{bmatrix} r_{bp,d} \\ \phi_{bp,d} \end{bmatrix} \tag{4.1-7}$$

where $[r_{bp,d}, \phi_{bp,d}]$ is the desired body trajectory, and the internal grasping force error $e_f \in R^{n-6}$ to be

$$e_f = x_o - x_{o,d} \tag{4.1-8}$$

where $x_{o,d}$ is the desired internal grasping force. Then, the control law specified by (4.1-9) realizes not only the desired body trajectory but also the desired internal grasping force.

$$\tau = N(\theta,\dot{\theta}) + J_h^t G^+ \begin{bmatrix} \omega_{bp} \times m v_{bp} \\ \omega_{bp} \times I\, \omega_{bp} \end{bmatrix} - M(\theta) J_h^{-1} \dot{J}_h \dot{\theta} + M_h \dot{U} \begin{bmatrix} \dot{r}_{bp} \\ \dot{\phi}_{bp} \end{bmatrix} \tag{4.1-9a}$$

$$+ J_h^t (x_{o,d} - K_I \int e_f) + M_h U \left\{ \begin{bmatrix} \ddot{r}_{bp,d} \\ \ddot{\phi}_{bp,d} \end{bmatrix} - K_v \dot{e}_p - K_p e_p \right\}$$

where

$$M_h = M(\theta) J_h^{-1} G^t + J_h^t G^+ \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \tag{4.1-9b}$$

and K_I is a martix that maps any vector in the null space of G into the null space of G.

Remarks: (1). (4.1-9) can be generalized to the redundant case and the results are given in Appendix A.

(2) The first four components in (4.1-9a) are used for cancellation of Coriolis, gravitational and centrifugal forces. These terms behave exactly like the nonlinearity cancellation terms in the computed torque control for a single manipulator; the term $J_h^t(x_{o,d} - K_I \int e_f)$ is the compensation for the internal grasping force loop, and the last term is the compensation for the position loop. We will see in the proof that the dynamics of the internal grasping force loop and that of the position loop are mutually decoupled. Consequently, we can design the force error integral gain K_I independently from the position feedback gains K_v and K_p.

(3) In the presence of stiction and frictional forces, a feedforward compensation can be used in (4.1-9). This has been shown to be very successful in implementing the control law on the Berkeley hand.

Proof:

The proof is very procedural and straightforward. First, we substitute (4.0-4) and (4.0-8) into (4.1-3) to get

$$M(\theta)\left\{J_h^{-1}G^t\begin{bmatrix}\dot{v}_{bp}\\ \dot{\omega}_{bp}\end{bmatrix} - J_h^{-1}\dot{J}_h\dot{\theta}\right\} + N(\theta,\dot{\theta}) = \tau - J_h^t\left\{G^+\begin{bmatrix}m & 0\\ 0 & I\end{bmatrix}\begin{bmatrix}\dot{v}_{bp}\\ \dot{\omega}_{bp}\end{bmatrix} + G^+\begin{bmatrix}\omega_{bp}\times m v_{bp}\\ \omega_{bp}\times I\,\omega_{bp}\end{bmatrix}\right\} - J_h^t x_o \tag{4.1-10}$$

Note that in (4.0-4) the generalized inverse for non-redundant fingers reduces to the regular inverse J_h^{-1} and $\ddot{\theta}_o = 0$. If we choose the following control in (4.1-10)

$$\tau = N(\theta,\dot{\theta}) + J_h^t G^t\begin{bmatrix}\omega_{bp}\times m v_{bp}\\ \omega_{bp}\times I\,\omega_{bp}\end{bmatrix} - M(\theta)J_h^{-1}\dot{J}_h\dot{\theta} + \tau_1 \tag{4.1-11}$$

where τ_1 is to be determined, we have that

$$\left\{M(\theta)J_h^{-1}G^t + J_h^tG^+\begin{bmatrix}m & 0\\ 0 & I\end{bmatrix}\right\}\begin{bmatrix}\dot{v}_{bp}\\ \dot{\omega}_{bp}\end{bmatrix} = \tau_1 - J_h^t x_o \tag{4.1-12}$$

or

$$M_h\begin{bmatrix}\dot{v}_{bp}\\ \dot{\omega}_{bp}\end{bmatrix} = \tau_1 - J_h^t x_o.$$

Substitute (4.1-6) into the above equation, we have

$$M_h\left\{U\begin{bmatrix}\ddot{r}_{bp}\\ \ddot{\phi}_{bp}\end{bmatrix} + \dot{U}\begin{bmatrix}\dot{r}_{bp}\\ \dot{\phi}_{bp}\end{bmatrix}\right\} = \tau_1 - J_h^t x_o. \tag{4.1-13}$$

Further, let the control input τ_1 be

$$\tau_1 = M_h U\left\{\begin{bmatrix}\ddot{r}_{bp,d}\\ \ddot{\phi}_{bp,d}\end{bmatrix} - K_v\dot{e}_p - K_p e_p\right\} + M_h\dot{U}\begin{bmatrix}\dot{r}_{bp}\\ \dot{\phi}_{bp}\end{bmatrix} + J_h^t(x_{o,d} - K_I\int e_f) \tag{4.1-14}$$

and apply it to (4.1-13) to yield:

$$M_h U\left\{\ddot{e}_p + K_v\dot{e}_p + K_p e_p\right\} = -J_h^t(e_f + K_I\int e_f). \tag{4.1-15}$$

Multiply the above equation by GJ_h^{-t}, we obtain the following equation.

$$GJ_h^{-t}M_h U\left\{\ddot{e}_p + K_v\dot{e}_p + K_p e_p\right\} = G(e_f + K_I\int e_f) = 0. \tag{4.1-16}$$

where we have used the fact that the internal grasping forces lie in the null space of G, i.e.,

$$G(e_f + K_I\int e_f) = 0, \tag{4.1-17}$$

Since $GJ_h^{-t}M_h = GJ_h^{-t}M(\theta)J_h^{-1}G^t + \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix}$ is positive definite and U is non-singular, (4.1-16) implies that

$$\ddot{e}_p + K_v\dot{e}_p + K_p e_p = 0 \tag{4.1-18}$$

Thus, we have shown that the position error e_p can be driven to zero with proper choice of the feedback gain matrices K_v and K_p.

The last step is to show that e_f also goes to zero. If we substitute (4.1-18) into (4.1-15) and notice that J_h is nonsingular, we have the following equation.

$$e_f + K_I\int e_f = 0, \tag{4.1-19}$$

With proper choice of K_I, the above equation implies that the internal grasping force error e_f converges to zero.

Q.E.D.

4.2. Simulation

Consider the two-fingered planar manipulation system shown in Figure 12. Where the two fingers are assumed to be identical. We model the contact to be a point contact with friction. The grasp matrix and the hand Jacobian are given in (3.1-19) and (3.1-20). It has been shown in Example 3-8 that the grasp configuration in the figure is both stable and manipulable. Assuming $l_{ij} = 1$, and $r = 1$ we have simulated the system to follow the following desired trajectory of the body:

$$x(t) = c_1\sin(t), \quad y(t) = c_2 + c_1\cos(t), \quad \alpha(t) = c_3\sin(t). \tag{4.2-1}$$

The dynamic equation of the i th finger (i=1,2) used in the simulation is

$$M_i = \begin{bmatrix} m_1h_1^2+m_1d_1^2+m_2l_1^2 & m_2l_1h_2C(\theta_{i2}-\theta_{i1}) \\ m_2l_1h_2C(\theta_{i2}-\theta_{i1}) & m_2(h_2^2+d_2^2) \end{bmatrix} \tag{4.2-2}$$

$$N_i = \begin{bmatrix} m_2l_1h_2\dot{\theta}_2^2S(\theta_{i2}-\theta_{i1})+m_1gh_1S\theta_{i1}+m_2gl_1S\theta_{i1} \\ m_2l_1h_2\dot{\theta}_2^2S(\theta_{i2}-\theta_{i1})+m_2gh_2S\theta_{i2} \end{bmatrix} \tag{4.2-3}$$

where m_j = mass of the j th link, d_j =radii of gyration of j th link, h_j =the distance between the j th joint and the c.m. of the j th link. The mass matrix of the object is

$$\begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I \end{bmatrix} \tag{4.2-4}$$

where m is mass of the object and I is the rotational inertial about the z axis of the object.

The simulation used a program designed to integrate differential equations with algebraic constraints. Figure 13 shows that the initial position error (in Cartesian space) diminishes exponentially as predicted by equation 4.2-18.

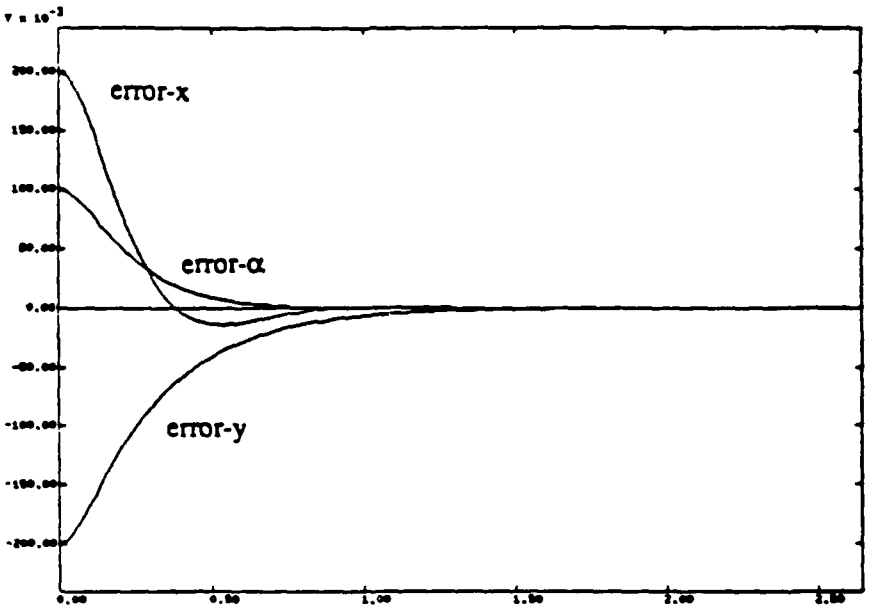


Figure 13. Position error.

The simulation was fed to a movie package which shows the continuous motion. Figure 14 and 15 are sequences of sampled pictures from a typical simulation. In both figures, the line segment at each contact shows the magnitude and the direction of the total force that is exerted to the object by the finger. The desired grasp forces are set to 0 and 10 unit force in figure 15 and 16 respectively. Note that without the grasping force (Figure 14), the total exerted force may be away from the friction cone and consequently break the contact if this were a real experiment rather than a simulation.

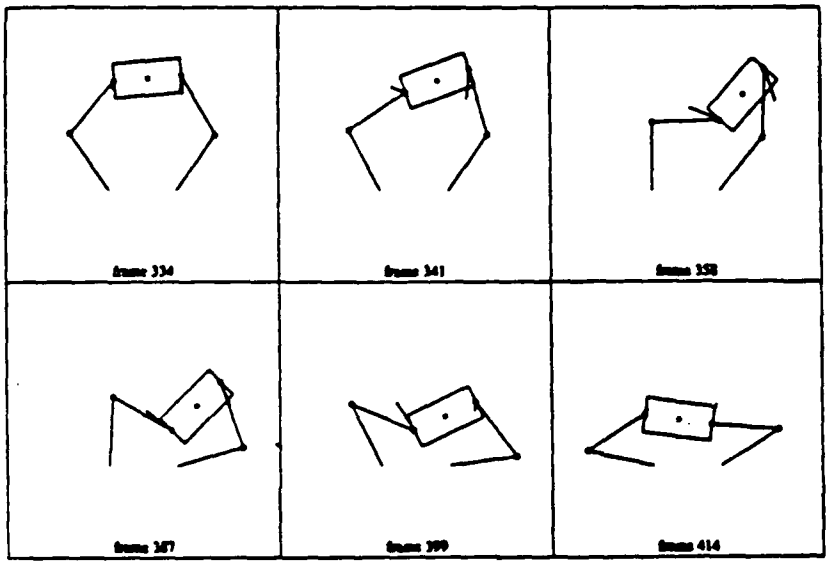


Figure 15. Simulation without internal grasping force.

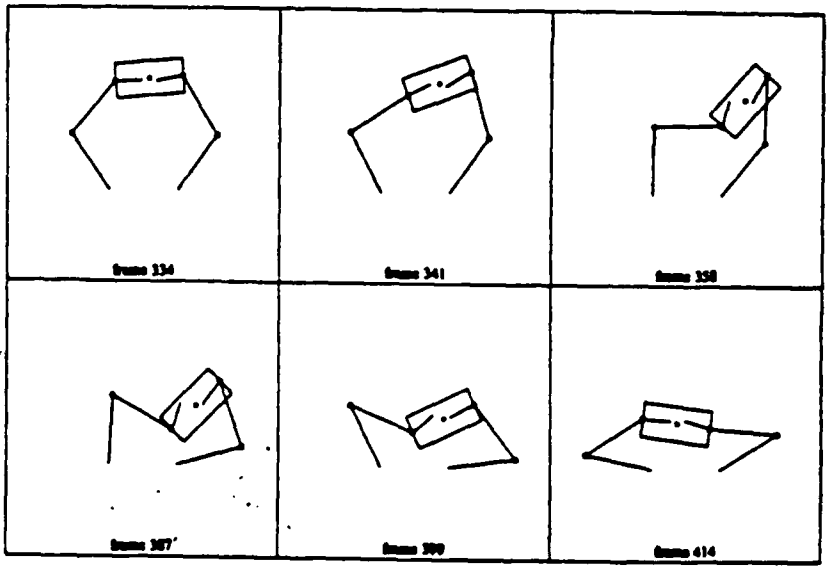


Figure 16. Simulation with 10 units of internal grasping force.

5. Redundancy Resolution

From Section 2, we see that there are two kinds of redundancy in a manipulation system: (1) when the grasp map G has more columns than the number of rows, i.e., $n>6$, we say that the grasp is redundant; (2) when the hand Jacobian J_h has more columns than the number of rows, i.e, $m>n$, then we say that the hand is redundant. Grasp redundancy is necseeary because nature supplies only uni-directional and finite frictional forces. As many researchers (for example, Salisbury [10], and Nguyen [3]) have also pointed out, that it takes at least seven frictionless point contacts, three frictional point contacts, or two soft finger contacts to completely immobilize a rigid body in space. Very often, the number of contacts used is much higher than this minum number, and there are abundant redundancies available. On the other hand, hand redundancy is necessary to achieve some additional objectives, such as singularity avoidance, dynamic collision avoidance with the object and enviroment obstacles, etc.

In this section, we formulate two kinds of redundancy control problems for multifingered robot hands: (1) redundancy resolution for grasp redundancy and (2) redundancy resolution for hand redundancy.

5.1 Grasp Redundancy Resolution: Optimal Distribution of Internal Grasp Force

From (4.0-7), we have that

$$x = G^{+}\begin{bmatrix} f_{bp} \\ m_{bp} \end{bmatrix} + x_0 \tag{5.1-1}$$

where G^{+} is the generalized inverse of G, and $x_0 \in \eta(G)$ is the component of internal force vector that renders x in the force domain K. We say that the grasp has extra redundancy when there exist more than one value of x_0 that can render x in the force domain K. Also, the control algorithm proposed in Appendix A will achieve any desired value of x_0.

IN THE FINAL DRAFT OF THE PAPER, WE WILL DESCRIBE A PROCEDURE FOR OPTIMAL GRASP REDUNDANCY REDUNDANCY.

5.2 Hand Redundancy Resolution

Hand redundancy is reflected by the fact that each finger has more degrees of freedom than needed to impart the constrained motion. This is stated in Eq. (4.0-2) as

$$J_h(\dot{\theta}) = G^{t}\begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix} \tag{5.2-1}$$

or in Eq. (4.0-3) as

$$J_h(\theta)\ddot{\theta} + \dot{J}_h(\theta)\dot{\theta} = G^{t}\begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} \tag{5.2-2}$$

Similar to redundancy resolution for a single manipulator, (5.2-1) will be used for redundancy resolution in the kinematic level, and (5.2-2) will be used in the dynamic level. To develop redundancy resolution in the kinematic level, we write from (5.2-1) that

$$\dot{\theta} = J_h^+ G^t \begin{bmatrix} v_{bp} \\ \omega_{bp} \end{bmatrix} + \dot{\theta}_0 \tag{5.2-3}$$

where $\theta_0 \in \eta(J_h)$ is any null space velocity vector to be choosen. Properly selecting $\dot{\theta}_0$ can achieve some additional objectives such as:

(1) Collision avoidance with the object, or environment obstacles;

(2) Maximize the structured manipulability measures defined in Section 3;

To develop redundancy resolution in the dynamic level, we write from (5.2-2) that

$$\ddot{\theta} = J_h^+ \left\{ G^t \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} - \dot{J}_h \dot{\theta} \right\} + \ddot{\theta}_0 \tag{5.2-4}$$

where $\ddot{\theta}_0 \in \eta(J_h)$ is any null space acceleration vector to be choosen. Properly selecting $\ddot{\theta}_0$ can achieve additional objectives, in addition to the two cases above, such as

(3) Maximize the structured stability measures defined in Section 3.

When choosing to achieve the third objective with the redundant degrees of freedom, we are really trying to minimize the joint torques.

IN THE FINAL DRAFT, WE WILL DISCUSS IN DETAIL HOW THESE PROBLEMS WILL BE FORMULATED.

6. Concluding Remarks

We have studied techniques for the determination of a grasp to an object by a multifingered hand. We have also provided a control algorithm to generate the appropriate motor torques required to manipulate an object in a certain prescribed fashion. The scheme is shown to converge in the sense that the true body trajectory converges to the desired body trajectory. An application of our scheme to a planar manipulation of an object by a two-fingered hand is presented. Finally, we have discussed how to use the available redundancy to achive some lower priority tasks such as collision avoidance, maximizing manipulability measures, or stability measures.

In future work we will study more sophisticated models for contact of a body by a multifingered hand and their implications for the schemes of this paper.

Acknowledgement

This research was supported in part by NSF under grant #DMC-8451129. We would like to thank John Hauser for performing the simulation of Figure 14 -15, Arlene Cole, Greg Heinzinger, Richard Murrary and Kris Pister for their help in prehending a number of ideas.

References

[1] J. Kerr, "An Analysis of Multifingered Hand", Ph.D. Thesis, Stanford University, Dept. of Mechanical Engineering, December 1984.

[2] Z.X. Li and S. Sastry, "Task Oriented Optimal Grasping By Multifingered Robot Hands", to appear in *IEEE Journal of Robotics and Automation*. Also, in Proc. *1987 IEEE International Conference on Robotics and Automation*, pp389-394, and UCB ERL Memo. NO. UCB/ERL M86/43, May 1986. University of California, Berkeley.

[3] V. Nguyen, "Constructing Stable Grasps in 3-D", "Constructing Force-closure Grasps in 3-D", in Proc. *1987 IEEE International Conference on Robotics and Automation*, pp234-245, March 1987.

[4] H. Kobayashi, " Geometric Considerations for a Multifingered Robot Hand", *International Journal of Robotics Research*, Vol.4, No.1, Spring 1985, pp3-12.

[5] D. Montana, " Tactile Sensing and Kinematics of Contact", Ph.D. Thesis, Harvard University, the Division of Applied Sciences, August, 1986.

[6] M. Cutkosky, "Grasping and Fine Manipulation for Automated Manufacturing", Ph.D. Thesis, Carnegie-Mellon University, Dept. of Mechanical Engineering, January, 1985.

[7] M. Brady, J. Hollerbach, T. Johnson, T. Lozano-Perez and M. Mason, *"Robot Motion Planning and Control"*, MIT Press, 1982.

[8] Y. Nakamura, K. Nagai and T. Yoshikawa, " Mechanics of Coordinative Manipulation by Multiple Robotic Mechanisms", in Proc. *1987 IEEE International Conference on Robotics and Automation*, pp991-998.

[9] T. Yoshikawa, " Manipulability of Robotic Mechanism", Technical Report, Automation Research Laboratory, Kyoto University, Japan.

[10] J. Salisbury, " Kinematic and Force Analysis of Articulated Hands", Ph.D. Thesis, Stanford University, Dept. of Mechanical Engineering, 1982.

[11] Iberall, T. " The Nature of Human Prehension: Three Dextrous Hands in One", *IEEE International Conference on Robotics and Automation*, April, 1987, Raleigh, N.C. pp390-401.

[12] S. Chiu, " Task Compatibility of Manipulator Postures", *1987 IEEE International Conference on Robotics and Automtion, pp.1718-1724. Also to appear in Internation Journal of Robotics Research.*

[13] J. Trinkle, "Mechanics and Planning of Enveloping Grasps", Ph.D. Thesis, 1987, Dept. of Computer and Information Science, University of Pennsylvania.

[14] Z. Ji and B. Roth, " Planning for Three-finger Tip-prehension Grasps", Tech. Report, 1987. Dept. of Mechanical Engineering, Stanford University, Ca.94305.

[15] M. Cutkosky, P. Akella, R. Howe, and I. Kao, " Grasping as a Contact Sport", Tech. Report, 1987. Dept. of Mechanical Engineering, Stanford University, Ca. 94305.

[16] H. Asada, "On the Dynamic Analysis of a Manipulator and its End Effector Interacting with the Environment", in Proc. *1987 IEEE International Conference on Robotics and Automation*, pp751-756.

[17] J. Barber and R. Volz, et al, " Automatic Two-fingered Grip Selection", in Proc. *1986 IEEE International Conference on Robotics and Automation*, pp890-896.

[18] J. Kerr and B. Roth, " Analysis of Multifingered Hands", *International Journal of Robotics Research*, Vol.4, No. 4, Winter 1986, pp3-17.

[19] S. Jacobsen, J. Wood, K. Bigger and E. Inversen, " The Utah/MIT Hand: Work in Progress", *International Journal of Robotics Research*, Vol. 4, No. 3, pp 221-250.

[20] R. Paul, *Robot Manipulators: Mathematics, Programming and Control*, PIT Press, 1981.

[21] Li, Z.X., P. Hsu and S. Sastry. "On Grasping and Dynamic Coordination of Multifingered Robot Hands:, Memo. No. UCB/ERL M87/63, Electronics Research Laboratory, University of California, Berkeley.

[22] V. Arnold, *Classical Mechanics* 2nd Ed. 1978.

[23] Y.F. Zheng and J.Y.S. Luh, " Control of Two Coordinated Robots in Motion", in *Proc. 24th Control and Decision Conference*, pp1761-1766, 1985.

[24] T. Tarn, A. Bejczy, and X. Yuan, " Control of Two Coordinated Robots", in Proc. *1986 IEEE International Conference on Robotics and Automation*, pp1193-1202.

[25] Arimoto, " Cooperative Motion Control of Multirobot Arms or Fingers", in Proc. *1987 IEEE International Conference on Robotics and Automation*, pp1407-1412.

[26] S. Hayati, "Hybrid Position/Force Control of Multi-Arm Cooperating Robots", in Proc. *1986 IEEE International Conference on Robotics and Automation*, pp82-89, San Francisco.

[27] A. Cole, J. Hauser, and S. Sastry, "Kinematics and Control of Multifingered Hands with Rolling Contact", submitted to 1988 *IEEE International Conference on Robotics and Automation.*

[28] Whitney, D.E. "Quasi-Static Assembly of Compliantly Supported Rigid Parts", *J. Dynamic Systems, Meas., and Control*, Vol. 104, March 1982, pp.65-77.

[29] Cutkosky, M. and P. Wright, " Modeling Manufacturing Grips and Correlations with the Design of Robotic Hands", *1986 IEEE International Conference on Robotics and Automation*, pp. 1533-1539.

[30] Polak, E. *Computational Methods in Optimization*, Academic Press, 1971.

Appendix A. Coordinated Control for Redundant Hand

In this appendix, we supplement to Theorem 4-1 the control algorithm for redundant fingers, where the number of joints n_i, $i = 1, ..., k$ is greater than the number of constrained directions n_i, $i = 1, ... k$. In many industrial applications, several robots which often have more than three degree of freedoms are integrated to maneuver a massive load. Under the frictional point contact model such a system is redundant. The dynamic distinction of a redundant hand from a nonredundant hand is the internal motion given by $\ddot{\theta}_0$ in (4.0-4). We claim that the following control law will achieve the desired control objective for a hand with redundant:

$$\tau = N(\theta, \dot{\theta}) + J_h^t G^+ \begin{bmatrix} \omega_{bp} \times m v_{bp} \\ \omega_{bp} \times I \omega_{bp} \end{bmatrix} - M J_h^+ \dot{J}_h \dot{\theta} + M J_h^+ (J_h M^{-1} J_h^t) M_h U \begin{bmatrix} \dot{r}_{bp} \\ \dot{\phi}_{bp} \end{bmatrix} \tag{a-1a}$$

$$+ M J_h^+ (J_h M^{-1} J_h^t)(x_{o,d} - K_I \int e_f) + M J_h^+ (J_h M^{-1} J_h^t) M_h U \left\{ \begin{bmatrix} \ddot{r}_{bp,d} \\ \ddot{\phi}_{bp,d} \end{bmatrix} - K_v \dot{e}_p - K_p e_p \right\}$$

where

$$M_h = (J_h M^{-1} J_h^t)^{-1} G^t + G^+ \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \tag{a-1b}$$

Remark: Please compare (a-1) with (4.1-9) and observe that the M_h is different here.

To see this, we use (4.0-4) and (4.0-8) in (4.1-3) and suppress the θ dependence of M to get

$$M \left\{ J_h^+ G^t \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} - J_h^+ \dot{J}_h \dot{\theta} \right\} + M \ddot{\theta}_0 + N(\theta, \dot{\theta}) = \tau$$

$$- J_h^t \left\{ G^+ \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} + G^+ \begin{bmatrix} \omega_{bp} \times m v_{bp} \\ \omega_{bp} \times I \omega_{bp} \end{bmatrix} \right\} - J_h^t x_0 \tag{a-2}$$

Comparing to (4.1-10) we see that there is an extra term $M \ddot{\theta}_0$ in (a-2), as well as J_h^{-1} is replaced by the generalized inverse J_h^+.

Choosing the following control input:

$$\tau = N(\theta, \dot{\theta}) + J^t G^+ \begin{bmatrix} \omega_{bp} \times m v_{bp} \\ \omega_{bp} \times I \omega_{bp} \end{bmatrix} - M J_h^+ \dot{J}_h \dot{\theta} + \tau_1 \tag{a-3}$$

where τ_1 is to be determined, we have that

$$\left\{ M J_h^+ G^t + J_h^t G^+ \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \right\} \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} + M \ddot{\theta}_0 = \tau_1 - J_h^t x_0 \tag{a-4}$$

Multiplying the above equation by $J_h M^{-1}$ and using the fact that $J_h \ddot{\theta}_0 = 0$ yields

$$\left\{ G^t + J_h M^{-1} J_h^t G^+ \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \right\} \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} = J_h M^{-1} \tau_1 - J_h M^{-1} J_h^t x_0 \qquad \text{(a-5)}$$

Since J_h is onto, $J_h M^{-1} J_h^t$ is nonsingular and we can further multiply (a-5) by $(J_h M^{-1} J_h^t)^{-1}$ to obtain

$$\left\{ (J_h M^{-1} J_h^t)^{-1} G^t + G^+ \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix} \right\} \begin{bmatrix} \dot{v}_{bp} \\ \dot{\omega}_{bp} \end{bmatrix} = (J_h M^{-1} J_h^t)^{-1} J_h M^{-1} \tau_1 - x_0 \qquad \text{(a-6)}$$

Substituting (4.1-6) into (a-6) and choosing the following remaining control input for τ_1 we obtain

$$\tau_1 = M J_h^+ (J_h M^{-1} J_h^t) \left\{ M_h U \left\{ \begin{bmatrix} \ddot{r}_{bp,d} \\ \dot{\phi}_{bp,d} \end{bmatrix} - K_v \dot{e}_p + K_p e_p \right\} + M_h \dot{U} \begin{bmatrix} \dot{r}_{bp} \\ \dot{\phi}_{bp} \end{bmatrix} + (x_{0,d} - K_I \int e_f) \right\} \qquad \text{(a-7)}$$

and

$$M_h U \{ \ddot{e}_p + K_v \dot{e}_p + K_p e_p \} = -(e_f + K_I \int e_f) \qquad \text{(a-8)}$$

Multiplying (a-8) by G and using the fact that $G(e_f + K_I \int e_f) = 0$ we obtain that

$$G\ M_h U \{ \ddot{e}_p + K_v \dot{e}_p + K_p e_p \} = 0 \qquad \text{(a-9)}$$

Since $G\ M_h = G(J_h M^{-1} J_h^t)^{-1} G^t + \begin{bmatrix} m & 0 \\ 0 & I \end{bmatrix}$ and U are both nonsingular (b-9) immediately implies that

$$\ddot{e}_p + K_v \dot{e}_p + K_p e_p = 0 \qquad \text{(a-10)}$$

which shows that the position error e_p can be driven to zero by proper choice of the feedback gain matrices K_v and K_p.

On the other hand, using (a-10) in (a-8) we finally obtain that

$$e_f + K_I \int e_f = 0 \qquad \text{(a-11)}$$

which shows that the internal grasping force error can also be driven to zero by proper choice of the forec integral gain matrix K_I.

Tactile Sensing for Dextrous Manipulation

Ronald S. Fearing †
Dept. of Electrical Engineering & Computer Science
University of California
Berkeley, CA 94720

ABSTRACT

Dextrous robot hands require intelligent control systems to manipulate unknown objects reliably in a complex environment. Tactile sensing and perception are needed to provide information on contact forces and local shape properties to ensure robust manipulation. This paper considers making global object inferences such as size, location and orientation from local geometric information at three fingers. Tactile sensors provide local shape information, including surface normals, contact location on the finger, and principal curvatures and their directions. It is assumed that the object is a cone (not necessarily circular) but is otherwise unknown. A grasping system should command finger forces to restore an object to a desired position using this tactile information. The advantage of not using specific object models is that the system will be flexible enough to handle a wide range of parts without reprogramming.

1. Introduction

Multifinger articulated hands have been developed [Okada, 1982; Salisbury and Craig, 1982; Jacobsen et al, 1985] that provide much potential for dextrous manipulation. Systems for controlling finger forces to grasp and manipulate objects in the hands have been proposed [Cutkosky, 1984; Jameson, 1985; Ji, 1987; Kerr, 1985; Salisbury and Craig, 1982], and demonstrated [Hanafusa and Asada, 1977; Okada, 1977 and 1982; Kobayashi, 1985; Fearing, 1986; Loucks et al, 1987]. None of these systems appear to use tactile sensing in the control loop, although Speeter [1987] simulates closed loop grasping using tactile force information.

---

† This work was performed while the author was a graduate student at Stanford University.

The need for tactile sensors for dextrous hands has been recognized for a long time [Tomovic and Boni, 1962; Hollerbach, 1982], but compatible sensors for dextrous fingers (which can not be planar) have just recently been developed (for example [Fearing et al 1986] and [Begej, 1986]). Tactile sensing is needed to provide contact location, forces, and basic curvature information such as distinguishing between planes, edges, and corners of objects. Because of the complicated compliance, friction, rolling, and slipping effects that occur at finger-object interfaces, it is generally not possible to predict object location exactly from a priori object models and finger joint angle measurements.

Most manipulation methods have assumed that a complete object model is available. A more dextrous system should be able to handle a variety of unknown objects in unknown positions. To pick up an unknown object, 4 stages were identified in [Fearing, 1984]; the approach, initial touch, initial grasp, and stable grasp phases. During the initial touch phase, the fingers first make contact with the object. The goal of this paper is to make as efficient use as possible of this first contact information from 3 fingers, so that in many cases enough can be inferred about the object to make a reasonable grasp attempt in the initial grasp phase. If we are lucky, this initial grasp could result in a stable grasp. Even if the grasp attempt fails, more tactile information should be available to interpret the object's shape. To make an initial grasp, the minimum representation of the object should include its location, orientation, some size constraint, and surface normal information to know whether forces will be within the friction cones at those finger sites.

2. Tactile Sensing

A tactile sensor has been packaged in a cylindrical rubber finger tip for the Stanford/JPL hand. The finger tip sensor uses an array of capacitors formed at the junctions of perpendicular copper strips [Siegel 1986], spaced at 3.3 mm along the length and 18° around the circumference of the finger. Only an 8×8 subset of the 8×20 elements is used. 3.8 mm of rubber covers the core and is essential to increase contact areas and reduce aliasing. Details of finger construction are in [Fearing et al 1986 and Fearing 1987].

After calibration, the sensor output is normalized to determine equivalent strain at each tactile element (tactel). The mean sensitivity of the tactels is 0.4 gram with a 3 mm diameter probe, and they are very linear up to 50 grams.

When the finger touches an unknown convex surface, principal curvatures, normal force and location are determined from a 4 by 4 window of strain measurements as described in [Fearing and Binford, 1988]. Sensor strains are predicted by convolving the spatial impulse response of the rubber skin with the assumed surface pressure distribution derived from a Hertz contact model. Gradient search finds the parameters of the convex second-order shape and the force that best fit the sensor data. Experiments show radius estimation within 10%, orientation within 2.5 degrees, and sub-tactel localization 3% of the element spacing.

The curvature algorithm can also be used to distinguish between contact features such as spheres, planes, cylinders, edges and vertices. The use of these features to interpret object shape will be discussed in the next section, assuming ideal curvature and orientation data.

3. Shape Interpretation Without Specific Models

There have been two extremes in determining the shape of an object from tactile sensing. Allen and Bajczy [1985] build up a map of the entire surface of unknown objects from local measurements. Gaston and Lozano-Perez [1983] and Faugeras and Hebert [1986] assume known objects, and determine object shape by matching features in the world to specific object models, keeping all consistent matches. To manipulate unknown objects, it is desired to determine as much about the global properties of the object, such as its size and orientation, from as little local sensing as is possible. Shape classification experiments with tactile sensors for very regular objects such as circular cylinders and cones have been performed [Gurfinkel et al, 1974; Kinoshita, 1977]. These techniques lack descriptive capability and generality for more complicated shapes.

We distinguish here among object classification, model matching, and shape description without specific models. Classification typically uses pattern recognition methods, and does not answer the question of where or what the object is, only which category of object it is. Gurfinkel et al [1974] used tactile curvature information obtained from a 3×3 array on a two-finger parallel-jaw gripper to classify simple shapes such as a circular cylinder, block, and sphere. Thresholds on curvature were used to classify plane, edge, vertex, spherical and planar points. The object surface was explored, and objects were classified by local curvature measurements. Kinoshita et al [1975, 1977] demonstrated discrimination of circular, square and triangular cylinders based on the total number of sensing sites activated, using 384 binary sensors on a 5

finger hand.

Hillis [1981] proposed to classify six small parts based on 3 "features"; the shape (object long or round), bumps, and whether the part rolled or not with finger motion. Ozaki et al [1982] used the pattern of surface normals around the object circumference to classify its shape. For example, a quadrilateral will have all surface normals in just four directions. One problem with the classification schemes is limited reliability. Gurfinkel correctly classified a cube from among 5 different objects in less than half the trials. Other problems are the lack of generality for complicated objects with slightly different features, and not determining the pose of the objects.

Unlike classification, model matching can localize and identify objects instead of only categorizing them. Model matching compares relations between sensed features and features on particular object models, for example, distances between points, and angles between surface normals. A consistent combination of features in the world and in the model determines the object and its position, but not necessarily uniquely. Faugeras and Hebert [1986] used surface primitive matching from range data. They minimized a "distance" measure between primitives in the model and in the sensed data to solve for object type and position. Gaston and Lozano-Perez [1983] and Grimson and Lozano-Perez [1984] showed rapid pruning of an interpretation tree of feature combinations (which is basically an exhaustive search of all possible feature matches). Ellis [1987] has developed a planning system to choose sensor paths that will prune the interpretation tree more efficiently using prior sensed data. The principal objection to these methods is that they need a specific model for each object that will be touched.

Shape description uses measurements and geometric constraints to derive a representation of the object, and does not require prior object models. Brady et al [1984] demonstrate using range data to extract the axes of surfaces of revolution using measured surface curvatures. Allen and Bajczy [1985] and Allen [1986] examined building up surface patches with a tactile sensor, in combination with vision, by exploration over the whole surface. Another type of shape description, which is used in vision, starts with all the edges in a scene linked into a line drawing, and determines valid interpretations of the edges (for example Malik [1985]). The objects in the scene are unknown, so *a priori* geometric constraints are used, for example that the objects in the scene are bounded by piecewise smooth surfaces.

These shape description methods have available rich global data on the object. Although it is possible to scan a tactile sensor over all the surfaces of an object, it will be interesting and more efficient to use the sparse local 3D tactile data that is available

from the first touches on an object. This partial preliminary information can be used to direct tactile exploration.

3.1. LSHGC Interpretation

A simplified form of the shape interpretation problem is considered here. Given three contact features on an unknown cone (not usually of circular cross-section), geometric considerations are derived to determine if the features are necessary and sufficient to determine the origin, orientation, and scale of a bounding right circular cone to the sensed cone. These contact features, which are convex elliptic, parabolic, and planar points and their sub-case vertices and edges, are determinable by the perception algorithms of [Fearing and Binford, 1988].

Objects need a representation constrained enough that global properties can be inferred from local measurements. A reasonable class constraint is the *generalized cylinder* [Binford, 1971], which is the volume generated by an arbitrary cross section translated with or without deformation along an arbitrary space curve. This representation gives an analytic expression for the whole body, and has well defined properties. The particular class constraint that is used in this paper is the convex Right Linear Straight Homogeneous Generalized Cylinder (RLSHGC), otherwise known as a simple convex cone. These objects have a convex constant cross section, which is translated along an orthogonal straight axis and scaled by a linear sweeping rule. The ends of the cone are parallel. Shafer and Kanade [1983] have shown that an axis orthogonal to the cross section exists for every LSHGC. For any arbitrary sweeping rule, a straight homogeneous generalized cylinder was shown by Ponce et al [1987] to have a unique axis if there are 2 zeros of curvature in both the cross section and the sweeping rule. That paper also shows that tangents at parabolic points at the same height on the cone intersect on the axis. This is a global property that can be used where enough of the image is available. We desire to find similar global properties for the LSHGC that can be determined from only local measurement.

We start by re-deriving the expressions for tangents and normals on the LSHGC. From Ponce et al [1987], a point on the LSHGC can be given in vector form:

$$\underline{x}(z,\theta) = (Az + B)\rho(\theta)\,(\cos\theta\hat{i} + \sin\theta\hat{j}) + z\hat{k} \tag{3.1}$$

where the LSHGC is aligned with the k axis, A is the scale-factor, the ends of the cone are given by $z_{Top} \le z \le z_{Bottom}$, and $0 \le \rho(\theta) \le 1$ is the reference cross section. z is

defined ≥ 0, and is equal to zero only if the cone's apex is not truncated. The cross-section closed curve represented by $\rho(\theta)$ is a star-shaped function, which has all convex cross sections as a subset [Ponce et al 1987]. We consider only convex cross sections.

The convention used here is that $A = 0$ for a *regular cylinder*, and for a *cone*, $B = 0$. Both cases are subsets of the LSHGC family. The end of the cone is parallel to the x–y plane, and because $B = 0$, the apex of the cone is at the origin. The maximum *radius* of the cone is $Az\,\rho_{max}$ (where $\rho_{max} = 1$), evaluated at $z = 1$, so the "radius" of the cone is given by A. The length of the cone is given by the distance between the bottom and top of the cone $(z_B - z_T)$. The radius and length parameters give a bounding right circular cone that contains the LSHGC. There is an equivalent expression with the LSHGC cross-section in cartesian coordinates,

$$\underline{x}(z,t) = (Az + B)x(t)\hat{i} + (Az + B)y(t)\hat{j} + z\hat{k}. \tag{3.2}$$

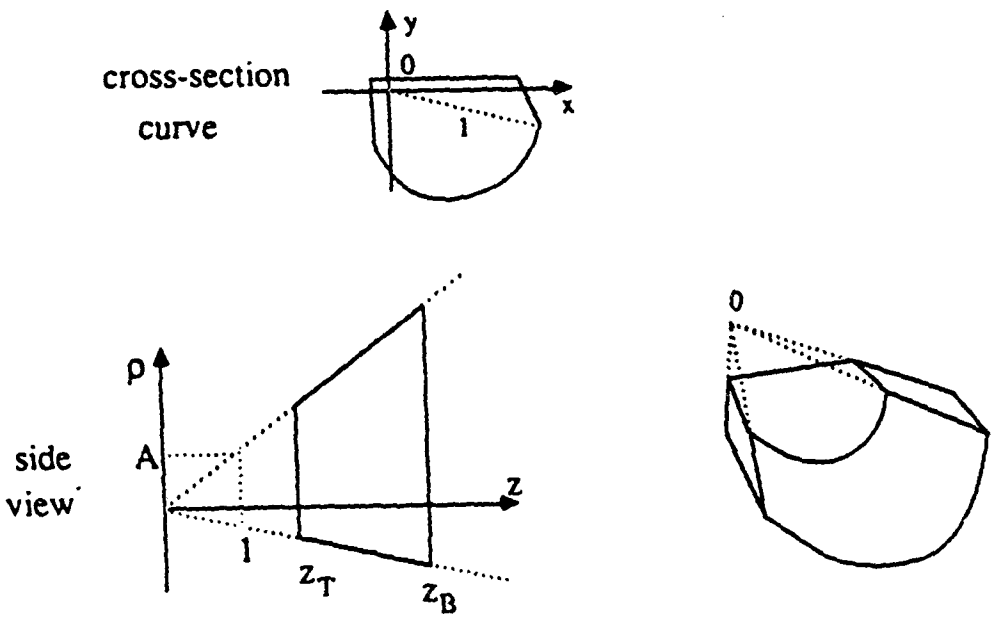


Figure 3.1. LSHGC Example

An example cone with $B = 0$ is shown in Fig. 3.1 with cross section and scale factor A indicated. Note that the origin does not need to be included inside the cross section, so cones can be slanted.

To predict what the tactile sensor will feel on this class of object, we need to determine expressions for surface normals and curvatures on the cone. These expressions will be used in later sections to solve for cone parameters. We will use cartesian coordinates to describe straight line segments, and polar coordinates for curved arcs in the cross section. Using simple differential geometry, the tangential vectors on the side of the cone are given by

$$\underline{x}_z = \frac{\partial \underline{x}}{\partial z} = \begin{bmatrix} Ax \\ Ay \\ 1 \end{bmatrix} = \begin{bmatrix} A\rho\cos\theta \\ A\rho\sin\theta \\ 1 \end{bmatrix} \tag{3.3}$$

and

$$\underline{x}_t = \frac{\partial \underline{x}}{\partial t} = (Az + B)\begin{bmatrix} x' \\ y' \\ 0 \end{bmatrix} \tag{3.4}$$

or in polar coordinates

$$\underline{x}_\theta = \frac{\partial \underline{x}}{\partial \theta} = (Az + B)\begin{bmatrix} -\rho\sin\theta + \rho'\cos\theta \\ \rho\cos\theta + \rho'\sin\theta \\ 0 \end{bmatrix}, \tag{3.5}$$

where $x' = \frac{\partial x}{\partial t}$ and $\rho' = \frac{\partial \rho}{\partial \theta}$.

By convention, the unit normal vector points out of the body. The unit normal vector for a cross-section described in cartesian coordinates is

$$\hat{n} = \frac{\underline{x}_t \times \underline{x}_z}{|\underline{x}_t \times \underline{x}_z|} = \frac{\begin{bmatrix} y' \\ -x' \\ A(yx' - xy') \end{bmatrix}}{\sqrt{y'^2 + x'^2 + A^2(yx' - xy')^2}}, \tag{3.6}$$

and for the star-shaped cross-section (which we choose to be convex) described in polar coordinates

$$\hat{n} = \frac{\underline{x}_\theta \times \underline{x}_z}{|\underline{x}_\theta \times \underline{x}_z|} = \frac{\begin{bmatrix} \rho\cos\theta + \rho'\sin\theta \\ \rho\sin\theta - \rho'\cos\theta \\ -A\rho^2 \end{bmatrix}}{\sqrt{\rho^2 + \rho'^2 + A^2\rho^4}} \tag{3.7}$$

The principal directions are the two perpendicular directions for which the normal curvatures take on maximum and minimum values [Lipschutz, 1969]. The first principal curvature is given by $\hat{k}_1 = \hat{x}_z$. Since the principal curvatures are orthogonal to each other and both lie in the tangent plane, the unit vector in the direction of the second principal curvature is:

$$\hat{k}_2 = \frac{\hat{k}_1 \times \hat{n}}{|\hat{k}_1 \times \hat{n}|} = \frac{\begin{bmatrix} -(A^2\rho^2+1)\rho\sin\theta + \rho' \cos\theta \\ (A^2\rho^2+1)\rho\cos\theta + \rho' \sin\theta \\ -A\rho\rho' \end{bmatrix}}{\sqrt{(A^2+1)^2\rho^2 + A^2\rho^2\rho'^2 + \rho^2}} . \tag{3.8}$$

It is important to note that $\hat{k}_2$ and the surface normal are generally not in the cross section plane unless $A = 0$, that is, the object is a cylinder. The principal direction $\hat{k}_2$ will be in the cross-section plane for a circular cross section ($\rho' = 0$) and when the cross-section curve goes through the origin ($\rho = 0$).

As expected for a parabolic point, κ_1 (curvature in the direction of $\hat{x}_z$) is zero. The other curvature is:

$$\kappa_2 = \frac{(A^2\rho^2+1)(\rho\rho'' - \rho^2 - 2\rho'^2)}{(Az+B)(\rho^2+\rho'^2+A^2\rho^4)^{\frac{3}{2}}} \tag{3.9}$$

The tactile sensor is able to measure κ_1, κ_2, $\hat{n}$, and $\hat{k}_1 = \hat{x}_z$.

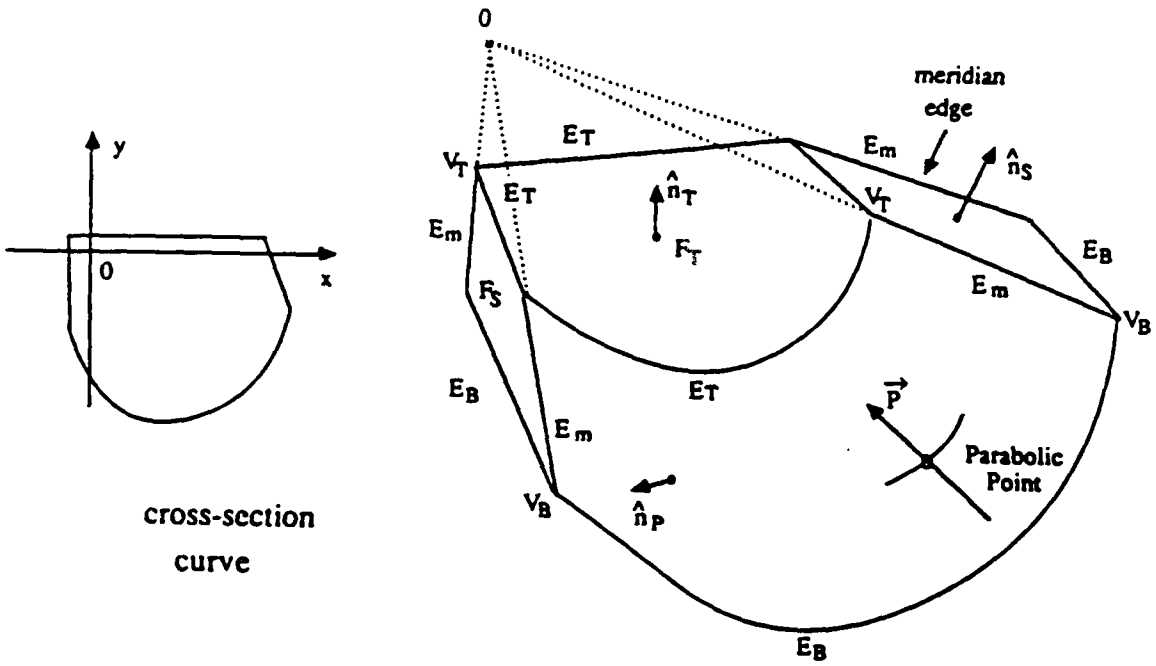


Figure 3.2. Labelled Contact Features on LSHGC.

3.2. Contact Features for LSHGC

An LSHGC has a limited set of contact features, which can be simply classified into point/vertex, plane/face, edge, and parabolic points. This section gives the relevant equations for these feature types. Each contact has a central location measurement, which corresponds to the center of pressure on the finger sensor. In addition, an edge has a direction vector (two components), a plane a surface normal, and a parabolic point has a direction vector, a surface normal and a curvature κ_2. Since these features will be mentioned frequently, they will be abbreviated as **V** for vertex (an elliptic point with very high curvature), **E** for edge, (a parabolic point with zero curvature in one direction, and very high curvature in the other), **P** for a parabolic point, (one curvature approximately zero) and **F** for a planar face, (a planar point with very small curvatures in all directions). We will classify any planar point, which corresponds to a zero of curvature in the cross section, as a face. The tactile sensor can not distinguish between a planar point and a plane such as the face of a polyhedron.

where c is the parameterization of the line.

Using the surface normal expression from eq. (3.6) and knowing that the cross section must be linear for a planar face, the surface normal for a planar contact on the side is

$$\hat{n}_S = \left[\sin\alpha, -\cos\alpha, -Ad\right]^T (A^2d^2+1)^{-1/2}. \tag{3.15}$$

The parabolic point has the same location and direction as E_m, the meridian edge, and the surface normal $\hat{n}_P$ is given by eq. (3.6):

$$\hat{n}_P = \frac{\begin{bmatrix} \rho\cos\theta + \rho'\sin\theta \\ \rho\sin\theta - \rho'\cos\theta \\ -A\rho^2 \end{bmatrix}}{\sqrt{\rho^2+\rho'^2+A^2\rho^4}}. \tag{3.16}$$

In addition, κ_2 is measured by the tactile sensor, and its expression is given by eq. (3.9).

3.3. Feature Measurement

The LSHGC has an unknown orientation and origin in the world. Thus consider an LSHGC that has been rotated about the origin by a rotation matrix **R** and translated from the origin by a vector $\underline{x}_o$. Then an object feature will be subject to the transformation

$$\underline{x}_S = \mathbf{R}\underline{x} + \underline{x}_o \tag{3.17}$$

where $\underline{x}$ is a point in object coordinates, and $\underline{x}_S$ is the sensed location in world coordinates. Since the reference cross section is not known *a priori*, any point on the cross-section can be arbitrarily chosen as $\theta = 0$. A rotation of the cross section in the $x-y$ plane will not be distinguishable, thus only 2 rotations need to be considered. Letting ϕ be the rotation around $\hat{z}$, and ψ the rotation about the $\hat{y}$ axis (i.e. spherical coordinates), the rotation matrix is:

$$\mathbf{R} = \begin{bmatrix} \cos\phi\cos\psi & -\sin\phi & \cos\phi\sin\psi \\ \sin\phi\cos\psi & \cos\phi & \sin\phi\sin\psi \\ -\sin\psi & 0 & \cos\psi \end{bmatrix}. \tag{3.18}$$

An LSHGC with labelled contact features is shown in Fig. 3.2. The subscripts to the feature labels are T for Top, B for Bottom, S for side, and m for meridian. Expressions for these contact features are derived in this section.

An edge of a cylinder end is not described well by a second order surface with $\kappa_1 \to \infty$ and $\kappa_2 \gg 0$. For a small radius of curvature, this contact can look like a vertex. Although this junction is not a C^2 surface it will be assumed that it will appear as an edge in one direction. The curvature algorithm of [Fearing and Binford, 1988] finds the best fitting paraboloid to the contact shape. The curvature algorithm is not sophisticated enough to distinguish between the line and cylinder edge types.

Let z_T and z_B be the height of the top and bottom of the LSHGC respectively as measured from the origin. A vertex can only occur at the junction of the top or bottom face and a side edge, thus the contact location is:

$$\underline{V}_T = \begin{bmatrix} (Az_T+B)\rho\cos\theta \\ (Az_T+B)\rho\sin\theta \\ z_T \end{bmatrix} \text{ or } \underline{V}_B = \begin{bmatrix} (Az_B+B)\rho\cos\theta \\ (Az_B+B)\rho\sin\theta \\ z_B \end{bmatrix} . \tag{3.10}$$

There are two types of edges, one corresponding to the discontinuities of the sweeping rule which occur at the top and bottom ends, designated as E_T and E_B respectively, and one corresponding to discontinuities in the slope of the cross section curve, designated as E_m. The edges E_m correspond to meridians on the surface. Consider a straight line segment in the cross section. It can be parameterized by

$$x(t) = d\sin\alpha + t\cos\alpha\,,\quad y(t) = -d\cos\alpha + t\sin\alpha \tag{3.11}$$

where d is the normal distance of the line segment from the origin. The derivatives are

$$x' = \cos\alpha,\quad y' = \sin\alpha\,. \tag{3.12}$$

The top edge line is given by

$$\vec{E}_T = \underline{E}_{To} + t\left[\cos\alpha, \sin\alpha, 0\right]^T \tag{3.13}$$

with t a free parameter. The meridian edge line is given by

$$\vec{E}_m = \underline{E}_{mo} + c\hat{x}_t = \underline{E}_{mo} + c\,\frac{(Ax, Ay, 1)^T}{\sqrt{1+A^2(x^2+y^2)}} \tag{3.14}$$

The global unknowns for a cone, that is the parameters that affect every local measurement, are seen to be A, z_o, ϕ, and ψ, for a total of 6 unidentified parameters. Given that the object is an LSHGC, with unknown cross-section, we want to localize it and bound its maximum radius. The scaling parameter A gives a right circular cone which bounds the LSHGC which should aid in planning an effective object grasping strategy. We shall show that it is not possible to determine the scaling parameter exactly with only local measurements.

3.4. Solving for Cone Parameters

Consider the cone with $B = 0$, that is, exclude the cylindrical case. (The set of feature measurements can be checked for consistency with a cone as in [Fearing 1987], to treat the cylindrical case separately.) We will show that sensed features can be combined to provide constraints on object orientation and scaling.

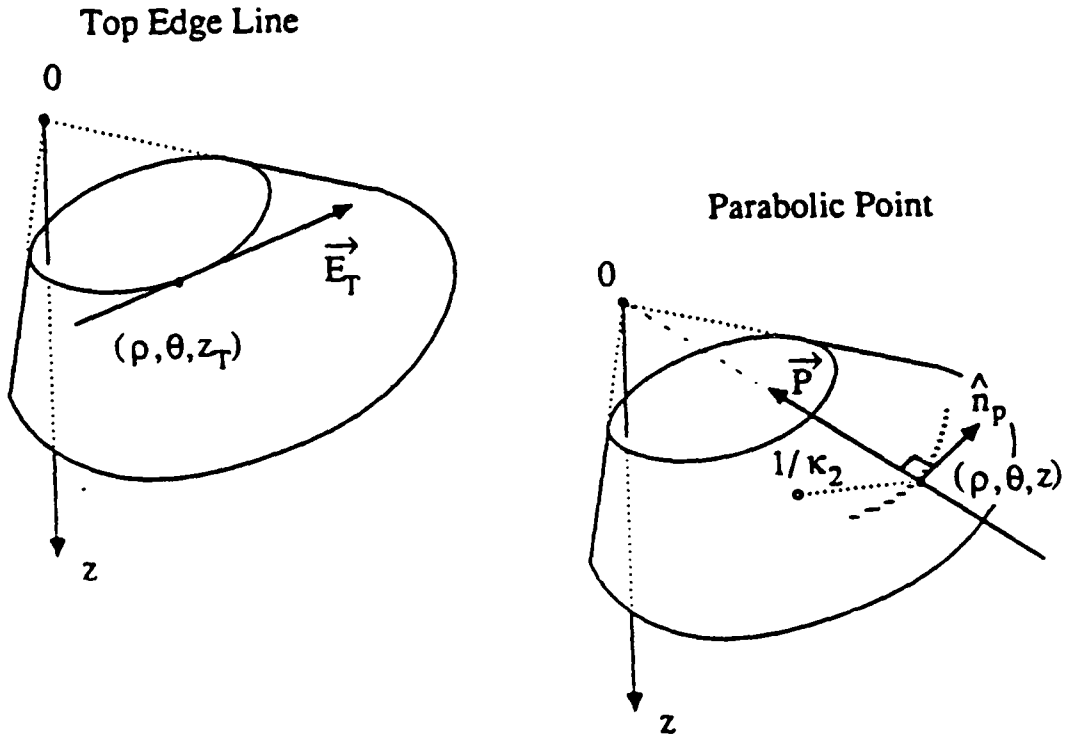


Figure 3.3. Contact Parameters for Top Edge and Parabolic Point

Examples of the parameters obtained from a contact feature are given in Fig. 3.3 for an edge on the top end of the cone, and a parabolic point on the side of the cone.

The top edge line has a contact location which is specified by three equations, and a direction in space which is specified by two independent equations (3.13), for a total of 5 constraint equations. These 5 equations involve 4 new unknowns, ρ, θ, z_T, and α, in addition to the 6 previous unidentified global parameters A, $\underline{x}_o$, ϕ, and ψ.

A meridian edge has 3 equations to specify contact location, and two independent equations to specify the line orientation (3.14). The edge has 3 new unknowns (ρ, θ, z) in addition to the 6 global unknowns. A parabolic point, also shown in Fig. 3.3, is similar to the meridian edge, but also determines a surface normal (3.16) and one curvature κ_2 (3.9). The surface normal is specified by one angle with respect to the meridian line $\vec{P}$ through the contact location, thus (3.16) provides one additional equation. With the curvature equation, there are 7 independent equations for the parabolic point. The surface normal and curvature equations introduce two new variables ρ' and ρ'', the first and second derivatives of the cross-section with respect to θ. There are thus 5 new unknowns at a parabolic point (ρ, ρ', ρ'', θ, and z).

Parabolic points provide two more equations than meridian edges, but have two more unknowns, thus there is no net gain in constraining equations. If ρ' is known, say equal to zero for a circular cross section, then a parabolic point would provide two more constraints than a meridian edge, providing 7 equations with 3 new unknowns.

Combinations of features are required to solve for unknowns; a single feature is not sufficient. For example, with a meridian edge there are 5 equations and 3+6 unknowns. When the number of equations is less than the number of unknowns, a unique solution is not obtainable.

The remainder of this section assumes that we have correctly interpreted the sensed features, that is, we know that a sensed face is an end face. From combinations of the feature equations, we can solve for some of the global cone parameters.

3.4.1. Solving for Cone Origin, $\underline{x}_o$

Two meridians of a cone always intersect at the apex (see Fig. 3.3). Consider a combination of two non-colinear meridian edges or parabolic points. Then the sensed contact points $\underline{\tilde{P}}_1, \underline{\tilde{P}}_2$ are given by

$$\underline{\tilde{P}}_1 = \mathrm{R}\underline{P}_1 + \underline{x}_o \qquad (3.19)$$

$$\underline{\tilde{P}}_2 = \mathrm{R}\underline{P}_2 + \underline{x}_o$$

where $\underline{P}_1, \underline{P}_2$ are in the object frame. The two meridian lines through the contact points (from eq. 3.14) can be shown to intersect at a point, say $\underline{P}_{12}$:

$$\underline{P}_{12} = \mathbf{R}\begin{bmatrix} A\rho_1\cos\theta_1 \\ A\rho_1\sin\theta_1 \\ 1 \end{bmatrix}(z+c_1)+\underline{x}_o = \mathbf{R}\begin{bmatrix} A\rho_2\cos\theta_2 \\ A\rho_2\sin\theta_2 \\ 1 \end{bmatrix}(z+c_2)+\underline{x}_o \qquad (3.20)$$

where c_1, c_2 are the free parameters of the lines. Since $\theta_1 \neq \theta_2$, the only solution is $z = -c_1 = -c_2$, so the intersection of the two lines gives the origin $\underline{P}_{12} = \underline{x}_o$.

3.4.2. Solving for Orientation, ψ and ϕ

To specify the cone orientation to one of two directions, it is sufficient to know the surface normal of an end. Then the sensed unit surface normal $\hat{s}$ at the top end of the cone is obtained from:

$$\hat{s} = \begin{bmatrix} s_x \\ s_y \\ s_z \end{bmatrix} = \mathbf{R}\hat{n}_T = -\begin{bmatrix} \cos\phi\sin\psi \\ \sin\phi\sin\psi \\ \cos\psi \end{bmatrix} . \qquad (3.21)$$

Thus the orientation is solved simply from

$$\tan\phi = \frac{s_y}{s_x} \text{ and } \cos\psi = -s_z \ . \qquad (3.22)$$

3.4.3. Solving for the Scaling Factor A

The orientation and origin of the cone need to be found before the scaling parameter, because A is defined on the x–y plane through $z = 1$ as the maximum distance of any point on the cross section curve from the origin. We have for any (x, y) on or inside the cross section:

$$\sqrt{x^2+y^2} \leq A \cdot z \cdot \rho_{max} . \qquad (3.23)$$

But the cross section is scaled such that $\rho_{max} = 1$, so

$$A = \max_{i,\, z=1}\left\{\sqrt{x_i^2+y_i^2}\right\} , \qquad (3.24)$$

where the maximum is taken over all (x,y) in the cross section. (Equivalently, this is the maximum $\rho(\theta)$ for $0 \le \theta \le 2\pi$.)

Once the orientation and origin have been found, one can determine the x and y location for any z location on any line through the origin from a meridian edge, side face, or parabolic patch. That is, any side contact location can be projected into the plane $z = 1$. Given only local information, not the whole contour, the maximum must be taken over the available points in the cross-section, thus only a lower bound on A can be found.

Using the convex cross-section assumption, 3 surface normals defined at side faces or parabolic points put an upper bound on A. Figure 3.4 shows a bounding triangle that must contain the cross section. The three side contacts will not necessarily form a triangle, in which case the maximum A is unbounded in some directions. The maximum radius of the bounding circle to the cross-section must be $\ge A_{min}$, the furthest contact from the origin, and $\le A_{max}$, the distance of the extremal vertex of the bounding polygon from the origin.

The included angle γ (at the vertex of the cone) is bounded by

$$\tan^{-1}A \le \gamma \le 2\tan^{-1}A \tag{3.25}$$

and has the maximum angle for a fixed A if the origin is in the "center" of the cross section.

3.5. Sufficient Feature Combinations

It was shown in the previous section that a combination of 2 meridian edges and a surface normal on an end face allows solution for the cone origin and orientation. Other combinations of features will also give a solution. This section considers which sets of features are sufficient. The origin and orientation solutions require different sets of features. We are assuming here that the features have been properly identified.

Table 3.1 gives combinations of features that are sufficient to determine the end face and thus the orientation. For example, three vertices V_{T1}, V_{T2}, V_{T3} define the top end. Table 3.1 is not exhaustive. Two vertices on the top end can define an equivalent edge (E_T), which could be used to solve for the end face in combination with a bottom edge (E_B). A top and bottom edge will not always be sufficient. The restriction that the edges are not parallel is noted in the table. These two parallel lines do not define a unique plane. However, if two edges are parallel, and on the same end plane ($E_{T1}E_{T2}$),

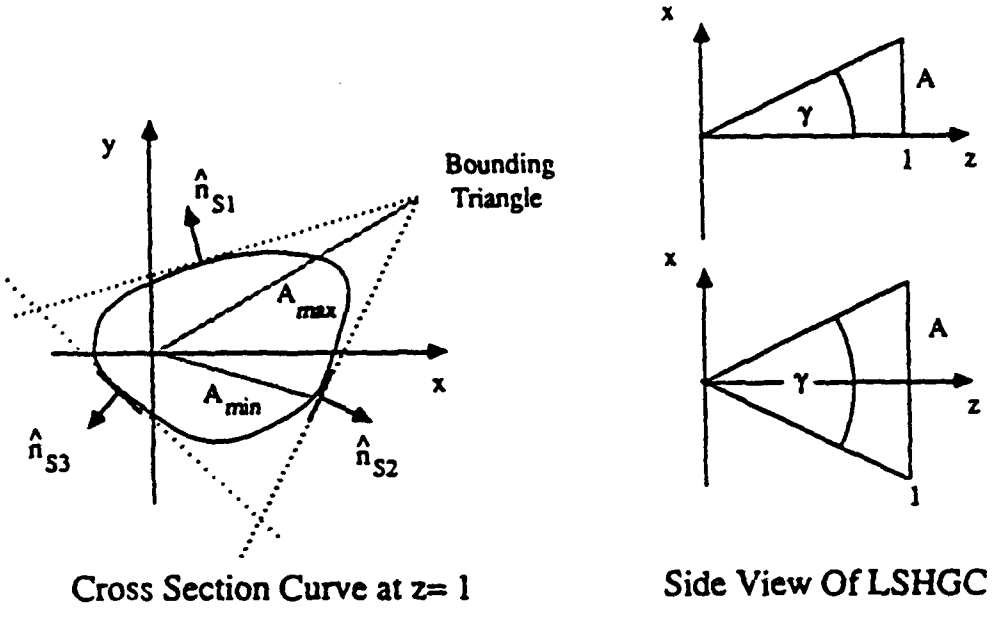


Figure 3.4. Constraints on Scale Parameter

there is a unique plane that contains both edges.

Table 3.1. Features for Orientation Solution.

Solve for ϕ and ψ	Restrictions
F_T or F_B	
$E_{T1}E_{T2}$ or $E_{B1}E_{B2}$	
$E_T E_B$	not parallel
$V_{T1}V_{T2}V_{T3}$ or $V_{B1}V_{B2}V_{B3}$	
$E_T V_T$ or $E_B V_B$	not coincident

Meridian edges are needed to solve for the origin of the cone. As for the orientation solution, features can be combined to give equivalent meridian edges. Since a side face corresponds to a straight line segment in the cross section, and a meridian edge corresponds to a vertex, it is apparent that the intersection line of two side planes F_{S1} and F_{S2} must also be a meridian, even if the sides are not adjacent. Therefore, three independent planar contacts on the sides of the LSHGC will define three meridian lines which must intersect at the origin. Table 3.2 gives combinations of features which can

Table 3.2. Features for Origin Solution.

Solve for $\underline{x}_o = (x_o, y_o, z_o)^T$	Restrictions
P_1P_2 or $E_{m1}E_{m2}$ or PE_m	not colinear
E_mF_S or PF_S	not coplanar
$F_{S1}F_{S2}F_{S3}$	none coplanar

be used to solve for the origin. Table 3.2 is not exhaustive. For example, corresponding vertices on the top and base of the cone define an equivalent meridian edge. The restrictions in Table 3.2 ensure the independence of the feature equations.

A solution for the cone orientation and origin requires one set of features from Table 3.1 and one set from Table 3.2. The minimum set of features is two meridian or parabolic edges or a meridian edge and a side face for the origin solution, and the top or bottom face for orientation. We are particularly interested in the minimum sets of features, because these are the only sufficient sets of features for a 3 finger hand with a single touch at each finger. If we are not fortunate enough to touch the cone and obtain one of the minimum sets of features, additional features will need to be identified until the feature combinations of Tables 3.1 and 3.2 are satisfied.

3.6. Necessary Feature Combinations

Previous sections have shown that 2 meridian edges and an end face are sufficient for cone origin and orientation determination, but are they necessary? Can orientation be determined or at least constrained without an end face or feature combination that gives an end face? We can show that no combination of parabolic points, side faces, or meridian edge points can determine orientation.

A side face, parabolic point, and meridian contain equivalent information about cone orientation. The surface normal at a parabolic point, eq. (3.16), gives one more equation than is obtained from a meridian edge, eq. (3.14). This equation specifies the orientation of the normal with respect to the meridian line. The parabolic point has 2 more equations and 2 more unknowns than the meridian edge, so there is no net gain in equations over the meridian edge. The surface normal orientation is a function of the derivative of the radius, ρ'. Without knowing ρ', which is not measured directly, the surface normal does not constrain the cone orientation.

A side plane provides similar information to an edge. A line coincident with this plane through the point of contact goes through the origin, just like a meridian edge. The surface normal of the plane is perpendicular to this line. Only one free parameter is required to specify orientation of a surface normal with respect to this line, hence a side face provides one more equation than a meridian edge. This surface normal is generally not in the cross-section plane, and does not provide any orientation constraint on the cone.

By the above discussion, we have shown an equivalence between meridian edges and parabolic lines in relation to the number of independent equations and parameters they provide. [Fearing, 1987] shows that no number of side edge contacts will allow the cone orientation to be determined.

Thus each meridian edge gives just 3 independent equations, but introduces three new unknowns ρ_i, θ_i, and z_i, in addition to the unknown orientation parameters ϕ and ψ. For N contacts, we have $3N$ equations, but $3N+2$ unknowns. Therefore, no unique orientation is obtainable. Similarly for parabolic points and side faces, the orientation parameter can not be determined. This result agrees with Shafer and Kanade's Pivot Theorem [1983]:

> A non-degenerate SHGC can be described as another SHGC with a different, intersecting axis, and the same cross section planes if and only if it is Linear.

In other words, the cross-section can not uniquely specify the cone axis; a linear SHGC can have an alternate axis with the same cross-section. Shafer's equivalent LSHGC's can have different beveled ends, which implies that one can't specify a unique right LSHGC until the end is sensed.

We have shown that side contacts can not specify the cone orientation. The end faces specify the cone orientation. Can end faces determine anything about the cone origin? Consider the orientation determined by a contact on F_T, and additional contacts at vertices on the top end, given by:

$$\underline{\tilde{V}}_{Ti} = z_T \mathbf{R}\left[Ax_i, Ay_i, 1\right]^T + \underline{x}_o \tag{3.26}$$

where **R** is known from the surface normal direction, and $\underline{\tilde{V}}_{Ti}$ is the sensed vertex position. There are 2N + 1 equations from N vertices (all z_T coordinates are the same), but 2N + 4 unknowns, Ax_i, Ay_i, z_T and $\underline{x}_o$. Thus contacts on just one end cannot be used to determine anything about the origin, other than that it must be above F_T. The top

contacts also provide no constraint on the scale parameter A, since the cross section scale depends on the distance of the top end from the origin. Vertices corresponding to the same point in the cross-section on top and bottom ends of the cone define side edges, and two side edges will define the origin.

3.7. Feature Consistency Matching/Labelling for LSHGC

We proceed in this section on the assumption that we are touching an object that is not a cylinder. Feature constraints for distinguishing a cone from a cylinder are given in [Fearing, 1987]. This section examines the combinations of one face and two edge contacts, and the combination of two faces and one edge. The edges are considered to be independent of the faces, that is, not coplanar. These are the minimum sets of three features, not including parabolic points, for determining the origin and orientation of the LSHGC. In section 3.4, we assumed that the feature set had been classified. We need to determine whether an edge is a meridian or end edge, and whether a face is a side or an end. This is not always possible. If this set violates no LSHGC geometric constraint, that is, it is consistent, it will be used to solve for the origin and orientation of the LSHGC.

If we have 2 parabolic points and one face, it is simple to determine whether the face is an end or side. The meridian lines at the parabolic points intersect at the origin. If the face plane intersects the origin, the set of features must be $F_S P_1 P_2$ and will not give an orientation solution. Otherwise we must have $F_T P_1 P_2$ which gives both the origin and the orientation of the cone.

3.7.1. Interpretation of Face and 2 Edges

Three finger sensors give simultaneous contact measurements on the object, and they are classified by curvature into a planar contact (F) and two edge contacts (E_1, E_2). It is now required to use the geometric constraints of the cone to determine whether these features are consistent with an end face and two meridian edges, which is a sufficient condition for determining the LSHGC. The measured feature set is checked for consistency with $F_E E_m E_m$, where F_E is an end face.

There are three face types ($F_T, F_S, or\ F_B$) and 3 possible edge types (E_T, E_B, or E_m). The total possible number of combinations of 1 face and 2 edges is:

$$\begin{bmatrix} 3 \text{ face types} \\ 1 \text{ sensed face} \end{bmatrix} \begin{bmatrix} 3 \text{ edge types} + 2 - 1 \\ 2 \text{ sensed edges} \end{bmatrix} = 3\frac{4!}{2!2!} = 18 \tag{3.27}$$

Lacking a more elegant approach, the 18 possible combinations are examined individually. There is an initial reduction in combinations because we need only consider combinations with intersecting edges. All meridian edges of a cone must intersect as was shown in section 3.4.1. If the edges do not intersect, we have an insufficient

measurement set, and need to explore the object further. An example of this is a set of features $F_T E_B E_T$ or $F_B E_B E_T$.

Table 3.3. Consistency of FEE with $F_{End} E_{m1} E_{m2}$

Case	Combination	Violates Constraint
I	$F_T E_B E_B$ $F_T E_B E_m$ $F_B E_T E_T$ $F_B E_T E_m$	meridian edge can not be parallel to end face
II	$F_S E_{m1} E_{m2}$	apex can not be on plane F_T
III	$F_S E_m E_T$	contact must be below apex
IV	$F_S E_{T1} E_{T2}$	angle at base must be < 90° (valid if cross section contains origin)
V	$F_T E_{m1} E_{m2}$	satisfies all requirements
V	$F_B E_{m1} E_{m2}$ $F_S E_B E_m$ $F_S E_B E_B$ $F_S E_B P$	face could be side or base

Table 3.3 lists possible combinations of two edges and a face. A single parabolic point is treated as a meridian edge. We sense a face and two edges, where the two edges intersect, but are not coplanar with the face. The intersection and independence requirement eliminates combinations including $F_S E_T E_B$ and 3 more combinations each of $F_T E_T$ and $F_B E_B$. Thus Table 3.3 lists only the remaining 11 combinations. We do not yet know the proper labelling for the sensed features. Let us suppose that one of the edges is perpendicular to the sensed surface normal of the face. From Case I of Table 3.3, this violates the constraint that a meridian edge is not parallel to an end face. We now know that we do not have two meridian edges, and need to sense one more. But we have found the top and bottom of the cone, because we know that the face and edge correspond to $F_T E_B$ or $F_B E_T$.

Cases I through IV are determined by simple geometric considerations; the details can be found in [Fearing, 1987]. All but 4 edge-face combinations are eliminated by the constraints, of which 3 are undecidable; a face can be either a base or a side.

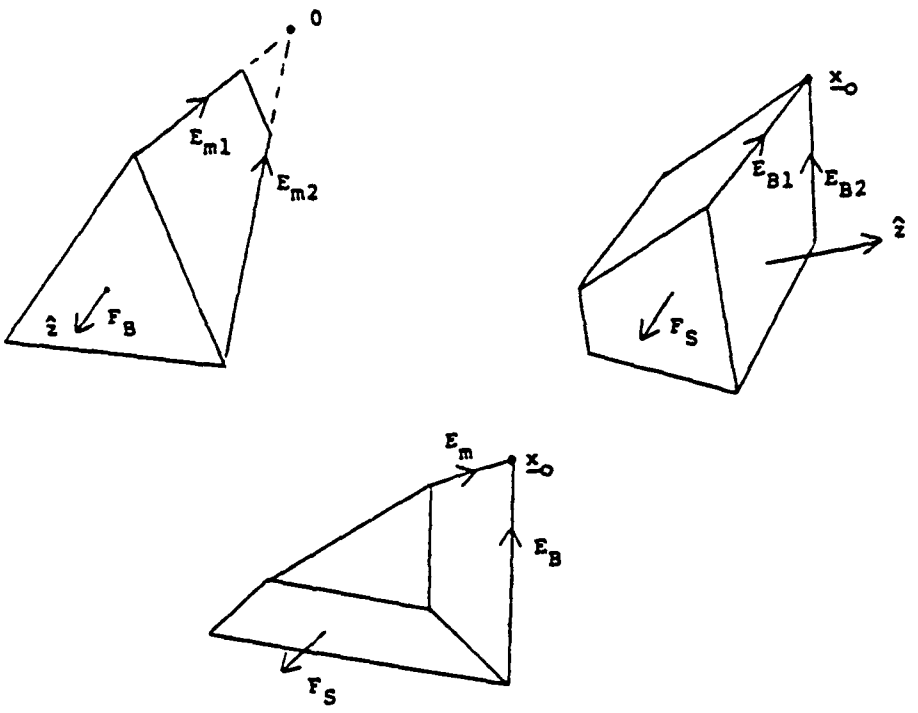


Figure 3.5. Multiple Cones for $F_B E_{m1} E_{m2}$

Case V:

There are four possible interpretations of one face and two edges (out of 18 total) that satisfy all the LSHGC constraints for $F_E E_m E_m$, the desired set. One of these, $F_T E_{m1} E_{m2}$, is unique because the origin is above the surface normal of the face. This can only happen if the face is the top of the cone. As for Cases III and IV, a side face can be interpreted as the base end, but never as the top end. Therefore, if the features $F_B E_{m1} E_{m2}$ are sensed, there are three possible consistent interpretations:

$F_B E_{m1} E_{m2}$: the desired result.

$F_S E_B E_m$: the measured face is a side face. The origin is given by the intersection of the side face and the meridian edge, but the orientation is not determined.

$F_S E_B E_B$: the measured face is a side face. The edges give the base plane and the orientation, but the origin is not determined. Figure 3.5 gives an example of the multiple consistent interpretations of the base face and two meridian edges. A parabolic

point in the set $F_S E_B P$ does not prevent misinterpretation. The orientation is not uniquely determined without the top face surface normal.

3.7.2. Interpretation of 2 Faces and an Edge

We consider now the combination of 2 faces and an edge. There are twelve possible combinations of two faces and one edge, but fewer combinations with the edge independent of the face and other restrictions. To solve for the origin and orientation of the LSHGC, the desired sets of sensed features are $F_T F_S E_m$ or $F_B F_S E_m$. For the desired features, the edge intersects both faces, and the edge is not coplanar to a face.

Table 3.4. Consistency of FFE with $F_{End} F_S E_m$

Case	Combination	Violates Constraint
I	$F_T F_S E_B$ $F_B F_S E_T$	$E \cdot \hat{n} \neq 0$ meridian edge can not be parallel to end face
II	$F_S F_S E_m$	apex can not be on plane F_T
V	$F_T F_S E_m$	satisfies all requirements
V	$F_B F_S E_m$ $F_S F_S E_T$ $F_S F_S E_B$	face could be side or base

As in the previous section, we measure two faces and one edge, and test this set for consistency with the desired set of features using the LSHGC constraints. Table 3.4 summarizes the constraints that can be used to distinguish a set of $F_E F_S E_m$ from the insufficient sets of two faces and one edge.

Case V:

Except for the combination $F_T F_S E_m$ which is correctly interpreted, all the other combinations can have a side face interpreted as the base. By similar reasoning as in case V of Table 3.3, the other 3 combinations are ambiguous. Since a face and an edge will intersect unless they are parallel, there is a much looser constraint from a face and an edge than from two faces. For example, the contact points will not be above the apex in the combination $F_S F_S E_T$ as they are in $F_S E_m E_T$, which is the roughly corresponding feature combination.

3.8. Using Partial Constraints

If the first 3 contacts are not adequate, we want to know where to move the fingers. The convexity assumption can be used with the partial constraints to choose finger paths that are very likely to make further contact with the cone. An insufficient set of features must be correctly labelled to use partial constraints. For example, with an edge parallel to a face and an object that is not a cylinder, we know from case I of Table 3.3 that the face is an end, and the edge must be on the other end. In this case we have the orientation as a consolation prize, but the length of the cylinder as a bonus. Tables 3.1 and 3.2 give necessary features for partial solutions to orientation or the origin of the cone. For example, given $F_S F_S F_T$, one obtains a line of points for the origin location.

Because of the cylindrical finger shape, we do not get a constraint at vertices or edges that can tell us which half space the object is in. A half-space constraint would help to localize the object with the three fingers. Without higher order edge sensing, to get the angles of planes that join to form an edge, or the shape of two surfaces intersecting, we cannot determine the attitude of the local surface from single contacts at curvature discontinuities.

3.9. Conclusions

Contrary to expectations, the LSHGC (a cone) is harder to characterize from local sensing with three contacts than was expected. There are only a few feature combinations that will distinguish a cone from a cylinder. There are only a few combinations of features that can be unambiguously identified and uniquely specify the cone's orientation and origin. The only unique solutions require sensing 1) the top end of the cone and two meridian edges or parabolic points, 2) the top end of the cone, a side face, and a meridian edge, or 3) the base end of the cone and two parabolic points. The scale parameter is unobtainable without sensing over the whole cross-section.

A similar analysis could be used to find the axis of a surface of revolution with local tactile sensing. In some ways it may be simpler to identify features on an SHGC without a linear sweeping rule: faces are unique, and must occur on the ends.

4. A Tactile Servo System

In principle, the results of the previous section allow us to determine the orientation, origin and size of an object with a three fingered hand, given at least 3 suitable contact locations. This section considers the simplified condition where an object has been previously grasped by a three finger hand, and the goal is to reorient this object in the hand. This capability is required when changing from an acquisition grasp to a grasp used for insertion, or a grasp used for stable transport as the hand moves through space. A control system using tactile feedback to reorient unknown objects reliably in the hand is outlined. This example demonstrates directions for future work, and shows how the work in this paper contributes to the dextrous manipulation problem.

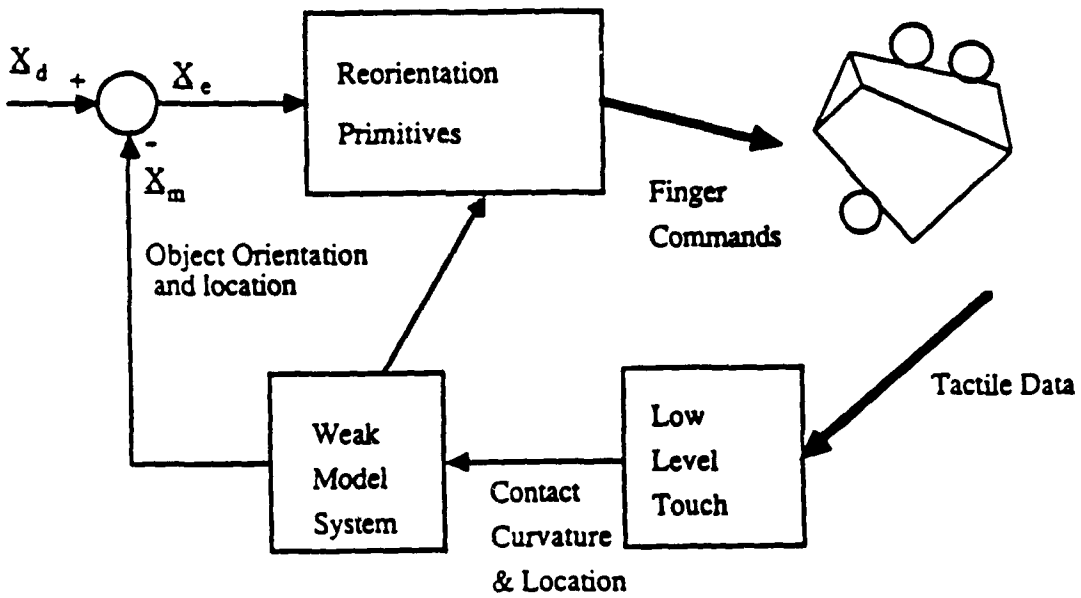


Figure 4.1. Proposed Dextrous Manipulation Servo System

4.1. A Tactile Servo System

Fig. 4.1 shows a high level block diagram for a proposed dextrous manipulation system. Consider an unknown cone (of LSHGC form from Section 3) stably grasped by a three finger hand. Input commands to the system may specify regrasping the object so that the axis of the LSHGC is normal to the palm of the hand. There are many common industrial objects that can use this axis description. For example, in attaching a nut to a bolt, the orientation of the nut can be specified without prior knowledge of the nut's diameter or length. This ability would give a flexible assembly system that would not need specific modelling or reprogramming for different nuts.

Tactile information from the fingers of a dextrous hand can be interpreted to yield contact curvature and location. Important object features such as vertices, edges, planes, and curved surfaces can be characterized by curvatures, surface normals, orientations and locations. At a lower level than is shown in this figure, contact force information from the fingers could be used in a faster servo to control finger and object forces during contact.

Contact features from multiple fingers can be interpreted in a Weak Model System to determine cone orientation and origin, as hinted at in Section 3. The weak model system has two functions, to continuously update the estimate of the position of the object with respect to the hand, and to refine estimates of the global properties of the object such as its size, ends, and cross section. With multiple continuous measurements, the data set is no longer sparse, and sensor uncertainty models can be used to get more reliable model estimates. For handling objects, the orientation of the cross section curve can be important, thus the sensed object position is specified by a six vector $\underline{X}_m$. The orientation about the cone axis can be specified relative to some sensed feature. In Section 3, this angle was unspecified, because the cross-section was not predetermined. The weak model system could also be used in developing exploration strategies for determining object properties before the object is grasped.

The input to this dextrous manipulation system would be $\underline{X}_d$, the desired location of the end of the cone, and its axis orientation in the hand. The system should also control the forces of objects in contact, such as during insertion. Relative rotations should also be specifiable, such as turning a nut on a bolt.

The relative position error $\underline{X}_e$ is input to the reorientation subsystem, which needs to plan a sequence of object rotations and translations to reposition the object. A successful plan will depend on coefficients of friction, finger workspace limitations, the angle of the apex of the cone, the size and ends of the object, and the shape of the cross

section. This plan requires the global object properties determined by the weak model system.

Finger force strategies depend on the object shape. For example, in regrasping a circular cone with a large apex angle with a three finger hand, the object must be stably grasped with two fingers while the third finger is repositioned to a new grasp location. Finger force strategies need to include gravitational force, which can sometimes be used as a "fourth finger". It is important to position the two fingers along a diameter to maintain grasp stability with low friction coefficients. The weak model information may also indicate that better grasp points are available to improve reorientation.

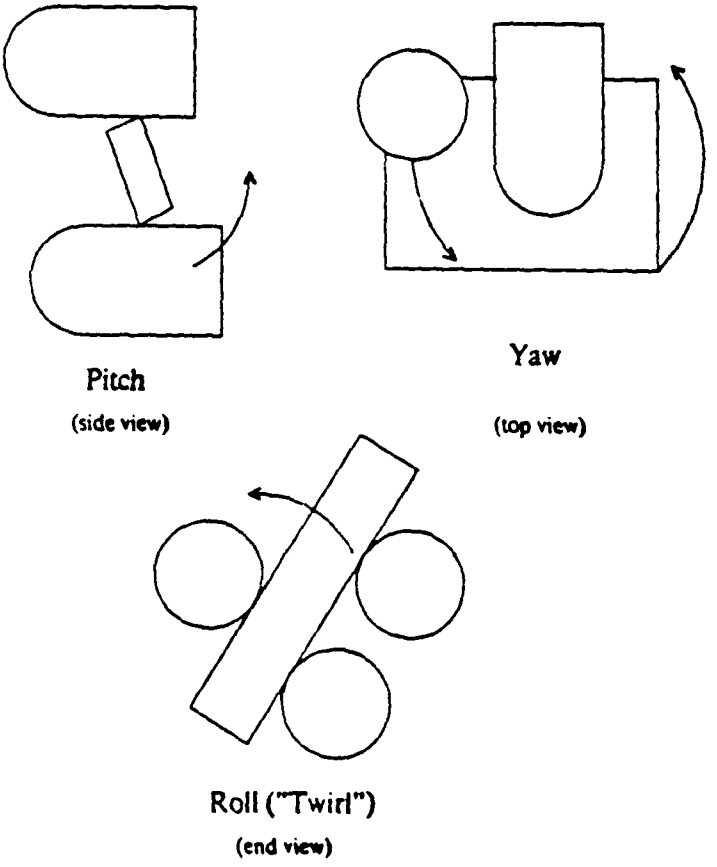


Figure 4.2. 3 Basic Reorientations

There has been little implemented work on motion operations with dextrous hands. Kobayashi, [1985] used a force control system to control local object motions with fingers in constant grasping locations. Reorientations where objects need to be

regrasped, have been shown for rotating a bar through many revolutions using preprogrammed finger position commands [Okada, 1982], and by [Fearing, 1986] using a sequence of applied forces at each finger. A useful set of re-orientation primitives for simple objects like blocks, is shown in Fig. 4.2. Regrasping objects with a simple parallel jaw gripper has been demonstrated by [Tournassoud et al, 1987]. Clearly, much work remains in developing regrasping/reorientation operations.

The reorientation sub-system would command finger force, stiffness and position. High speed tactile or force sensing can monitor the grasping operation for unexpected forces, contact slip, and loss of contact. The reorientation planner may also continuously monitor object position during reorientation, and develop error recovery strategies if unexpected object loads are encountered.

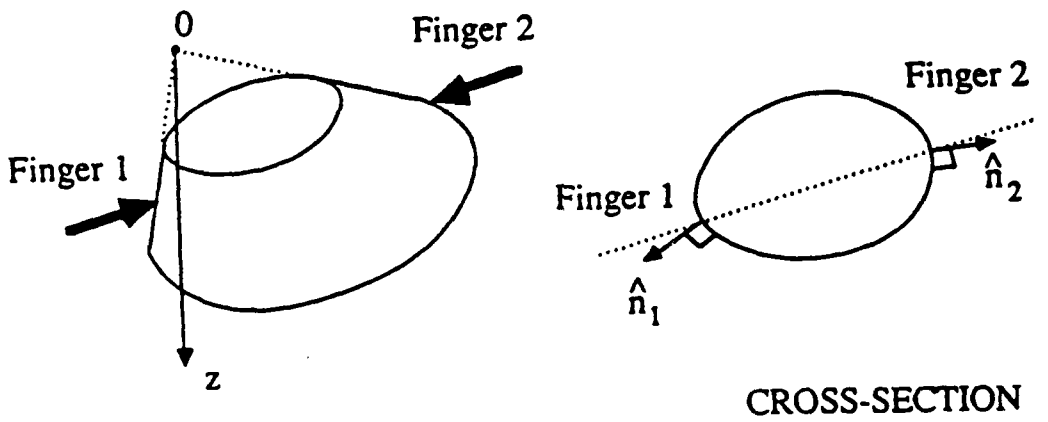


Figure 4.3. Grasping Cone with Two Fingers

The right circular cone is the most difficult kind of LSHGC to grasp. Fig. 4.3 illustrates the problem of grasping a non-circular cone. With two fingers, contacts must be placed nearly diametrically opposite to each other to get colinear surface normals in the cross-section plane, and thus finger forces within the friction cones. There is high slip potential for this shape; as the fingers slip, surface normals at the finger contacts get further away from colinear, causing greater slip. For a non-circular cross-section, there are always at least 2 pairs of colinear surface normals [Jameson, 1985]. One of these pairs will give a more stable grasp with two fingers. A circular cross-section has equally unstable grasping for every diameter.

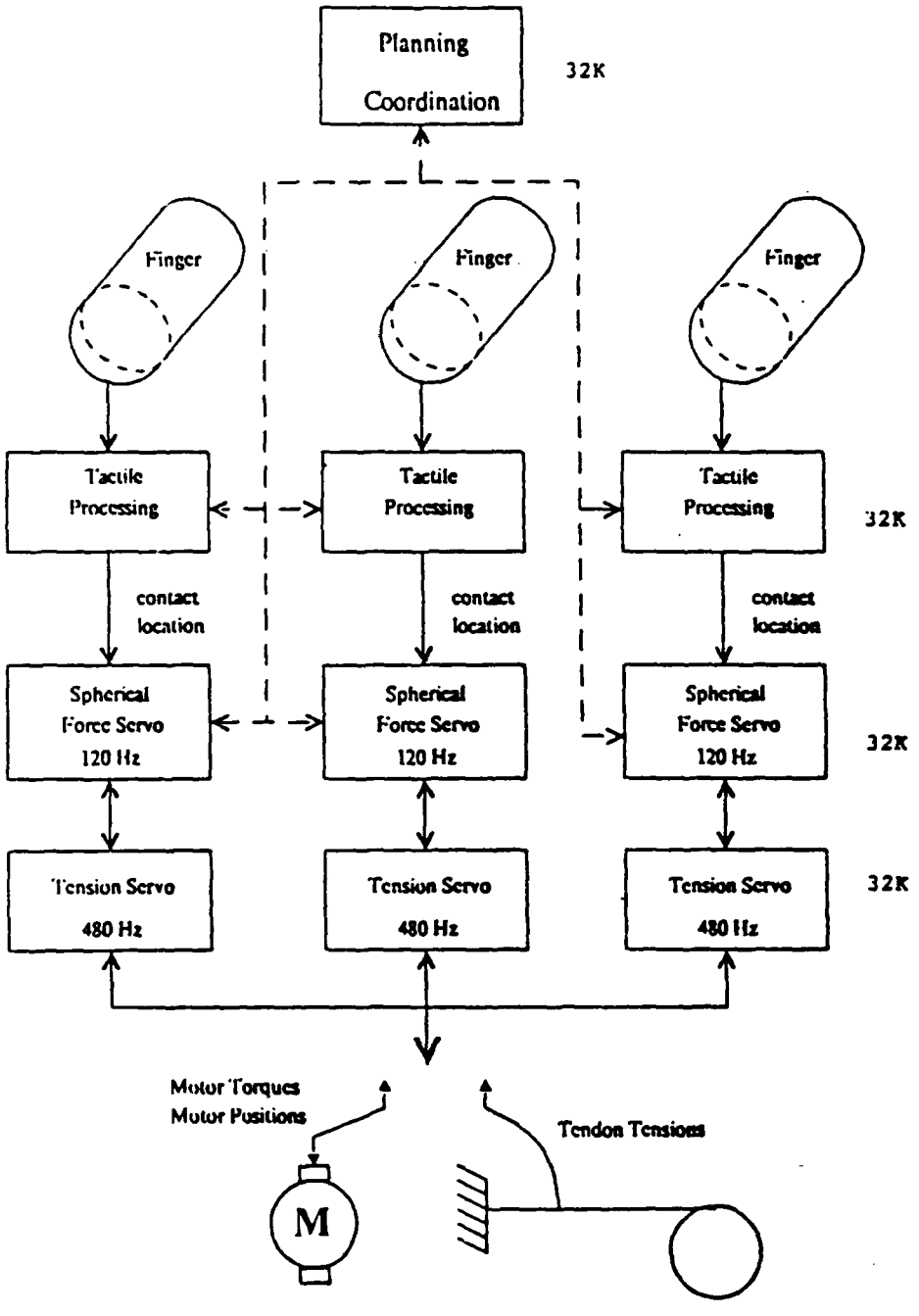


Figure 4.4. Proposed Architecture for Dextrous Manipulation System

The proposed high level dextrous manipulation system of Fig. 4.1 requires a large amount of computational power both for force control for fingers, and sensor processing. The NYMPH multiprocessor system has been implemented at Stanford for this task [Chen et al 1986a, 1986b]. Fig. 4.4 shows a proposed processor arrangement using 10 National 32016 32-bit computers. A 7-processor version for hand control

without tactile processing has been tested for reorienting objects in one plane ("TWIRL"), and work is in progress at incorporating three additional processors for tactile-based object control. At the lowest level, two processors are used for each finger for a force control system in spherical coordinates [Fearing, 1986]. At the highest level, a planning and coordination processor directs the sequencing of finger force commands.

5. Conclusion

We have shown that only a few specific combinations of three features on an LSHGC are sufficient to determine the cones orientation and origin. Some combinations of sensed features will partially constrain the object's origin. Other combinations of features will constrain the object to be in one of three distinct orientations. Features such as edges and faces are ambiguous; they could be on the sides or ends of the cone. It is possible to determine the labellings of these features that are consistent with the geometric constraints. To simplify the labelling problem, only sets of contact features that are sufficient to determine the cone attitude and location were considered. Expressions were derived for various surface features such as edges, vertices, planes, and curved regions. The origin and orientation of the cone can be found from a set of necessary and sufficient combinations of labelled features.

If three contacts are not consistent with touching a cylinder, the features are examined for consistency with the desired set of contact features (one end contact, and two side contacts), using geometric constraints for the cone class of objects. When a unique labelling is possible, the cone origin and orientation will be determined.

A dextrous manipulation system that could manipulate a priori unknown objects was described as a motivating reason for using weak models for shape interpretation.

Acknowledgements

Many thanks to Jean Ponce and Tom Binford for many discussions and insightful comments on LSHGCs and interpretations. This work was supported at Stanford University by DARPA contract MDA903-86-K-0002.

References

[1] P. Allen and R. Bajczy, "Object Recognition using Vision and Touch," 9th Intern. Joint Conf. on Artificial Intelligence, Los Angeles, August, 1985.

[2] P. Allen, "Sensing and Describing 3-D Structure", IEEE Intern. Conf. on Robotics and Automation, San Francisco, CA, April 1986.

[3] Begej Corporation, "Model FTS-2 Fingertip-shaped Tactile Sensor," Technical Bulletin #2, October 1986.

[4] T.O. Binford, "Visual Perception by Computer", Proc. IEEE Conference on Systems and Control, Miami, FL, Dec. 1971.

[5] M. Brady, J. Ponce and A. Yuille, "Describing Surfaces", Proc. of the 2nd International Symposium on Robotics Research, Kyoto, Japan, August 1984.

[6] M.R. Cutkosky, "Mechanical Properties for the Grasp of a Robotic Hand", CMU Robotics Institute Tech. Report CMU-RI-TR-84-24, 1984.

[7] R.E. Ellis, "Acquiring Tactile Data for the Recognition of Planar Objects", IEEE Intern. Conf. on Robotics and Automation, Raleigh, NC April 1987.

[8] O.D. Faugeras and M. Hebert, "The Representation, Recognition, and Positioning of 3-D Shapes from Range Data," *Intern. Jnl. of Robotics Research*, vol. 5, no. 3, Fall 1986.

[9] R.S. Fearing, "Simplified Grasping and Manipulation with Dextrous Robot Hands", American Control Conference, San Diego, CA, June 1984.

[10] R.S. Fearing, "Implementing a Force Strategy for Object Re-orientation," IEEE Intern. Conf. on Robotics and Automation, San Francisco, CA April, 1986.

[11] R.S. Fearing, A. Rise, and T.O. Binford, "A Tactile Sensing Finger Tip for a Dextrous Hand," SPIE Conference on Intelligent Robotics and Computer Vision, Cambridge, MA October 1986.

[12] R.S. Fearing, "Tactile Sensing, Perception, and Shape Interpretation", Ph.D. Thesis, Dept. of Electrical Engineering, Stanford University, Dec. 1987.

[13] R.S. Fearing and T.O. Binford, "Using a Cylindrical Tactile Sensor for Determining Curvature," IEEE Intern. Conf. on Robotics and Automation, Philadelphia, PA, April 1988.

[14] P.C. Gaston and T. Lozano-Perez, "Tactile Recognition and Localization Using Object Models: the Case of Polyhedra on a Plane", M.I.T. A.I. Memo 705, March 1983.

[15] W.E.L. Grimson and T. Lozano-Perez, "Model-Based Recognition and Localization from Sparse Range or Tactile Data", *Intern. Jnl. of Robotics Research*, vol. 3, no. 3, Fall 1984.

[16] V.S. Gurfinkel, A. Yu. Shneyder, Ye. M. Kanayev, and Ye. V. Gurfinkel, "Tactile Sensitizing of Manipulators", *Engineering Cybernetics*, vol. 12, no. 6, Nov. 1974.

[17] H. Hanafusa and H. Asada, "Stable Prehension by a Robot Hand With Elastic Fingers," *7th ISIR*, Oct. 1977.

[18] W.D. Hillis "Active Touch Sensing," S.M. thesis, M.I.T. Department of Electrical Engineering and Computer Science, January 1981.

[19] J.M. Hollerbach, "Workshop on the Design and Control of Dexterous Hands," MIT AI Memo 661, April 1982.

[20] S.C. Jacobsen, J.E. Wood, D.F. Knutti, K.B. Biggers, and E.K. Iversen, "The Version I Utah/MIT Dextrous Hand," in *Robotics*, H. Hanafusa and H. Ione, Eds. Cambridge, MA: MIT Press, 1985.

[21] J.W. Jameson, "Analytic Techniques for Automated Grasping ", Ph.D. Thesis, Dept. of Mechanical Engineering, Stanford University, June 1985.

[22] Z. Ji, "Dexterous Hands: Optimizing Grasp by Design and Planning", Ph.D. Thesis, Dept. of Mechanical Engineering, Stanford University, June 1987.

[23] J.R. Kerr, "An Analysis of Multi-Fingered Hands", Ph.D. Thesis, Dept. of Mechanical Engineering, Stanford University, 1985.

[24] G.I. Kinoshita, A. Aida, and M. Mohri, "A Pattern Classification by Dynamic Tactile Sense Information Processing", *Pattern Recognition*, vol. 7, pp. 234-251, 1975.

[25] G.I. Kinoshita "Classification of grasped object's shape by an artificial hand with multi-element tactile sensors" *IFAC Symposium on Information Control Problems in Manufacturing Technology*, Tokyo: pp. 111-118. 1977.

[26] H. Kobayashi, "Control and Geometrical Considerations for an Articulated Robot Hand," *International Journal of Robotics Research*, vol. 4, no. 1, Spring 1985.

[27] C.S. Loucks, V.J. Johnson, P.T. Boissiere, G.P. Starr, J.P.H. Steele, "Modelling and Control of the Stanford/JPL Hand", IEEE International Conf. on Robotics and Automation, Raleigh, NC, April, 1987.

[28] J. Malik, "Interpreting Line Drawings of Curved Objects", Ph.D. Thesis, Dept. of Computer Science, Stanford University, Dec. 1985.

[29] T. Okada and S. Tsuchiya, "Object Recognition by Grasping", *Pattern Recognition*, vol. 9, pp. 111-119, 1977.

[30] T. Okada, "Computer Control of Multijointed Finger System for Precise Object-Handling," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-12, no. 3, May/June 1982.

[31] H. Ozaki, S. Waku, A. Mohri, and M. Takata, "Pattern Recognition of a Grasped Object by Unit-Vector Distribution," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-12, no. 3, May/June 1982.

[32] J. Ponce, D. Chelberg, W. Mann, "Invariant Properties of Straight Homogeneous Generalized Cylinders and their Contours", submitted to IEEE PAMI, 1987.

[33] J.K. Salisbury and J.J. Craig, "Articulated Hands: Force Control and Kinematic Issues," *International Journal of Robotics Research*, vol. 1, no. 1, Spring 1982.

[34] S. A. Shafer and T. Kanade, "The Theory of Straight Homogeneous Generalized Cylinders", Carnegie Mellon Univ. Comp. Sci. Tech. Rept. CMU-CS083-105, 1983.

[35] D.M. Siegel, "Contact Sensors for Dextrous Robot Hands", MIT AI Lab Technical Report 900, June 1986.

[36] T.H. Speeter, "Analysis and Control of Robotic Manipulation", Ph.D. Thesis, Dept. of Biomedical Engineering, Case Western Reserve University, Jan. 1987.

[37] R. Tomovic and G. Boni, "An Adaptive Artificial Hand," *IRE Trans. on Automatic Control*, vol. AC-7, no. 3, April 1962.

WHAT KIND OF SENSORS ARE REQUIRED FOR DEXTEROUS FINGERS ?

Tokuji Okada

Mechanical Engineering Department
Niigata University
8050 2-no-cho, Ikarashi
Niigata 950-21, Japan

Abstract

Requirements of sensors for dexterous fingers have been discussed and appropriate sensors to answer the requirements have been proposed.

Miniaturization of a tactile sensor element has been considered, however the number of sensors which might be attached to the surface of a finger body is limitted. Also, it is difficult to make all of the surface sensitive and to handle a bundle of signal lines. In order to solve these problems, new tactile sensor having no blind sector is proposed. The sensor uses a suspension shell covering the surface of the finger body. Tactile sense is obtained by detecting the relative displacement of the suspension shell from the finger body.

1. Introduction

Visual information is important to control a robot hand in a position which is very close to an object [1],[2],[3]. On the other hand, sensors attached to an end effector of the robot hand are characterized by the fact that the sensors are sensitive to invisible information like pressure, force and moment which are obtained by touching an object. For instance, it becomes clear by the touch whether an object is simply located on or fixed to another object. Once the hand reached the object, the invisible information is needed. This information can be said local or micro information as compared with global or macro information of the visual information. Specifically, in complicated assembly tasks, tactile information is indispensable for a fine control of the hand, in addition to information about geometrical relationships among assembly parts.

When the sensor detects unexpected signal while the hand moves, we can recognize an obstacle invasion or disturbance caused by abnormal task completion, and can revise or make a change in programs to cope with the unexpected situation. Unusual programs will be also adopted according to demand. Necessity of the sensor is not only in gathering information for the control of the hand but also in recognizing size and shape of an object. For the purpose of recognition, the hand is actively scanned to be a probe [4]. And the information from the sensor is synthesized with control history of the hand position to search, verify and recognize the object. For instance, the hand can grope for a specific object form and pick it up from a box in which a variety of objects in size and shape exist. Other attributes of the object like hardness and ruggedness are also evaluated by synthesizing the sensor information. Thus the information is very important to determine grasp force of the hand so that an fragile object might be handled without crashing or dropping down.

Further, sensors are utilized to reduce a load of computation since they get information directly. Since real data makes it easy to realize a fine control of the hand, sensor-based control programs will give the robot high adaptability to the real world (i.e. autonomy), with high accuracy and reliability.

In this paper, we discuss tactile sensors which are required for dexterous fingers and we propose an appropriate sensor. Kinematical analysis and measurement principle are presented.

2. Requirements of Sensors for Dexterous Fingers

Various tactile sensors of opto-electrical [5],[6],[7], magnetic, electrostatically capacitive, mechanical and electro-conductive means have been proposed so far [8],[9],[10],[11], [12]. A strain gauge and a load cell have been suggested as a device for measuring and resolving forces and moments applied thereon into their orthogonal components [13], [15]. The device may be accomplished by connecting spring elements to the loaded input and measuring the angular and translational displacements as a measure of forces and moments.

Generally, the strain gauge is troublesome in handling because of its fragility. Thus an appropriate protection of the gauge from extraneous forces and moments must be considered [14]. Further, the measurement is considerably influenced by temperature. While electrical compensation techniques have been used in the past for correcting the signals influenced by temperature, it has been difficult to obtain reasonable signals to be measured. Similar things are said in the load cell, since the load cell is basically constructed by using the strain gauges with flexure beams, cantilever bars, tensiontraps and so on.

The load cell lacks the ease in manufacturing and assembly or the accuracy necessary to provide an inexpensive but accurate device. Multi-axis load cells through which vectors representing all components of forces and moments have a relatively large amount of cross-talk or undesired co-dependence [15]. Some load cells have a resonant frequency at a low frequency. Therefore, the fabrication of the device is rather difficult and expensive, leading to a comparatively expensive load cell. These are undesirable features existing in the strain gauge type and the load cell type devices.

on the fingers. Several difficulties arise. For instance, a large number of tactile elements have to be arranged in order to obtain tactile information over a wide range on the surface of the finger body [16],[17]. For this reason, insensitive regions inevitably arise actually [18]. Flexible rubber covers are used to make the insensitive regions small [19]. Moreover, it is difficult to reduce the size and weight of the sensor as a whole since a large number of elements are arranged in a discrete fashion [20], and signal processing increases in number with increase in the number of elements involved, leading to increased complexity in use.

From the above discussions, we can extract such sensor requirements for the dexterous fingers. These are

1)smart enough to be housed in a finger body, since the dexterous fingers can perform not only such simple motions as bending and extending but also such lateral flexing motions as adduction and abduction.
2)capable of measuring the magnitude, direction and point of action of the external force applied on the finger body without insensitive regions.
3)light weight which causes no error in a fine control of finger joints.
4)high sensitivity with robustness to endure the external force.
5)low drift against the change of the temperature,

and

6)reduced in number of electric lines by adopting an appropriate scanning of the sensor element.

To answer these requirements, we propose tactile sensors by taking such a fact into consideration that finger bodies have cylindrical forms.

3. Structure of Proposed Sensors

Most of the fingers get tactile information by electrical or mechanical sensor elements installed on a surface of a finger body. Appropriate sensor elements in number, size and arrangement might be considered according to a demend of accuracy and resolution. For instance, if the sensor is utilized to identify a partial shape of an object, a lot of sensor elements will be installed to make the resolution high. Therefore, in general, the finger is overvieved as shovn in Fig.1(a). Evidently, the surface of the finger becomes grotesque and disturbs flexible motion.

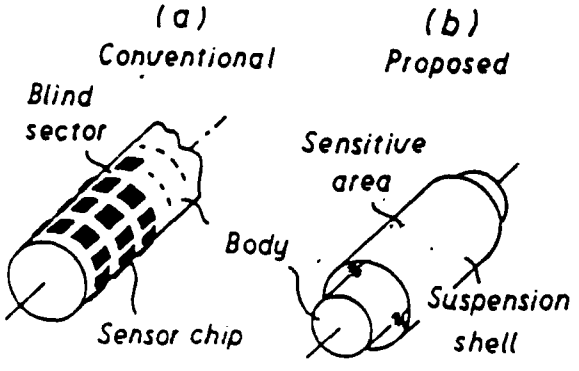


Fig.1. Sensor structure.

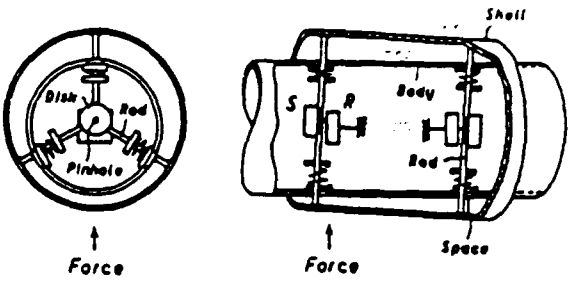


Fig.2. Inside viev of proposes sensors.

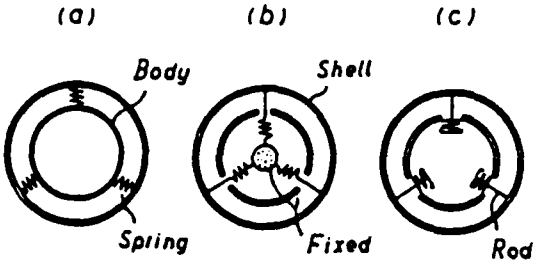


Fig.3. Three suspension types.

On the contrary, a finger installing the proposed tactile sensors is overvieved as shovn in Fig.1(b). That is, the finger body is veiled in a cylinder vhich ve call a *suspension shell*. The suspension shell is free to move axially and radially since materials connecting the shell to the finger body are elastic. The force operating on the finger body vill cause the shell move. Thus, the force vill be measured by analyzing radial, axial, and angular displacements of the shell. In order to make it possible to detect the displacement of the shell, ve use optical means located in the central area of the finger body, by taking advantage of the fact that the finger body is hollov, in general.

Figure 2 shovs rough sketches of a cross sectional and longitudinal views of a finger equipped vith a sensing unit. Symbols S and R mean a light beam source and a 2-D photosensor array, respectively. Notches express an elastic material like a spring. A form of the finger cross section is not limited bo be circular. Exact measurement of the deviation of the shell against the finger body is described later.

3.1 Suspension Types

In order to suspend the shell around the finger body coaxially, three suspension types are considered. Figure 3 shovs three cross sections of the sensors corresponding to the types A, B and C vhen three elastic materials are constructed by springs. Ends of each spring are connected vith the shell or the body, directly or indirectly through a rod. In the type A, the spring connects the shell and the body directly, thus the spring is housed in the gap betveen the body and the shell.

On the contrary, the springs in the types B and C are located in a space of the finger body by using the rod for connection. Dotted area in the type B is a part of the finger body. Central spaces are occupied for the suspension in the types B and C. Hovever, the space in the center of the body still remains in the type C.

Since the shell covering the finger body becomes sensitive to a force operated, tactile information is obtainable at all around the finger body. We investigate a stiffness of the sensor depending on the elasticity of a spring.

Let consider a spring (stiffness; k) crosssing the basic line χ with the angle α, then restoring force of the spring to the direction θ is expressed as $kr\cos(\alpha+\theta)$ for a displacement r (see Fig.4). Therefore, in such a system having a lot of springs coupled to the mass m, composite stiffness k_θ is written as

$$k_\theta=\sum_{i=1}^{n}k_i\cos^2(\alpha_i+\theta). \qquad (1)$$

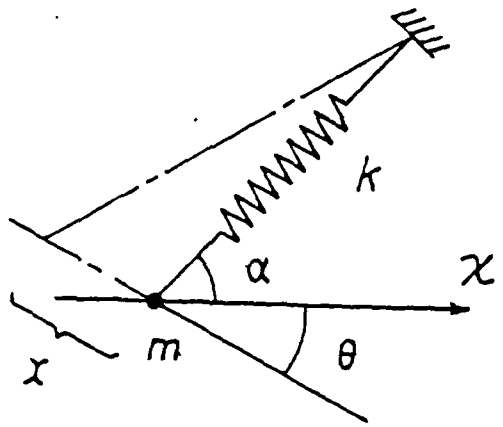


Fig.4. Installing condition of a spring.

By utilizing Eq.(1), we can calculate sensor stiffness around the finger axis. When springs are arranged so as to divide a circumference with equal angles, we obtain

$$k_\theta=(k_1+k_2)\cos^2\theta, \text{ when } n=2. \qquad (2a)$$

$$k_\theta=\frac{1}{4}(4k_1+k_2+k_3)\cos^2\theta+\frac{3}{4}(k_2+k_3)\sin^2\theta + \sqrt{3}/4(k_2-k_3)\sin 2\theta, \text{ when } n=3. \qquad (2b)$$

$$k_\theta=k_1+k_3+(k_2+k_4-k_1-k_3)\sin^2\theta, \text{ when } n=4. \qquad (2c)$$

Figure 5 shows spring arrangements and the stiffness patterns calculated. Regardless of the value of θ, the composite stiffness becomes constant when $n=3$ in $k_1=k_2=k_3$, that is, a circular pattern is formed. Same things appear when $n=4$ in $k_1+k_3=k_2+k_4$.

Above discussions are said when all springs are either extension springs or compression springs. If a spring is useful for both extension and compression, only one spring is enough to have the stiffness pattern as shown in Fig.5(a). When lateral stiffness of the spring is not negligible small, the stiffness pattern in Fig.5(a) becomes a single loop. Evidently, the spring is utilized for simplicity of a sensor mechanism.

4. Kinematic Analysis of the Sensors

In order to make it possible to sense tactile information from a wide area of a finger surface, we suspend a cylindrical shell at two sections A and B which are close to the ends of the shell (see Fig.6). The force operating on the shell will cause vibration and the sensor response unreliable. To

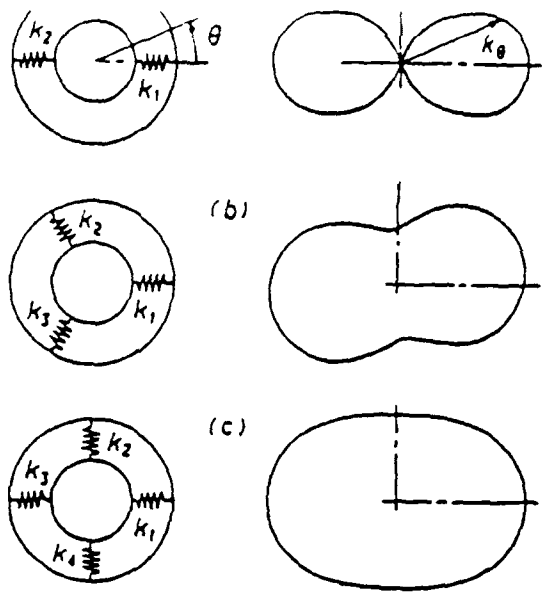


Fig.5. Characteristics of a stiffnes around a finger axis.

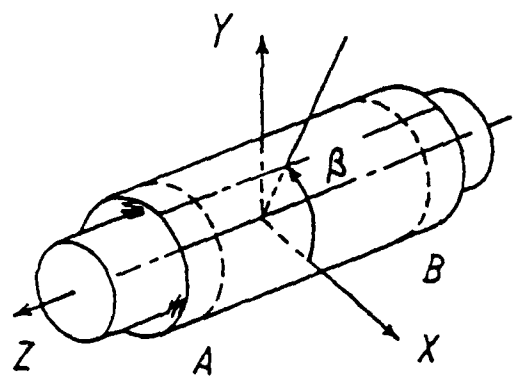


Fig.6. Specific plane for kinematical analysis.

reduce the vibration, it is very important to know a behavior of the shell, specifically the natural frequency. Therefore, we analyze the kinematics of the shell in this section by supposing that the shell has a circular form. At the beginning of the analysis, we define symbols.

4.1 Nomenclature

F: external force.
f_x, f_z, f_r: rectangular components of F.
a,b: subscript for suspension sections A and B.
G: gravity center of a suspension shell.
g: gravity.
h: position of the contact point along Z axis.
I: moment of inertia.
i: subscript for discriminating springs.
k: longitudinal stiffness of a spring.
k_r: lateral stiffness of a spring.
l_s: length of the suspension shell.
l_a: distance of the suspension section A from the suspension end (see Fig.8).
l_b: distance of the suspension section B from the

l_g: distance between G and ... the section A.
M: mass of a suspension shell.
m_a, m_b: division of the mass M.
r_o: length of a spring.
r_1: inner radius of the shell.
r_2: outer radius of the shell.
T: kinetic energy.
U: potential energy.
X,Y,Z: rectangular coordinate system.
α: angle for the spring arrangement (see Fig.7).
β: rotation angle about the Z axis.
δ: displacement of a spring length (from an off-load condition).
ρ: angular displacement of the shell about the body axis.
λ: displacement of G from a stable equilibrium (positive in downside).
λ_a: displacement of m_a.
λ_b: displacement of m_b.
φ: rotation angle about the axis passing through G and is parallel to X axis (positive in CCW).

4.2 Analysis of Sensor Behavior

Since behavior of the shell in arbitrary attitude is too difficult to analyze, such specific conditions are supposed that the finger axis is horizontal or vertical. In this section, we consider the former case, that is the finger axis is horizontal.

Let the shell be suspended by adopting the suspension type C in Fig.3. Note that analyzing plane crosses the X-Z plane with the angle β. relations among the shell, spring and finger body in the cross section A are illustrated in Fig.7.

Figure 8 illustrates the parameters in the analyzing plane. From the figure, we can express the kinetic energy as

$$T=\frac{1}{2}M\dot{\lambda}^2+\frac{1}{2}I\dot{\varphi}^2. \quad (3)$$

When the shell is a uniform medium, we can write

$$l_g=\frac{1}{2}l_a. \quad (4)$$

$$I=\frac{M}{12}(l_a^2+3(r_1^2+r_2^2)). \quad (5)$$

Concerning the displacement of the mass m_a and m_b, we have (see Figs.7 and 8)

$$\lambda_a=\lambda+(l_g-l_a)\varphi. \quad (6)$$

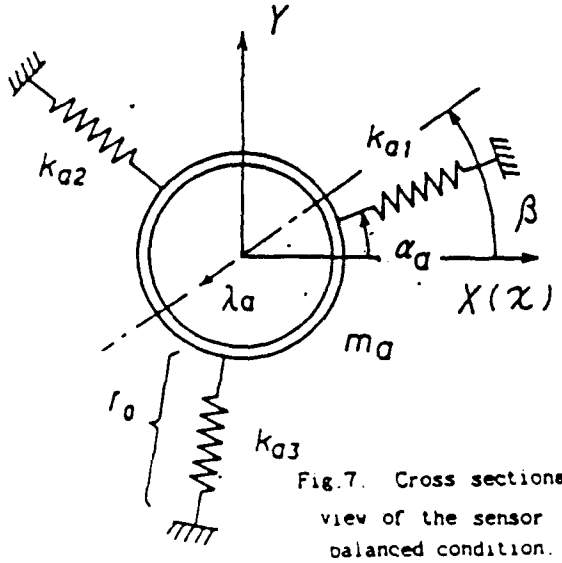


Fig.7. Cross sectional view of the sensor in balanced condition.

$\lambda_b=\lambda-(l_a-l_g-l_b)\varphi$.

Note that the potential energy of the spring at the section A is

$$U_{ai}=\frac{1}{2}k_{ai}(\delta_{ai}^2-(\delta_{ai}-\sqrt{(r_o\cos(\beta-\theta_{ai})+\lambda_a)^2+(r_o\sin(\beta-\theta_{ai}))^2}+r_o)^2). \quad (8)$$

In the same manner, we obtain the potential energy U_{bi} in the form

$$U_{bi}=\frac{1}{2}k_{bi}(\delta_{bi}^2-(\delta_{bi}-\sqrt{(r_o\cos(\beta-\theta_{bi})+\lambda_b)^2+(r_o\sin(\beta-\theta_{bi}))^2}+r_o)^2). \quad (9)$$

It follows that the potential energies of the mass m_a and m_b are

$$U_{ap}=-m_a g\lambda_a\sin\beta. \quad (10)$$

$$U_{bp}=-m_b g\lambda_b\sin\beta. \quad (11)$$

Collecting the above mentioned terms yields a total of the potential energies such that

$$U=\sum U_{ai}+\sum U_{bi}+U_{ap}+U_{bp}. \quad (12)$$

Consider the balanced condition of the spring, one can verify that

$$\sum k_{ai}\delta_{ai}\cos(\beta-\theta_{ai})=m_a g\sin\beta. \quad (13)$$

$$\sum k_{bi}\delta_{bi}\cos(\beta-\theta_{bi})=m_b g\sin\beta. \quad (14)$$

The dynamic equations of motion for the suspension system can be derived using Lagrange's equations:

$$\frac{\partial}{\partial t}\left(\frac{\partial T}{\partial \dot{q}_n}\right)-\frac{\partial T}{\partial q_n}+\frac{\partial U}{\partial q_n}=Q_n, \quad (15)$$

where

T total kinetic energy,
U total potential energy,
Q_n generalized external forces,
q_n generalized coordinates, and
$\dot{q}_n$ generalized velocities.

Therefore, insert Equations (3) and (12) into Eq.(15) to obtain

$$M\ddot{\lambda}+I\ddot{\varphi}+\sum\frac{\partial U_{ai}}{\partial\lambda}+\sum\frac{\partial U_{ai}}{\partial\varphi}+\sum\frac{\partial U_{bi}}{\partial\lambda}+\sum\frac{\partial U_{bi}}{\partial\varphi}=0. \quad (16)$$

Suppose that $\lambda_a/r_o<<1$, then we have

$$\frac{\partial U_{ai}}{\partial\lambda}\approx k_{ai}\frac{\lambda_a}{r_o}[\delta_{ai}-(\delta_{ai}+r_o)\cos^2(\beta-\theta_{ai})]+k_{ai}\delta_{ai}\cos(\beta-\theta_{ai}), \quad (17)$$

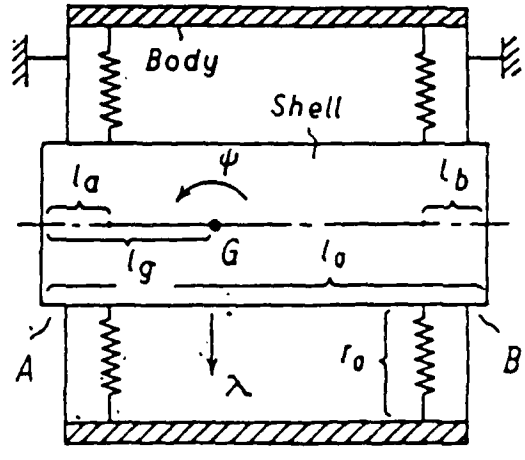


Fig.8. Side view of the sensor in balanced condition.

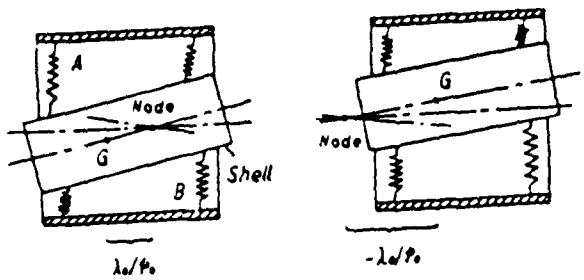


Fig.9. Sketches of the end of the finger body.

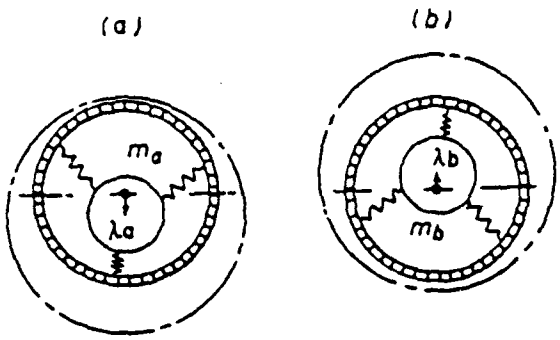


Fig.10. Vibration modes: (a)$\lambda_o l_\varphi \varphi_o > 0$ and (b)$\lambda_o l_\varphi \varphi_o < 0$.

$$\frac{\partial U_{ai}}{\partial \varphi} \approx (l_\varphi - l_a)\frac{U_{ai}}{\partial \lambda}. \tag{18}$$

Also, by supposing that $\lambda_o/r_o \ll 1$, we have

$$\frac{\partial U_{bi}}{\partial \lambda} \approx k_{bi}\frac{\lambda_b}{r_o}\{\delta_{bi} - (\delta_{bi}+r_o)\cos^2(\beta-\theta_{bi})\} + k_{bi}\delta_{bi}\cos(\beta-\theta_{bi}), \tag{19}$$

$$\frac{\partial U_{bi}}{\partial \varphi} \approx (l_\varphi + l_b - l_a)\frac{U_{bi}}{\partial \lambda}. \tag{20}$$

Using the approximation expressed by Equations from (17) to (20), one can find from Eq.(16) that

$$M\ddot{\lambda} + (u+v)\lambda + \{u(l_\varphi - l_a) + v(l_\varphi + l_b - l_a)\}\varphi = 0, \tag{21}$$

$$I\ddot{\varphi} + \{u(l_\varphi - l_a) + v(l_\varphi + l_b - l_a)\}\lambda + \{u(l_\varphi - l_a)^2 + v(l_\varphi + l_b - l_a)^2\}\varphi = 0, \tag{22}$$

where

$$u = \frac{1}{r_o}\sum_{i=1}^{3} k_{ai}\{\delta_{ai} - (\delta_{ai}+r_o)\cos^2(\beta-\theta_{ai})\}, \tag{23}$$

$$v = \frac{1}{r_o}\sum_{i=1}^{3} k_{bi}\{\delta_{bi} - (\delta_{bi}+r_o)\cos^2(\beta-\theta_{bi})\}. \tag{24}$$

An imaginable solution of Equations (21) and (22) is

$$\lambda = \lambda_o \cos\omega t, \tag{25}$$

$$\varphi = \varphi_o \cos\omega t. \tag{26}$$

Hence, we have

$$MI\omega^4 - [(u+v)I + M\{u(l_\varphi - l_a)^2 + v(l_\varphi + l_b - l_a)^2\}]\omega^2 + uv\{l_a - (l_a + l_b)\}^2 = 0. \tag{27}$$

Then we obtain the angular velocity ω. Furthermore, combining Equations (21) and (25) produces the result that

$$\frac{\lambda_o}{l_\varphi \varphi_o} = \frac{u(l_\varphi - l_a) + v(l_\varphi + l_b - l_a)}{(mw^2 - (u+v))l_\varphi}. \tag{28}$$

This result is also obtained by combining Equations (22) and (26).

Vibration modes are classified into three types depending where a node exists. Since we defined that λ and φ are positive in downside and counter clockwise, the node exists 1) on the point G, 2) on the right and 3) on the left, when the value of $\lambda_o/(l_\varphi \varphi_o)$ is zero, positive and negative, respectively. Deviation of the node from the point G is expressed by λ_o/φ_o. Figure 9 shows vibration modes when $\lambda_o/(l_\varphi \varphi_o)$ is positive and negative. Cross sectional behaviors are shown in Fig.10. To compress the vibration of the sensor, the angular velocity ω is desired to be designed small.

5. Displacement Measurement Mechanisms of the Sensors

In this section we describe displacement measurement mechanism for two cases depending on the measurement accuracy. Characteristics of the sensor is also presented.

5.1 Rough Measurement

Figure 11 shows an embodiment of the proposed tactile sensor for a fingertip use. Referring to the Fig.11, a cylindrical sensor body has an end plate 1 provided with a composite bearing 2 consisting of an axial slider 2a and a spherical bearing 2b at the center of the end plate 1. The axial slider 2a is supported in the spherical bearing 2b. In this embodiment, the sensitive shell has a hemispherical form and is unified with a support rod extending from the inner surface. The support rod is supported for axial movement and rotation in the composite bearing 2.

The sensitive shell is disposed at a position slightly spaced apart from the end plate 1 of the sensor body such that it is tiltable in all direction with respect to the sensor body and is

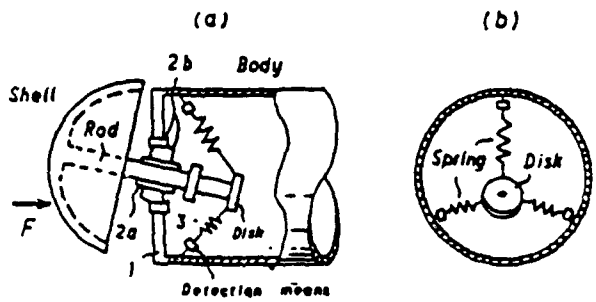


Fig.11. Displacement detection mechanism of a tip sensor.

also axially displaceable. A disk ... the free end of the support rod projecting from the bearing 2 into the sensor body. A plurality of radially uniformly spaced-apart extension spring are connected at one end to the disk and coupled at the other end to the inner periphery of the sensor body via respective detection means.

In the absense of any external force applied to the sensitive shell, a ring-like stopper 3 provided on an intermediate portion of the support rod comes into contact with the bearing 2, and the sensitive shell is biased such that it can balance at a reference position in a reference orientation. The detection means provided on the springs detect the deformation of deforming force of the springs. If a beam with a load cell or strain gauge is applied to the spring as the detection means, the extension force of the spring, i.e., deforming force, can be detected. Figure 11 shows an example of this arrangement. Output signals from the detection means will be supplied to an operational unit. And the force vector of the spring is calculated by using known parameters such as spring coefficient and the angles at which the springs are arranged.

The basis of the calculation is as follows: Since the detection means senses the displacement of springs, the end point of a spring can be expressed as a circular arc. When the support rod is rotated about its axis, however, the range is generally expressed as a spherical surface. Since the distance between the ends of the springs on the disk is known and also a straight line passing through the center of the disk at an angle of 90 degrees passes through the center of the composite bearing 2, the position and inclination of the disk are determined by geometric calculations. It should be noted that since the position of the disk is

passes through the center of the composite bearing 2, the position and inclination of the disk are determined by geometric calculations. It should be noted that since the position of the disk is expressed as a function of the deforming force, it is possible to estimate the torque operated on the rod, i.e., an external force acting on the hemispherical sensitive shell.

When the external force is supposed to be parallel to the axis of the finger body, the magnitude and point of action of an external force can be determined uniquely. When two or more external forces are simultaneously applied, the results of calculation represent values concerning the resultant force.

When the external force F is exerted on the sensitive shell in a state where the cylindrical sensor body and cylindrical sensitive shell are coaxial with each other as shown in Fig.2, the two cylindrical bodies become eccentric with respect to each other. More specifically, the elastic member (expressed by the springs) at positions closer to the point of action of the external force F are elongated, while the elastic members on the other side are contracted, whereby the sensitive shell is stabilized. When the point of action of the external force F shifts progressively to the right from the position shown in Fig.2(a), the extent of deformation of elastic members on the right side is progressively increased and eventually becomes greater than that of elastic members on the left side.

The connecting disk to which the end of each support rod is connected is unified with the sensitive shell. Also, the connecting disk may be provided with a pinhole at the center, i.e., at the position corresponding to the axis of the sensitive shell. A light-emitting element S and a position detection element R may be provided on the axis on

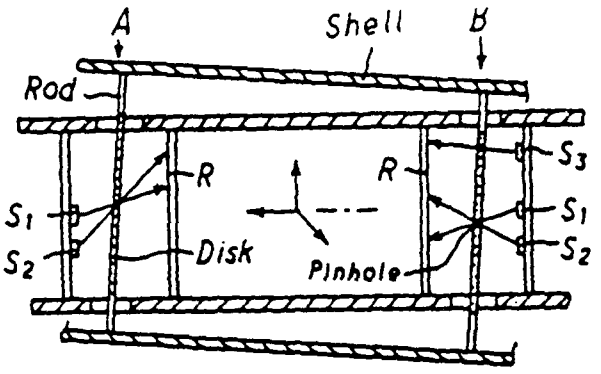


Fig.12. Displacement detection mechanism of a body sensor.

the opposite sides of the pinhole, whereby it is possible to readily detect a slight displacement of the sensitive shell with respect to the sensor body. And the data obtained can be processed to calculate the magnitude and the point of action of the external force F applied to the sensitive shell. In this case, there is no need of providing any detection element on the elastic members. A single light-emitting element S as shown in Fig.2 is sufficient only for the detection of the displacement of the sensitive shell in the radial direction.

5.2 Fine Measurement

When it is desired to obtain force detection in all directions, two light-emitting elements S_1 and S_2 may be mounted on a substrate unified with the sensor body as shown in Fig.12 from which the elastic members and detection processors have been omitted by way of structural simplification.

The positions of the light spot from the light-emitting elements are detected by a position detection element, so that it is possible to determine the position of the pinhole in the disk as an intersection of two light beam paths by a calculation unit which is not shown. However, when the detection element cannot simultaneously detect two or more light spot points as in the PSD (Position Sensitive Device), it is necessary to cause flickering of the light-emitting S_1 and S_2 one after another to determine the light spot positions separately. For this purpose, light emitting sources will be scanned so that only one light source is allowed to be active at the same time.

If the light-emitting element S_3 and the decentralized pinhole in the disk are provided on either side of the sensitive shell, the rotation angle of the sensitive shell about the axis of the sensor body can be calculated, by using the positional information of a light spot on the photosensor R, in addition to the information regarding inclination and axial deviation of the sensitive shell that has already been obtained. Obviously, the positional information can be obtained by using the same detection element. When the inclination, axial deviation and rotational displacement of the sensitive shell are determined, the deformations due to the elastic members are all calculated, so that it is possible to determine the

magnitude, direction and point of action of the external force on the sensitive shell in the strict sense, however the number of the points is limited to only one.

For example, when the elastic members at two sides of the sensitive shell have the same elastic modulus, the position of action of the force F is determined according to the radial displacements at the two sides. If the displacements are equal, the position of action is determined to be at the middle point of the sensitive shell, and the magnitude of the force is expressed as a sum of these deformations.

By the above-mentioned optical measurement, we can calculate shift vectors of two disk centers at the cross section A and B, say V_a ($c_{1a}x+c_{2a}y+c_{3a}z$) and V_b($c_{1b}x+c_{2b}y+c_{3b}z$), respectively. Rotation angle, say ρ, related to the suspension shell is also computed. We suppose that vectors V_a and V_b exist in a same plane since the suspension shell is rigid and the number of the external force is only one. That is, the plane including vectors V_a and V_b also includes the axis Z and is expressed by the plane μ in Fig.13. Thus, the angle β is determined. Actually, we have

$$\beta=\cos^{-1}[c_{1a}/(c_{1a}^2+c_{2a}^2)^{1/2}]. \tag{29}$$

The angle β is equally obtained by replacing parameters (c_{1a},c_{2a}) by (c_{1b},c_{2b}) in Eq.(29). These results are used to determine the force vector F and the position where the force F operates.

Let the three components of F be f_a,f_s,f_r as illustrated in Fig.13. The components f_s and f_r are shear forces at the contact point. By utilizing the result β, stiffness of the suspension at the cross section A and B are determined from Eq.(1), say $k_{\beta a}$, $k_{\beta b}$, for the stiffness when $\theta=\beta$ at sections A and B.

Accordingly, the normal force f_{na} and f_{nb} (see Fig.13) are

$$f_{na}=k_{\beta a}\sqrt{c_{1a}^2+c_{2a}^2}. \tag{30}$$

$$f_{nb}=k_{\beta b}\sqrt{c_{1b}^2+c_{2b}^2}. \tag{31}$$

Then, we obtain

$$f_n=f_{na}+f_{nb}. \tag{32}$$

Further, the unknown parameter expressing the position of the contact point is calculated to make a force balance such that

$$h=\frac{l_0(f_{na}-f_{nb})}{f_{na}+f_{nb}}. \tag{33}$$

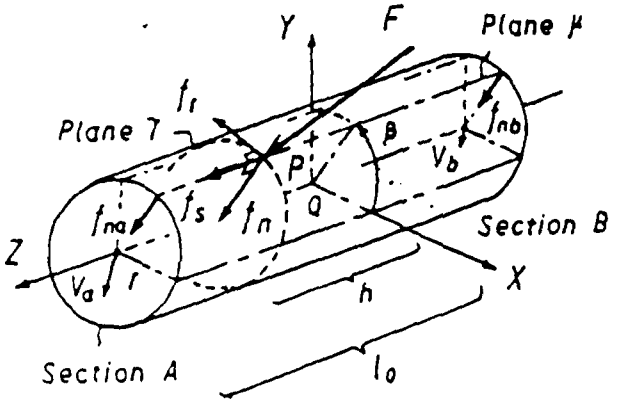


Fig.13. Force components of the external force F

We neglected lateral stiffness of the spring so far because of smallness in general. But to the shear forces f_r and f_s, the lateral stiffness k_r takes important role to make the sensor shell stable. It is not so easy to express the stiffness since the length of the spring affects the stiffness. However, in our application, the displacement of the spring is so small as compared with the spring length. Therefore, we use the value such that

$$k_r=k_\theta/J, \tag{34}$$

where J is the constant depending on the radius of a spring, length of a spring and so on [21]. In general, k_r is smaller than k_θ.

By using the stiffnesses k_{ra_i} and k_{rb_i} at the sections A and B, we can approximate the force components f_a and f_r such that

$$f_a=\sum_{i=1}^{3}k_{ra_i}c_{3a}+\sum_{i=1}^{3}k_{rb_i}c_{3b}. \tag{35}$$

$$f_r=\sum_{i=1}^{3}(k_{ra_i}+k_{rb_i})r_1\rho. \tag{36}$$

All of these calculations are valid when the suspension shell is rigid. The position parameter h will be accurate when the external force operates in the interval between the section A and B. Finally, the force F is written as

$$F=(f_n\cos\beta+f_r\sin\beta)x+(f_n\sin\beta+f_r\cos\beta)y+f_a z. \tag{37}$$

Contact point is expressed by the parameters β and h. As a result, we can determine the force and the point to which the external force operates.

5.3 Characteristics of the Sensor

Proposed sensor having the sensitive shell and the sensor body are unified such that there is no insensitive zone, and the effects of the sensors can be obtained to the utmost when the sensors are utilized in artificial fingers having articulations. When an object is handled with only the fingers, the object can be handled without dropping it only when the entire surface of the fingers is provided with the sense of touch. This sensor makes it possible to provide an artificial finger with a sophisticated sense like the sense of human skin. Further, by appropriately selecting elastic members to be used, the displacement versus force characteristics of the sensitive shell can be varied according to the rotation angle of the sensor body about the axis. For example, it is possible to increase the sensitivity of the sensor in a particular direction or produce a uniform sensitivity independent of the rotation angle.

The sensor body can accomodate optical or electric components inside, so that the operation can be performed without hindrance from signal lines. Also, the entire finger can be reduced in size. Further, information equivalent to that obtainable when an infinite number of sensors are provided is collected with a limited number of signal lines, thus it is possible to dispense with means for scanning sensors and reduce the weight of the overall sensor device. Even if an unexpected great force should be applied to the sensitive shell, the sensor body will safely receive this force. Therefore, the sensitive shell need not be made of a metal but may be made of a material lighter

so long as it has sufficient rigidity to maintain a shape similar to that of the sensor body. From this standpoint, the sensor body can be readily reduced in weight.

6. Concluding Remarks

Appropriate tactile sensors have been proposed for dexterous fingers. The sensors have abilities to detect external forces applied not only to the end surface of the sensor body but to the entire outer periphery thereof and is thus free from any insensitive zone. More specifically, all of electrical and mechanical elements for the detection are housed in the sensor body. It is thus possible to simplify the detection and facilitate the size reduction of the tactile sensor for smartness. We intend to attain an experiment to show how smart the sensor becomes.

When the sensors are applied to a robot, it is possible to make a hand or a finger sensitive with a function like the tactile sense of a human skin. It is thus possible to realize a smart sensor for multi-jointed fingers which has heretofore been difficult to obtain.

Acknowledgement

The author expresses his sincere gratitude to Dr. Y. Shirai, head of the Electrotechnical Laboratory where he belonged to previously and to Dr. E. Takano, Professor of Niigata University where he belongs to currently, for their assistance to carry this research.

References

[1] A.R.Johnston: Optical Proximity Sensors for Manipulators, JPL Technical Memo No.33-612 (1973).
[2] T. Kanade and T.M. Sommer: An optical proximity sensor for measuring surface position and orientation for robot manipulation, Proc. of 3rd RoViSeC, Cambridge, 301/308 (1983).
[3] T. Okada: Development of an optical distance sensor for robots, Int. J. of Robotics Res. 1-4, 3/14 (1982).
[4] W. Daniel Hillis: A high-resolution imaging touch sensor, Int. J. of Robotics Res. 1-2, 33/44 (1982).
[5] J. Rebman and K. A. Morris: A tactile sensor with electrooptical transduction, Proc. of 3rd RoViSeC, Cambridge, 341/347 (1983).
[6] P.M. Taylor et al.: Software and hardware aspects of a sensory gripping system, Proc. of 11th ISIR, Tokyo, 267/272 (1981).
[7] Rolls Royce Co Ltd.: Touch trigger probe, UK Patent No.1445977 (1973).
[8] M.H.E. Larcombe: Carbon fibre tactile sensors, Proc. of 1st Int. RoViSeC, UK., 273/276 (1981).
[9] R. Bardelli et al.: Piezo and pyroelectric polymers skin-like tactile sensors for robots and prostheses, Proc. of 13th ISIR, Chicago, 14-34/49 (1983).
[10] P. Dario et al.: Ferroelectric polymer tactile sensors with anthropomorphic features, Proc. of IEEE Int. Conf. on Robotics, Atlanta, 332/340 (1984).
[11] L.D. Harmon: Automated tactile sensing, Int. J. of Robotics Res. 1-2, 3/32 (1982).
[12] Paolo Dario et al.: An anthropomorphic robot finger for investigating artificial tactile perception, Int. J. of Robotics Res. 6-3, 25/48 (1987).
[13] K. Salisbury: Integrated language, sensing and control for a robot hand, Proc. of Int. Symp on Intelligent Automation, 54/61 (1985).
[14] S.S.M. Wang and P.M. Will: Sensors for computer controlled mechanical assembly, The Industrial Robot, 5-1, 9/18 (1978).
[15] J.J. Edmond: Multi-axis load cell with arcuate flexures, United States Patent No.4099409 (1978).
[16] G. Kinoshita et al.: A pattern classification by dynamic tactile sense information processing, Pattern Recognition 7, 243/251 (1975).
[17] M. Raibert and J.E. Tanner: Design and implementation of a VLSI tactile sensing computer, Robotics Res. 1-3, 3/18 (1982).
[18] R.Tomovic and Z. Stojiljkovic: Multifunctional Terminal device with adaptive grasping force, Automatica, 11, 567/571 (1975).
[19] R. Bajcsy: What can we learn from one finger experiments ?, Proc. of ISRR-83, New Hampshire, (1983).
[20] S.C. Jacobsen, et al.: The UTAH/MIT dexterous hand: Work in progress, Int. J. of Robotics Res., 3-4, 21/68 (1984).
[21] A.M. Wahl: Mechanical springs, 2nd edition, 70, McGraw-Hill (1966).

ANALYSIS OF GRASPING AND MANIPULATION BY MULTI-FINGERED ROBOT HANDS

TSUNEO YOSHIKAWA* and KIYOSHI NAGAI+

*Automation Research Laboratory, Kyoto University
Uji, Kyoto 611, Japan
+Faculty of Science and Engineering, Ritsumeikan University
Kyoto, Kyoto 603, Japan

ABSTRACT

Manipulating force and grasping force for skillful manipulation of objects by multi-fingered robot hands are discussed in this paper. Firstly a short discussion of grasping and manipulating forces for two-fingered hands is given. Then, for three-fingered hands, a new representation of the internal force among the fingers is given. Based on this representation, the grasping force is defined as an internal force which satisfies the static friction constraint. The concept of grasp mode is also introduced. The manipulating force is then defined as a fingertip force which satisfies the following three conditions: 1) It produces the specified resultant force. 2) It is not in the inverse direction of the grasping force. 3) It does not contain any grasping force component. An algorithm for decomposing a given fingertip force into manipulating and grasping forces is presented.

I. INTRODUCTION

Various multifingered hands for manipulating objects skillfully have been developed so far [1]-[5]. Analytical studies of grasping and manipulation by robot hands have also been done by many researchers. Hanafusa and Asada [2] proposed the prehension potential for obtaining a stable finger position of an arbitrarily shaped object. Salisbury et al. [4] have done a basic kinematic analysis of articulated hands using the concept of mobility and connectivity. They have also given an expression relating the total resultant force to fingertip forces and having an explicit parameterization of internal force . To determine a suitable grasping posture, Salisbury et al. [4] proposed

to use the condition number of the Jacobian matrix, while Yoshikawa [6] proposed to use the manipulability measure. Kobayashi [7] defined the norm of the handling force for an expected external force, and proposed to use it for obtaining a suitable grasping posture. Hanafusa et al. [8] defined the magnitude of grasping force as the square root of the area of a triangle formed by the internal forces of three fingers. Kerr and Roth [9] proposed to determine the optimal internal forces as those with the minimum norm under an approximated frictional constraints. Hanafusa et al. [5] have used the terms of "manipulating force" and "grasping force" with respect to the joint torques of the fingers. In spite of these studies, the authors still think that the forces involved in grasping and manipulation have not been fully understood. This understanding is important for designing hands and for developing grasping and manipulation algorithms.

In this paper, the manipulating force and the grasping force are difined explicitly for articulated multingered hands, and the relation between fingertip force and these two forces is made clear. Firstly the definition of manipulating and grasping forces for two-fingered hands is given such that they agree with our intuitive image of these forces. Then a new representation of the internal force is given for three-fingered hands. Based on this representation, the grasping force and the manipulating force are defined . Finally an algorithm for decomposing a given fingertip force into manipulating and grasping forces is presented.

This paper is a revised version of our previous paper[10].

II. TWO-FINGERED HAND

First we consider the case of two-fingered hands shown in Fig.1. Assume that the two fingertips make point contancts with friction with a grasped object. The i-th finger (i=1,2) applies force $f_i \in R^1$ (R^n is the n-dimensional Euclidian space) on the object along the x axis. Then the relation between fingertip force $[f_1, f_2]^T$ and the resultant force $t \in R^1$ exerted on the object is given by

$$t = f_1 + f_2 = A \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \tag{1}$$

$$A = [1, 1] \tag{2}$$

Therefore the fingertip force $[f_1, f_2]^T$ which satisfies (1) for a specified resultant force t is given by the general solution

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = A^*t + [E_2 - A^*A] \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \tag{3}$$

where A^* is the pseudoinverse [11] of A , $[y_1, y_2]^T \in R^2$ is an arbitrary constant vector, and $E_n \in R^{n\times n}$ ($R^{n\times n}$ is the set of all $n\times n$ real matrices) denotes the unity matrix. The first term of (3) represents the fingertip force with the minimum norm among those which produce the resultant force t. The second term for any $[y_1, y_2]^T$ represents the fingertip force which produces zero resultant force, that is, this fingertip force is an internal force. Substituting (1) into (3) yields

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = A^*A \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} + [E_2 - A^*A] \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \tag{4}$$

This equation represents a decomposition of any fingertip force into two components. The first term on the right hand side $A^*A[f_1, f_2]^T \triangleq [\tilde{f}_{m1}, \tilde{f}_{m2}]^T$ might be considered as the component for manipulation , and the second term $[E_2 - A^*A]\,[f_1, f_2]^T \triangleq [\tilde{f}_{g1}, \tilde{f}_{g2}]^T$ as the component for grasping. Then from (2) we have $A^* = A^T(AA^T)^{-1} = [1/2, 1/2]^T$, hence from (4)

$$\begin{bmatrix} \tilde{f}_{m1} \\ \tilde{f}_{m2} \end{bmatrix} = \begin{bmatrix} (f_1 + f_2)/2 \\ (f_1 + f_2)/2 \end{bmatrix} \tag{5}$$

$$\begin{bmatrix} \tilde{f}_{g1} \\ \tilde{f}_{g2} \end{bmatrix} = \begin{bmatrix} (f_1 - f_2)/2 \\ -(f_1 - f_2)/2 \end{bmatrix} \tag{6}$$

If we consider the case of $f_1 = f_0 > 0$ and $f_2 = 0$, (5) and (6) become

$$\begin{bmatrix} \tilde{f}_{m1} \\ \tilde{f}_{m2} \end{bmatrix} = \begin{bmatrix} f_0/2 \\ f_0/2 \end{bmatrix} \tag{7a}$$

$$\begin{bmatrix} \tilde{f}_{s1} \\ \tilde{f}_{s2} \end{bmatrix} = \begin{bmatrix} f_0/2 \\ -f_0/2 \end{bmatrix} \tag{7b}$$

Hence the force component for grasping $[\tilde{f}_{s1}, \tilde{f}_{s2}]^T$ is not zero. Fig.2(b) shows this result schematically. It is easily seen that, under this state of grasping, since $f_2 = 0$, the object will easily slip out of the two fingers by an application of any small external force which is not parallel with the x axis. According to our intuitive image, the grasping force for this case should be zero. Hence (4) is not suitable for the definition of grasping force. Based on this consideration, we now propose to define the grasping force and manipulating force by the following equations.

$$\begin{bmatrix} f_{m1} \\ f_{m2} \end{bmatrix} = \begin{bmatrix} \ell \\ -(1-\ell) \end{bmatrix} |f_1 + f_2| \tag{8}$$

$$\begin{bmatrix} f_{s1} \\ f_{s2} \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \min(|f_1|, |f_2|) \tag{9}$$

where ℓ is an auxiliary parameter which takes value 1 when $f_1 + f_2 \geq 0$ and value 0 when $f_1 + f_2 < 0$. According to these definitions, the fingertip force $[f_1, f_2]^T = [f_0, 0]^T$ is decomposed into

$$\begin{bmatrix} f_{m1} \\ f_{m2} \end{bmatrix} = \begin{bmatrix} f_0 \\ 0 \end{bmatrix} \tag{10a}$$

$$\begin{bmatrix} f_{s1} \\ f_{s2} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{10b}$$

This result shown in Fig.2(c) agrees with our intuitive image of the grasping forces. In the following section, these definitions will be extended to the three-fingered hands.

III. THREE-FINGERED HAND

3.1 Fingertip Force and Resultant Force

Basic relations of forces in manipulation of a rigid object by a

three-fingered hand will be given under the following assumptions.

1) Each fingertip makes a frictional point contact with the object.

2) These contact points on the object do not change their positions due to slip or relative rotation between the object and the fingertips.

3) The three contact points are not located on a straight line.

4) The mechanism of each finger is such that each fingertip can exert a force to the object in any direction.

Assumption 1) means that each finger can apply a force but not a moment to the object. Assumption 2) implies that the curvature of the fingertip is infinitly large. If not, relative rotation between the object and the fingertip during manipulation will cause some change of the contact position, resulting in a motion with a non-horonomic constraint which is difficult to analyze [9]. Assumption 3) and 4) are necessary for arbitrary manipulation of the object.

Fig. 3 shows a robot hand and an object under consideration. In the figure, Σ_0 is the object coordinate frame fixed to the object, C_i is the contact point of the i-th finger (i=1, 2, 3), $r_i \in R^3$ is the position vector of C_i from the origin of Σ_0, and $f_i \in R^3$ is the fingertip force applied to the object by the i-th finger.

The resultant force $f \in R^3$ and resultant moment $n \in R^3$ due to $\{f_i,\ i=1,2,3\}$ are given by

$$f = \sum_{i=1}^{3} f_i \tag{11}$$

$$n = \sum_{i=1}^{3} (r_i \times f_i) \tag{12}$$

From (11) and (12), the relation between the total fingertip force $F \triangleq [f_1^T, f_2^T, f_3^T]^T$ and the total resultant force $T \triangleq [f^T, n^T]^T$ is expressed as

$$T = AF \tag{13}$$

where

$$A \triangleq \begin{bmatrix} E_3 & E_3 & E_3 \\ R_1 & R_2 & R_3 \end{bmatrix} \in R^{6\times 9} \tag{14}$$

$$R_i \triangleq \begin{bmatrix} 0 & -r_{iz} & r_{iy} \\ r_{iz} & 0 & -r_{ix} \\ -r_{iy} & r_{ix} & 0 \end{bmatrix} \in R^{3\times 3} \tag{15}$$

$$r_i \triangleq [\, r_{ix}, r_{iy}, r_{iz} \,]^T \tag{16}$$

The manipulation of the object is done through an appropriate choice of the total resultant force T.

3.2 Internal Force

The general solution of (13) is given by

$$F = A^*T + (E_9 - A^*A)y \tag{17}$$

where $y \in R^9$ is an arbitrary constant vector. The second term $(E_9 - A^*A)y$ in (7) represents the internal force. However, this expression of the internal force is not convenient due to the following fact. From assumption 3) it can be easily shown that the rank of A is 6 and

$$\text{rank}(E_9 - A^*A) = 3 \tag{18}$$

Hence the essential arbitrariness of the internal force is three. Therefore determining the internal force by the vrctor y means determining a three-dimensional quantity by operating a nine-dimensional quantity.

In order to avoid this inconvenience, a new expression of the internal force will be given. Let

$$e_{ij} \triangleq (r_j - r_i)/\| r_j - r_i \|, \quad i,j=1,2,3, \; i \neq j \tag{19}$$

where $\| r \|$ denotes the Euclidian norm of vector r. Then e_{ij} is the unit vector directing from C_i to C_j on the grasp plane Q including the three contact points, and $e_{ij} = -e_{ji}$. Fig. 4 shows these unit vectors on the plane Q. The following proposition holds

Proposition 1: A total fingertip force F is an internal force if and only if there exist a vector $z = [z_{23}, z_{31}, z_{12}]^T \in R^3$ such that

$$F = Gz \tag{20}$$

where

$$G \triangleq \begin{bmatrix} 0 & e_{13} & e_{12} \\ e_{23} & 0 & e_{21} \\ e_{32} & e_{31} & 0 \end{bmatrix} \in R^{9 \times 3} \tag{21}$$

Proof: The set of all internal forces S_I is given by

$$S_I \triangleq \text{Range}(E_9, -A^*A) = \text{Null}(A) \tag{22}$$

where Range($\cdot$) and Null($\cdot$) are, respectively, the range space and the null space of a matrix. Therefore, it is sufficient to show that

$$S_I = \text{Range}(G) \tag{23}$$

From (14), (15), (19) and (21), we obtain

$$AG = 0 \tag{24}$$

Hence

$$\text{Range}(G) \subset \text{Null}(A) = S_I \tag{25}$$

On the other hand,

$$\text{rank}(G) = 3 \tag{26}$$

From (18), (25) and (26), we finally obtain (23), completing the proof. □

The internal forces $\hat{f} \triangleq [\hat{f}_1^T, \hat{f}_2^T, \hat{f}_3^T]^T$ have the following geometric characteristics as is shown in Fig. 5. The load lines of the three fingertip forces $\hat{f}_i$ intersect at a point P on the plane Q and the three forces form a closed triangle (force triangle) due to the balance of force and moment. The three degrees of freedom of the internal force can be interpreted as the sum of the two degrees of freedom in the position of the point P on the plane Q, and one degree of freedom in the scale of the force triangle. Therefore, there is

one-to-one relation between choosing an internal force and determining both the position of P and the scale of the force triangle. The point P will be called the focus of the internal force hereafter.

IV. GRASPING FORCE

By experience we know that, when we handle an object by our fingers, not only the manipulating force but also the grasping force is necessary for not dropping the object. In this section, the grasping force is defined and the concept of grasp mode is introduced.

Definition 1: A fingertip force F is called a grasping force if the following two conditions are satisfied.

1) $AF = 0$ (27)

2) $$\frac{f_i^T a_i}{\| f_i \|} > \frac{1}{\sqrt{1+\mu_i^2}}, \quad i=1,2,3 \tag{28}$$

where a_i is the inward unit normal vector of the object surface and μ_i is the static friction coefficient at the contact point C_i.

Condition 1) implies that F is an internal force. Condition 2) means that each fingertip force satisfies the friction constraint. Due to this constraint, the location of point P on the plane Q in Fig .5 for a grasping force F is also restricted. This restriction will now be analyzed.

Let the position vector of focus P be r_p, and

$$e_{p\,i} \triangleq \frac{r_p - r_i}{\| r_p - r_i \|}, \quad i=1,2,3 \tag{29}$$

The vector $e_{p\,i}$ is the unit vector directing from contact point C_i to focus P. In the case of Fig. 5, the fingertip force $\hat{f}_i$ and the $e_{p\,i}$ are in the same direction. However, this does not necessarily hold for all cases. For example, in the case of Fig. 6, $\hat{f}_2$ and $e_{p\,2}$, and $\hat{f}_3$ and $e_{p\,3}$ are in the opposite direction. Taking this point into consideration, the constraint (28) can be shown to be equivalent to the following two relations:

$$f_i = \mathrm{sgn}(e_{p\,i}{}^T a_i)\,\| f_i \|\, e_{p\,i} \tag{30}$$

$$|\, e_{p\,i}{}^T a_i \,| > \frac{1}{\sqrt{1+\mu_i{}^2}}, \quad i=1,2,3 \tag{31}$$

In order to express the state of the internal force described by (20), let

$$\alpha \triangleq [\, \alpha_1, \alpha_2, \alpha_3 \,] \tag{32}$$
$$\alpha_i \triangleq \mathrm{sgn}(z_{(i+1)(i+2)}), \quad i=1,2,3$$

where the subscript i is interpreted as (i-3) when $i \geq 4$ for notational convenience, and sgn(a) denotes the sign of a, i.e.,

$$\mathrm{sgn}(a) = \begin{cases} +1, & \text{if } a>0 \\ \pm 1, & \text{if } a=0 \\ -1, & \text{if } a<0 \end{cases} \tag{33}$$

Since $\mathrm{sgn}(z_{i(i+1)})$ implies whether the grasping force between fingers i and i+1 is compression or tension, we can categorize the internal forces by α. A fingertip force which satisfies (20) will be called the internal force of mode α. For example, the force in Fig. 5 is of mode [+1, +1, +1]. When F is a grasping force, since this mode is useful for classifying it, α will be called grasp mode. Fig. 7 shows the four essentially different grasp modes. Let β be

$$\beta \triangleq [\mathrm{sgn}(e_{p\,1}{}^T a_1),\ \mathrm{sgn}(e_{p\,2}{}^T a_2),\ \mathrm{sgn}(e_{p\,3}{}^T a_2)] \tag{34}$$

This parameter β represents the relation between the location of focus P and the shape of the object. Define the regions I through VII on the plane Q and their code $\gamma = [\gamma_1, \gamma_2, \gamma_3]$. ($\gamma_i$ =+1 or -1) as shown in Fig. 8. Then we can easily show that for an internal force to be a grasping force, one of the relationships among α, β, and the regions I -VII listed in Table 1 should hold. These relationships can further be condensed to the following relation.

$$\beta = \gamma \text{ or } -\gamma \tag{35}$$
$$\alpha = (\gamma_1 \cdot \gamma_2 \cdot \gamma_3)\beta \tag{36}$$

Variables α, β and γ will be denoted by only signs + and - hereafter. For example, [+1, +1, -1] will be expressed as [+ + -].

Note that (35) is a condition on the relation between the location of P and the shape of the object with respect to P, and (36) tells the grasp mode realizable under the given location of P and shape of object. Relation (36) is schematically shown in Fig. 9. Summarizing the above argument, we have,

Proposition 2: Suppose that the grasping positions r_i, the inward normal directions a_i, and the static friction coefficients μ_i are given. Then a grasping force can exist if there is a focus P satisfying the following two conditions.

1) a_1, $e_{p\,i}$, and μ_i satisfy the constraint (31).
2) β and γ satisfy condition (35).

Using the expression (20) for the internal force and the mode α (i.e., the signs of z_{ij}), an expression of the grasping force $F_g \triangleq [f_{g1}^T, f_{g2}^T, f_{g3}^T]^T \in R^9$ is given by

$$F_g = B_g h_g \tag{37}$$

where

$$B_g \triangleq \begin{bmatrix} 0 & \hat{e}_{13} & \hat{e}_{12} \\ \hat{e}_{23} & 0 & \hat{e}_{21} \\ \hat{e}_{32} & \hat{e}_{31} & 0 \end{bmatrix} \tag{38}$$

$$\hat{e}_{i(i+1)} \triangleq \alpha_{i+2} \cdot e_{i(i+1)},\ i=1,2,3 \tag{39a}$$

$$\hat{e}_{i(i+2)} \triangleq -\hat{e}_{(i+2)i},\ i=1,2,3 \tag{39b}$$

$$h_g \triangleq [h_{g1}, h_{g2}, h_{g3}]^T,\ h_{gi} \geq 0 \tag{40}$$

Note that in (39) the first subscript i of $\hat{e}_{ij}$ implies the finger i. Vectors $\hat{e}_{ij}$ play the role of unit vectors for the grasping force.

Although there is a close relation between the shape of object and the mode α of realizable grasping force, there are cases where two or more modes are possible for a given shape of object. Fig. 10 shows the realizable modes of grasping force for a cylindrical object. The plane Q is assumed to be parpendicular to the axis of the object, so that the vectors a_i are in the plane Q. It is also assumed that $\mu_i = 0.4$. Fig. 10(a) and (b) are the cases where there is only

one mode $\alpha = [\ +\ +\ +\ ]$ and $\alpha = [\ -\ +\ +\ ]$, respectively, while Fig. 10(c) is the case where two modes $\alpha = [\ +\ +\ +\ ]$ and $\alpha = [\ -\ +\ +\ ]$ are realizable. When there are more than one realizable mode, some additional consideration for selecting one mode will be necessary. This would be a topic for furher study.

V. MANIPULATING FORCE

Manipulating force is defined and its relation to the grasping force introduced in the previous section is discussed.

Definition 2: For a given resultant force T, the fingertip force $F=[f_1^T, f_2^T, f_3^T]^T$ is called the manipulating force of mode α if F satisfies the following three conditions.

1) $T = AF$ (41)

2) $f_i^T \hat{e}_{ij} \geq 0, \quad i=1,2,3, \quad j=(i+1),(i+2)$ (42)

3) $(f_i^T \hat{e}_{i(i+1)})(f_{i+1}^T \hat{e}_{(i+1)i}) = 0, \quad i=1,2,3$ (43)

Note that $\hat{e}_{ij}$'s are defined by (39), and so are dependent on mode α. Condition 1) implies that F produces the resultant force T. Condition 2) means that the manipulating force is not in the inverse direction of the grasping force. Condition 3) implies that the grasping force components between C_1 and C_2, C_2 and C_3, and C_3 and C_1 are zero.

In order to give an explicit expression for the manipulating force, we first introduce the following matrix.

$$B_m \triangleq \begin{bmatrix} \ell\,\bar{e}_{12} & 0 & (1-n)\,\bar{e}_{13} & \bar{e}_{10} & 0 & 0 \\ (1-\ell)\,\bar{e}_{21} & m\,\bar{e}_{23} & 0 & 0 & \bar{e}_{20} & 0 \\ 0 & (1-m)\,\bar{e}_{32} & n\,\bar{e}_{31} & 0 & 0 & \bar{e}_{30} \end{bmatrix} \tag{44}$$

where

$$\bar{e}_{i0} = \frac{\hat{e}_{i(i+1)} \times \hat{e}_{i(i+2)}}{\| \hat{e}_{i(i+1)} \times \hat{e}_{i(i+2)} \|} \tag{45a}$$

$$\bar{e}_{i(i+1)} = \hat{e}_{i(i+2)} \times \bar{e}_{i0} \tag{45b}$$

$$\bar{e}_{i(i+2)} = \bar{e}_{i0} \times \hat{e}_{i(i+1)} \tag{45c}$$

and ℓ, m, and n are parameters which can take 1 or 0. Vectors $\bar{e}_{ij}$ given by (45) play the role of unit vectors for manipulating force.

Fig. 11 shows the vectors $\hat{e}_{ij}$ and $\bar{e}_{ij}$ schematically for a typical case. Note that $\bar{e}_{i0}$'s are normal to the grasp plane Q and

$$\bar{e}_{ij}^T \hat{e}_{ij} \geq 0, \quad i=1,2,3, \quad j=(i+1),(i+2) \tag{46}$$

The matrix B_α is a function of α and (ℓ, m, n). The parameter ℓ (m, n) determines which of the two vectors $\langle \bar{e}_{12}, \bar{e}_{21} \rangle$ ($\langle \bar{e}_{23}, \bar{e}_{32} \rangle$, $\langle \bar{e}_{31}, \bar{e}_{13} \rangle$) is included in the matrix B_α. With these preparations, we can give the following proposition.

Proposition 3: A fingertip force F which produces the resultant force T, is a manipulating force of mode α if it satisfies

$$F = B_m h_m \tag{47}$$

for some α, (ℓ, m, n), and $h_m \triangleq [h_{m1}, h_{m2}, \cdots, h_{m8}]^T$ with $h_{mi} \geq 0$, i=1,2,3.

Proof: Suppose a fingertip force $F \triangleq [f_1^T, f_2^T, f_3^T]^T$ satisfies (47), i.e.,

$$\begin{aligned} f_1 &\triangleq \ell h_{m1} \bar{e}_{12} + (1-n) h_{m3} \bar{e}_{13} + h_{m4} \bar{e}_{10} \\ f_2 &\triangleq m h_{m2} \bar{e}_{23} + (1-\ell) h_{m1} \bar{e}_{21} + h_{m5} \bar{e}_{20} \\ f_3 &\triangleq n h_{m3} \bar{e}_{31} + (1-m) h_{m2} \bar{e}_{32} + h_{m6} \bar{e}_{30} \end{aligned} \tag{48}$$

It is then enough to show that these f_i satisfy (42) and (43). From (45) we have

$$\hat{e}_{i(i+1)}^T \bar{e}_{i(i+2)} = \hat{e}_{i(i+1)}^T \bar{e}_{i0} = 0 \tag{49a}$$

$$\hat{e}_{i(i+2)}^T \bar{e}_{i(i+1)} = \hat{e}_{i(i+2)}^T \bar{e}_{i0} = 0. \tag{49b}$$

Therefore, from (48), (49) and (46)

$$f_1^T \hat{e}_{12} = \ell h_{m1} \bar{e}_{12}^T \hat{e}_{12} \geq 0 \tag{50a}$$

$$f_1^T \hat{e}_{13} = (1-n) h_{m3} \bar{e}_{13}^T \hat{e}_{13} \geq 0 \tag{50b}$$

and

$$\begin{aligned}&(f_1{}^T\hat{e}_{12})(f_2{}^T\hat{e}_{21})\\&= \ell(1-\ell)h_{m1}{}^2(\bar{e}_{12}{}^T\hat{e}_{12})(\bar{e}_{21}{}^T\hat{e}_{21})\\&= 0\end{aligned} \tag{51}$$

Hence (42) and (43) for i=1 are satisfied. It can also be shown by similar arguments that (42) and (43) for i=2 and 3 are satisfied by F. □

The manipulating force will be expressed by $F_m \triangleq [f_{m1}{}^T, f_{m2}{}^T, f_{m3}{}^T]^T$ hereafter.

Proposition 4: *For any given manipulating force F_m of mode α,*

$$\| F_m \| \leq \| F_m + F_g \| \tag{52}$$

for any grasping force F_g of the same mode α.

Proof. From (37) and (42)

$$F_m{}^T F_g = \sum_{i=1}^{3} (f_{mi}{}^T f_{gi}) \geq 0 \tag{53}$$

Therefore,

$$\begin{aligned}\| F_m + F_g \|^2 &= \| F_m \|^2 + 2F_m{}^T F_g + \| F_g \|^2\\&\geq \| F_m \|^2\end{aligned} \tag{54}$$ □

A manipulating force F_m of mode α which produces the resultant force T, satisfies

$$T = AF_m = AB_m h_m \tag{55}$$

Hence, for obtaining an F_m for a given T, we first calculate $h_m \triangleq [h_{m1}\ h_{m2}\ \cdots\ h_{m8}]^T \in R^8$ from

$$h_m = (AB_m)^{-1}T \tag{56}$$

for eight triples (ℓ , m, n) (ℓ , m, n = 1 or 0). We then select the one for which $\hat{h}_{m\,i} \geq 0$ (i=1, 2, 3) as the vector h_m. Then $F_m = B_m h_m$ with the above selected values h_m and (ℓ , m, n) is the desired manipulating force. The triple (ℓ , m, n) which satisfies $\hat{h}_{m\,i} \geq 0$ (i=1, 2, 3) is uniquely determined except for degenerate cases where at least one of the three $\hat{h}_{m\,i}$ is zero.

VI. DECOMPOSITION OF FINGERTIP FORCE INTO MANIPULATING AND GRASPING FORCES

A method for decomposing a given fingertip force into a manipulating force and a grasping force will be given. Suppose that r_i , a_i , μ_i (i=1, 2, 3), and a fingertip force F are given, and that this F can be expressed as a sum of a manipulating force F_m and a grasping force F_g for some mode α. Then

$$F = F_g + F_m = BH \tag{57}$$

where

$$B \triangleq [\, B_g , B_m \,] \tag{58}$$
$$H \triangleq [\, h_g{}^T , h_m{}^T \,]^T \tag{59}$$

An algorithm for decomposition of F is as follows.

1) Find the set of all rearizable modes α from r_i , a_i , μ_i (i=1,2,3).

2) Pick up one rearizable mode α from the set obtained in 1). Calculate $\hat{H} \triangleq [\, \hat{h}_g{}^T \; \hat{h}_m{}^T \,]^T$ by

$$\hat{H} = B^{-1}F \tag{60}$$

for the eight triples (ℓ , m, n) (ℓ , m, n=1 or 0).

3) Select the triple (ℓ , m, n) for which $\hat{h}_{g\,i} \geq 0$ (i=1,2,3) and $\hat{h}_{m\,i} \geq 0$ (i=1,2,3). Let $\hat{H}$ for this triple be H. Then $\langle\, F_m = B_m h_m , F_g = B_g h_g \,\rangle$ is a decomposition of F.

4) Repeat the steps 2) and 3) until all the rearizable modes are checked.

Note that the decomposition may not be unique. Also there are cases

where such decomposition does not exist.

A simple example of decomposition will be given in the following. The object is a cylinder of radius 5 with its axis coinciding with the z axis of the object coordinate frame $\sum_o$. Suppose that the three contact points and the fingertip forces are given by

$$\begin{aligned} r_1 &= [\ 0,\ -5,\ 0\ ]^T, & f_1 &= [\ 1,\ 8,\ 0\ ]^T \\ r_2 &= [\ 5\sqrt{3}/2,\ 5/2,\ 0\ ]^T, & f_2 &= [\ -3,\ -2,\ 0\ ]^T \\ r_3 &= [\ -5\sqrt{3}/2,\ 5/2,\ 0\ ]^T, & f_3 &= [\ 6,\ -2,\ 0\ ]^T \end{aligned} \tag{61}$$

Therefore the plane Q is given by the x-y plane of $\sum_o$, and the a_i's are

$$\begin{aligned} a_1 &= [\ 0 \quad 1 \quad 0\ ]^T \\ a_2 &= [\ -\sqrt{3}/2 \quad -1/2 \quad 0\ ]^T \\ a_3 &= [\ \sqrt{3}/2 \quad -1/2 \quad 0\ ]^T \end{aligned} \tag{62}$$

Also suppose that $\mu_i = 0.4$, i=1, 2, 3. Then using the above algorithm we obtain the following manipulating and grasping forces of mode [+ + +].

$$\begin{aligned} f_{m1} &= [\ 7/4,\ 4-3\sqrt{3}/4,\ 0\ ]^T, & f_{g1} &= [\ -3/4,\ 4+3\sqrt{3}/4,\ 0\ ]^T \\ f_{m2} &= [\ 0,\ 0,\ 0\ ]^T, & f_{g2} &= [\ -3,\ -2,\ 0\ ]^T \\ f_{m3} &= [\ 9/4,\ 3\sqrt{3}/4,\ 0\ ]^T, & f_{g3} &= [\ 15/4,\ -2-3\sqrt{3}/4,\ 0\ ]^T \end{aligned} \tag{63}$$

This is shown in Fig. 12. It can be easily seen that these forces satisfy Definitions 1 and 2.

Note that the expressions (47) and (37) of the manipulating and grasping forces for three-fingered hands correspond to the expressions (8) and (9) of those for two-fingered hands. In fact, (8) and (9) are obtained by the following steps.

(i) Similarly to (47) and (37), let

$$\begin{bmatrix} f_{m1} \\ f_{m2} \end{bmatrix} = \begin{bmatrix} \ell \\ -(1-\ell) \end{bmatrix} h_m, \quad h_m \in R^1 \tag{64}$$

$$\begin{bmatrix} f_{g1} \\ f_{g2} \end{bmatrix} = \begin{bmatrix} 1 \\ -1 \end{bmatrix} h_g, \quad h_g \in R^1 \tag{65}$$

(ii) Obtain ℓ, h_m and h_g such that $h_m \geq 0$, $h_g \geq 0$.

(iii) Substitute the above obtained h_m and h_g into (64) and (65).

Note also that (64) and (65) represent one grasp mode and there is another grasp mode which appears in the case of Fig. 13. For the latter grasp mode, we have

$$\begin{bmatrix} f_{m1} \\ f_{m2} \end{bmatrix} = \begin{bmatrix} -\ell \\ (1-\ell) \end{bmatrix} h_m, \quad h_m \in R^1 \tag{64'}$$

$$\begin{bmatrix} f_{g1} \\ f_{g2} \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix} h_g, \quad h_g \in R^1 \tag{65'}$$

instead of (64) and (65).

VII. CONCLUSION

The concepts of the manipulating force and grasping force, which are often used for describing some kinematic aspects of manipulation of object by human hand, have been studied for three-fingered robot hands. The main results obtained in this paper are summarized as follows.

1) A new representation of the internal force whose physical meaning is very clear, has been given.

2) The grasping force has been defined as an internal force which satisfies the static friction constraint. The concept of grasp mode

has been introduced.

3) The manipulating force has been defined as a fingertip force satisfying the following three conditions: 1) It produces the specified resultant force. 2) It is not in the inverse direction of the grasping force. 3) It does not contain any grasping force component. An explicit expression of the manipulating force has been given.

4) An algorithm for decomposing a given fingertip force into manipulating and grasping forces has been given.

We can easily extend these results to the case of four-fingered hands. These results are expected to be useful for developing control algorithms for multifingered robot hands and cooperated manipulation of objects by multiple robots. In particular, the concepts of grasping force and grasping mode would be helpful for secure grasp of objects with various shapes. As for the utilization of the results in this paper to the determination of fingertip forces for given manipulation tasks, see reference [12].

The Authors would like to thank Prof. H. Hanafusa, Ritsumeikan University, and Dr. Y. Nakamura, University of California, Santa Barbara, for their helpful discussions.

REFERENCES

[1] T. Okada, "Object-Handling System for Manual Industry", IEEE Trans. on Systems, Man, and Cybernetics, vol.SMC-9-2, pp.79-89, 1979.

[2] H. Hanafusa and H. Asada, "Stable Prehension by a Robot Hand with Elastic Fingers", Proc. 7th International Symposium on Industrial Robots, pp.361-368, Tokyo, 1977

[3] S.C. Jacobsen et al. "The Version I UTA/MIT Dextrous Hand", Robotics Research, The Second International Symposium, MIT Press, pp.301-308, 1985

[4] J.K. Salisbury and J.J. Craig, "Articulated Hands: Force Control and Kinematic Issues", International Jounal of Robotics Research, Vol. 1, No. 1, pp.4-17, 1982

[5] H. Kobayashi, "Control and Geometrical Consideration for an Articulated Robot Hand", International Jounal of Robotics Research, Vol. 4, No. 1, pp.3-12, 1985

[6] T. Yoshikawa, "Manipulability of Robotic Mechanisms",

International Jounal of Robotics Research, Vol. 4, No. 1, pp.3-9, 1985

[7] H. Kobayashi, "On the Articulated Hands", Robotics Research, The Second International Symposium MIT Press, pp.293-300, 1985

[8] H. Hanafusa, T. Yoshikawa, Y. Nakamura and K. Nagai, "Structural Analysis and Robust Prehension of Robotic Hand-Arm System, " Proc. '85 International Conference on Advanced Robotics, pp.311-318, Tokyo, 1985

[9] J. Kerr and B. Roth, "Analysis of Multifingered Hands", International Jounal of Robotics Research, Vol. 4, No. 4, pp.3-17, 1986

[10] T.Yoshikawa and K. Nagai, "Manipulating and Grasping Forces in Manipulation by Multi-Fingered Hands", Proc. 1987 IEEE International Conference on Robotics and Automation, pp .1998-2004, 1987

[11] T.L. Boullion and P.L. Odell, "Generalized Inverse Matrices", Wiley, New York, pp.1-11, 1971

[12] T.Yoshikawa and K. Nagai, "Evaluation and Determination of Grasping forces for Multi-Fingered Hands", to appear in the Proc. 1988 IEEE International Conference on Robotics and Automation, 1988

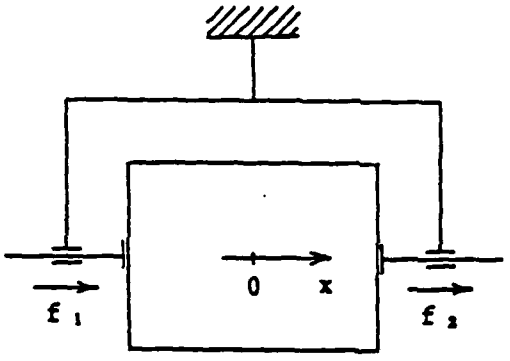
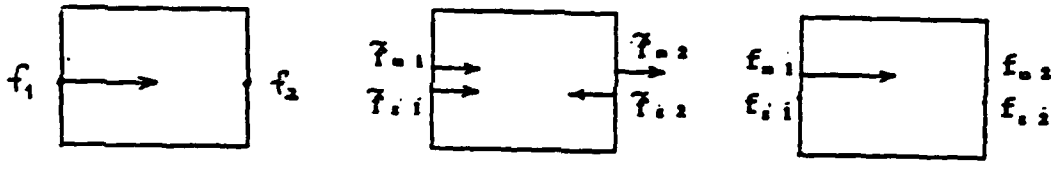


Figure 1 Two fingered hand



(a) Fingertip force (b) Decomposition by (5) and (6) (c) Decomposition by (8) and (9)

Figure 2 Decomposition of fingertip force

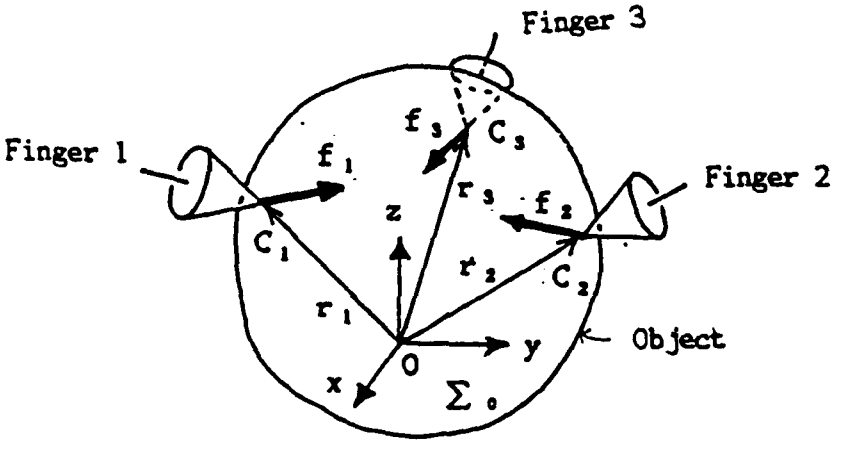


Figure 3 Three-fingered hand and object

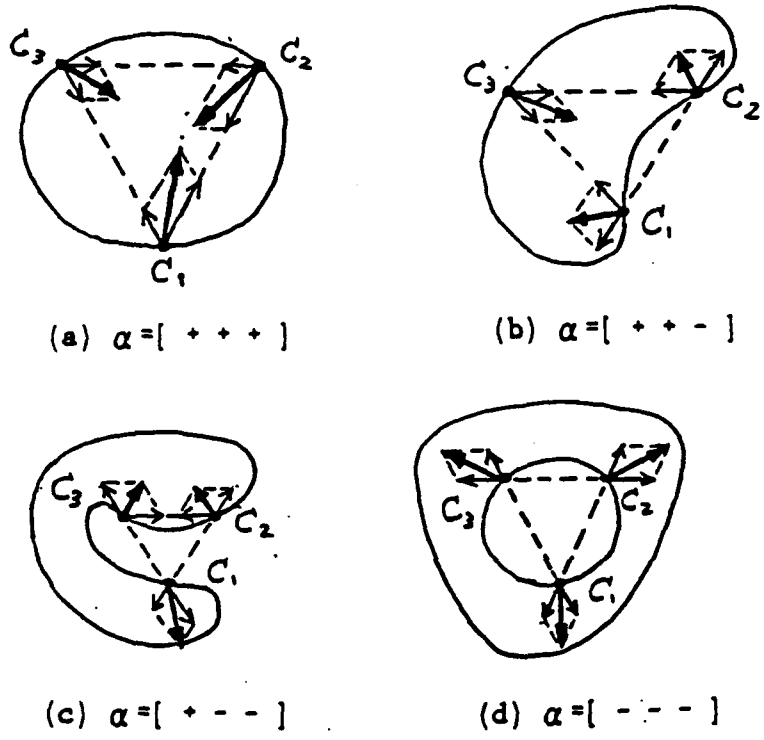


Figure 7 Four grasp modes

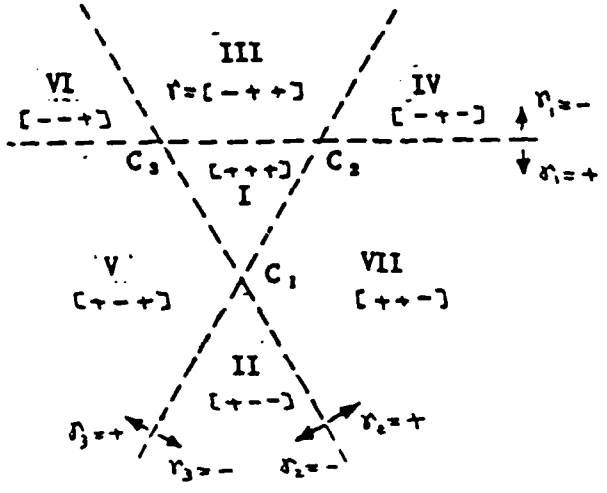


Figure 8 Seven regions on the plane Q

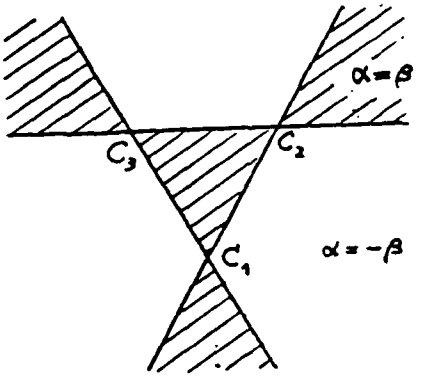
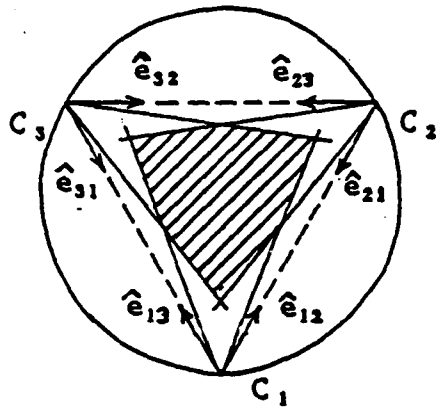
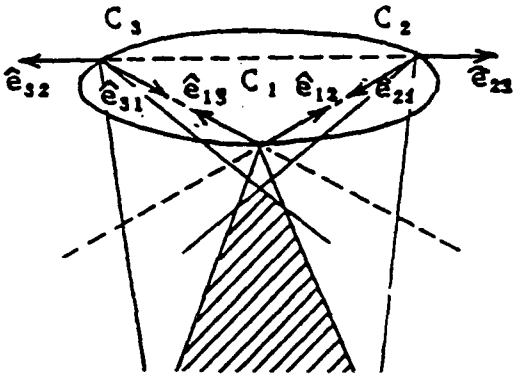


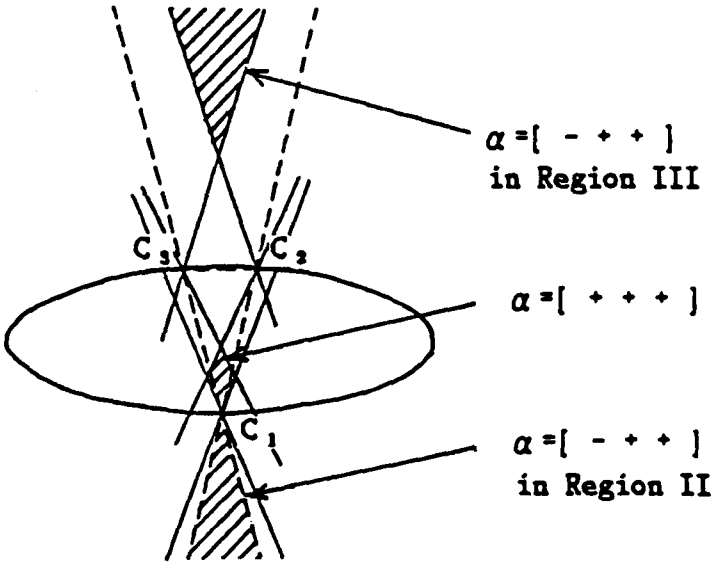
Figure 9 Relation (36)



(a) only α=[+ + +] is realizable.



(b) only α=[- + +] in Region II is realizable.



(c) α=[+ + +] and α=[- + +] in Region II and III are realizable.

Figure 10 Examples of realizable mode of grasping force

——— : Object
– – – – : Boundary of regions
——— : Frictional limit

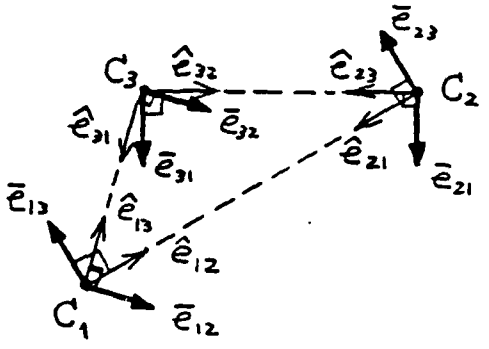


Figure 11 Relation between $\hat{e}_{ij}$ and $\bar{e}_{ij}$
($\bar{e}_{i0}$:Upward and normal to the Plane Q)

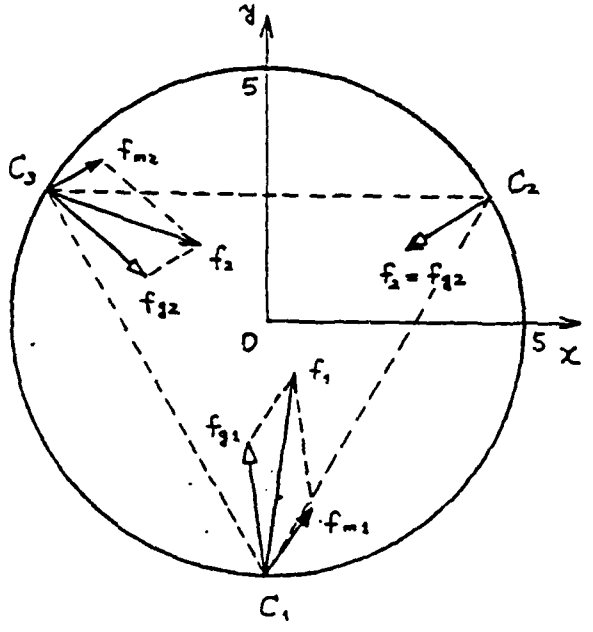


Figure 12 An example of decomposition of finger force into manipulating and grasping forces

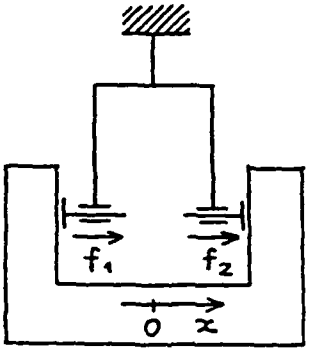


Figure 13 Two-fingered hand and concave object

Table 1 Relation among α, β and γ

Region	γ (code of region)	β (shape of object)	α (grasp mode)
I	[+ + +]	[+ + +]	[+ + +]
		[- - -]	[- - -]
II	[+ - -]	[+ - -]	[+ - -]
		[- + +]	[- + +]
III	[- + +]	[- + +]	[+ - -]
		[+ - -]	[- + +]
IV	[- + -]	[- + -]	[- + -]
		[+ - +]	[+ - +]
V	[+ - +]	[+ - +]	[- + -]
		[- + -]	[+ - +]
VI	[- - +]	[- - +]	[- - +]
		[+ + -]	[+ + -]
VII	[+ + -]	[+ + -]	[- - +]
		[- - +]	[+ + -]

5.5 .Control and Tactile Sensing for the Utah/MIT Hand

I.D. MCCAMMON AND S.C. JACOBSEN

ABSTRACT

The Center for Engineering Design at the University of Utah has been engaged in creating a dextrous arm and hand system for use in robotics and teleoperation. Initial efforts have produced experimental hardware such as the Utah/MIT Dextrous Hand (a research tool for exploring issues in dextrous manipulation), and a number of robot arms and hands for entertainment applications. These are practical, reliable systems that breed a unique demand for large amounts of tactile data. We believe that the problem of tactile data acquisition is not solved simply by fabricating arrays of closely spaced sensors, but is best approached by considering the sensing system in its totality. Sensors, preprocessors, multiplexers, data transmission, and control schemes must all be balanced to ensure the reliability and practicality of a particular design. This paper describes the development of a tactile sensing system, and presents the specific steps that are being taken to achieve the goal of a reliable, practical system. Beginning with a description of specific relationships between elements such as sensors, data channels, and data bandwidths, we show how system requirements favor particular system architectures, and how these architectures affect transducer design. Next, we describe a capacitive transducer, developed in collaboration with the Artificial Intelligence Laboratory at MIT, and present some preliminary results. Methods of reducing array density by using multiaxis sensors are then discussed, and preliminary experimental data from a six axis tactile sensor are presented. This device is a result of the ongoing microsensor work at CED, and permits the measurement of normal forces, shears, and torques about the point of contact.