SORCERER'S APPRENTICE: HEAD-MOUNTED DISPLAY AND WAND

by

Donald Lee Vickers

July 1974 UTEC-CSc-74-078

This research was supported in part by the University of Utah Computer Science Division and by the Advanced Research Projects Agency of the Department of Defense, monitored by Rome Air Development Center, Griffiss Air Force Base, New York 13440, under Contract F30602-70-C-0300.

TABLE OF CONTENTS

Page

ABSTRACT*

Sorcerer's Apprentice is an interactive computer graphics system utilizing a head-mounted display and a three-dimensional wand. The system allows three-dimensional interaction with line drawings which are displayed in real time, that is about 20 frames per second. The display, worn like a pair of eyeglasses, gives an illusion to the observer that he is surrounded by three-dimensional, computer-generated objects. The observer's view is continuously modified to compensate for his motion, allowing the objects to appear stationary as he walks among them. A hand-held wand lets the observer interact with the objects by "touching" them, moving them, changing their shapes, or joining them together. With the wand the observer can also create new objects and add to existing ones.

The major activity described here dealt with communication from the observer to the computer. Problems inherent to this communication such as sending commands to the computer in a simple yet expandable manner, sensing when the wand is "touching" an object, and real time processing of a changing data base have been solved. In the solutions we have used a wall chart, a hash addressed data structure, and a dual-copy data structure, respectively. Many minor problems still distract one while using the Sorcerer's Apprentice system, yet the ability to observe and modify three-dimensional objects in real time and in natural manner is very striking and very realistic.

*This report reproduces a dissertation of the same title submitted to the Department of Electrical Engineering, University of Utah, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

CHAPTER I
INTRODUCTION

The Problem

Ever since computers have been used to draw pictures of objects on a cathode-ray tube (CRT), computer scientists have been interested in finding better ways to display, define, and interact with these computer-generated objects, which will hereafter be referred to as "synthetic objects." Initially two-dimensional objects such as squares and circles were represented; soon after, perspective was added making them appear as cubes and spheres. Later, stereoscopic methods [1] of viewing were developed to give the objects a more truly three-dimensional (3-D) appearance. One of the most conceptually advanced 3-D displays [2] puts the observer in a 3-D environment wherein he seems to be surrounded by 3-D synthetic objects.

Advancement was also made in the area of interaction, but a basic problem still existed. The "state of the art" of interaction went from simple one- and two-dimensional devices such as knobs, buttons, joysticks, and teletypes to the more sophisticated light pens and 2-D tablets. The pens and tablets are also valuable for "drawing" or describing 2-D objects and have even been used for interacting with 2-D representations of 3-D objects [3]. Some systems provide the user with actual, though limited, 3-D control of 3-D objects with 2-D viewing [4] and even with 3-D viewing [5]. One problem with these systems offering 3-D control is that the volume of control does not coincide with the volume of observation, thus, utilization of a person's natural pointing ability is restricted.

Even with 2- and 3-D interaction and 3-D viewing capabilities available, some of the more reputable computer graphics institutions still work with 3-D synthetic objects using 1-D interaction and 2-D viewing. At the University of Utah, for example, the system almost exclusively used for defining and interacting with 3-D synthetic objects is a (1-D) teletype and a (2-D) CRT display. Certainly availability of equipment, existing programs and convenience tend to dictate which systems are used. But many two- and three-dimensional systems are simply not very useful because of inadequacies and ambiguities, for example, in trying to "draw" 3-D objects with 2-D devices, trying to visualize 3-D objects on 2-D displays, or trying to coordinate 3-D drawing and 3-D visualizing of objects on present systems.

The problem addressed herein is the development of a system which can be used to display, describe, and interact with 3-D synthetic objects in a realistic 3-D environment. The questions raised by this topic are: What methods of display and interaction should be used? Is it possible to use the same volume for interaction and display? What level of complexity should be used for communication from the observer to the computer? What data structure will best suit the system? How much information can be displayed still allowing real time motion without flicker? How much will the interaction device and data structure slow down the display rate?

Having built a system we need to ask: Is the system useful? What are its strong points and weak points? How easy is it to use? Is

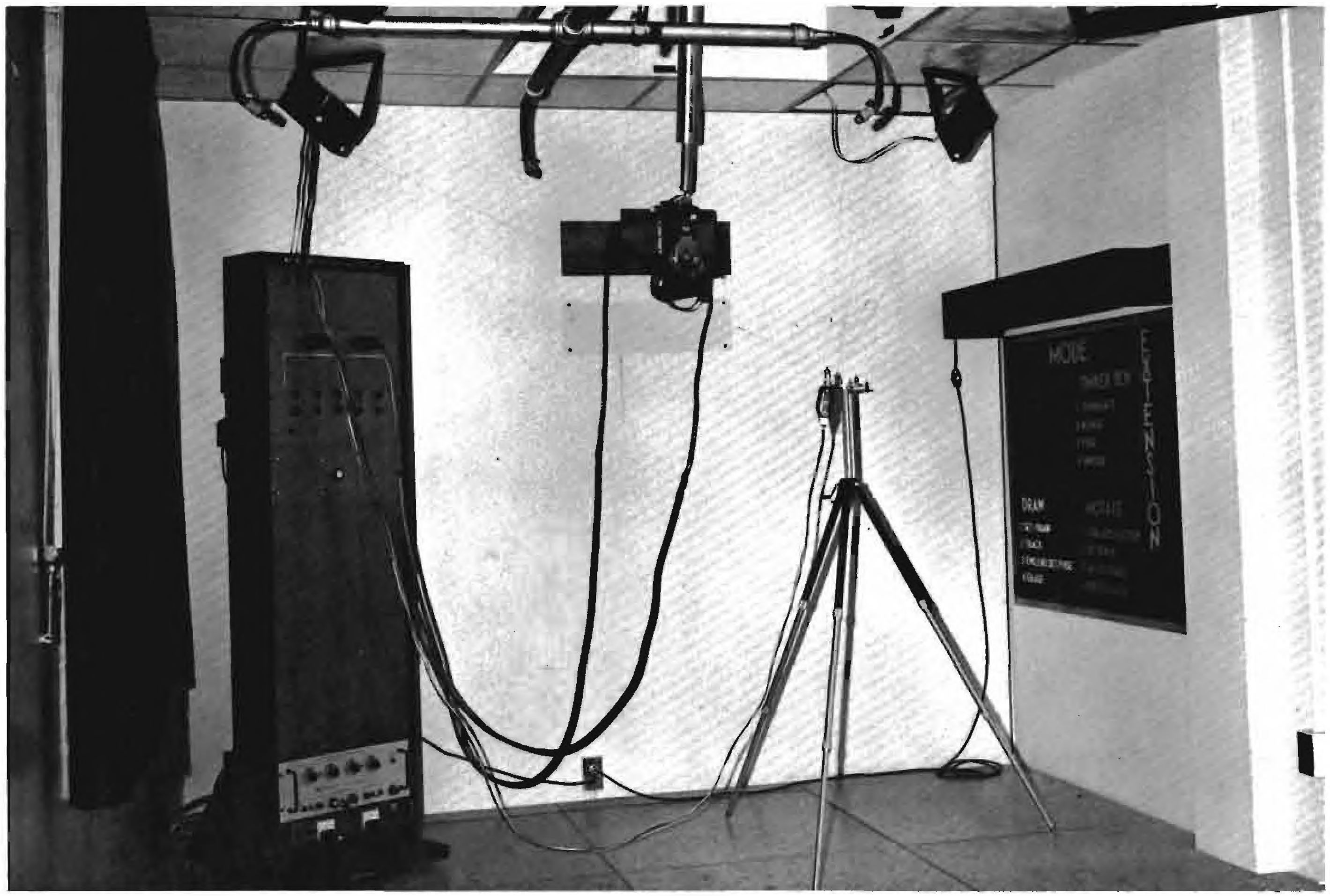


Fig. 1: Room containing Sorcerer's Apprentice apparatus including (from left) the analog circuitry rack with cables to the head-mounted display, the wand on its reference tripod, and a wall chart. A portion of the head position sensor can be seen extending upward from the head-mounted display. Near the ceiling are two of the three "puppet" boxes and the crossed pipes forming the ultrasonic "shower stall."

the illusion realistic? What are the contributions of the system to future systems of a similar nature?

The System

The Sorcerer's Apprentice system (Fig. 1) explored here is a combination of two systems, a head-mounted display [2] and a wand. The display portion of the system was chosen for its availability and novelty. A hand-held 3-D wand was designed for use in interaction. The major efforts of the author centered around this equipment and the programming for it.

For several years the head-mounted display (Fig. 2) has provided the University of Utah with a unique three-dimensional environment for looking at 3-D, computer-generated objects which exist only as data in a computer [6]. The illusion presented to one wearing this display is that he is surrounded by synthetic objects which he sees in addition to the features of his surroundings. These synthetic objects exhibit the size, perspective, and stability characteristics of real objects as the observer freely walks among them. Like real objects they are seen only when the observer is facing them; as he turns or walks past them, they leave his field of view.

A small wand (Fig. 3) with several buttons on it was built to enable an observer to interact with the synthetic objects by reaching out and "touching" them. It was thought that the ability to draw, change, and join 3-D objects in a 3-D environment using natural arm, body, and eye coordination would have many advantages over existing interactive systems. As an aid to reaching and "touching," the wand

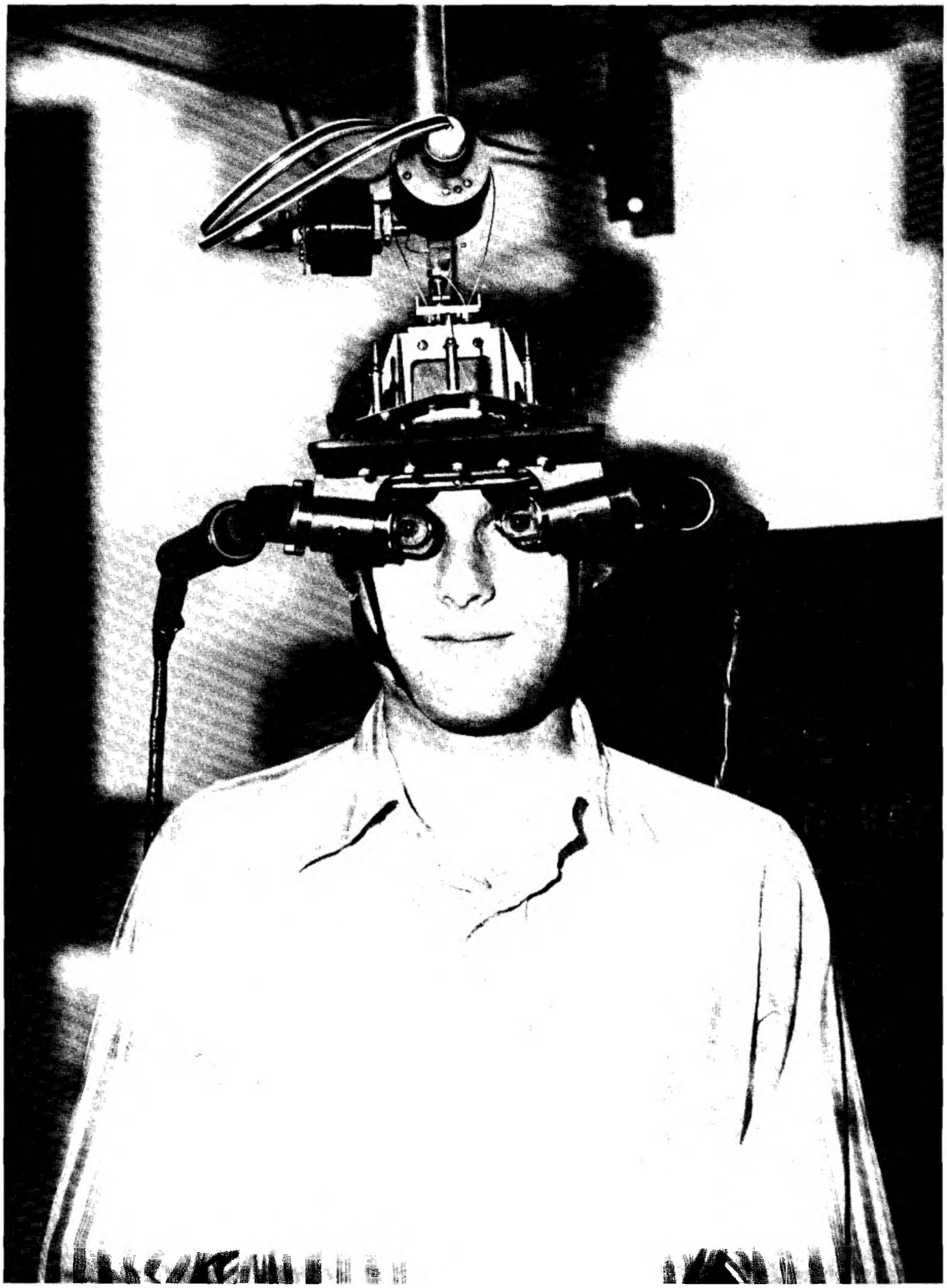


Fig. 2: Observer wearing the head set and harness of the head-mounted display.

as seen through the head-mounted display is marked by a spot of light, a cursor, which normally moves as though attached to the wand. Lines drawn using the wand appear as glowing wires and form "wire-frame" drawings. These lines do not fade in time but appear to be stationary in space. With the wand the lines can be joined to form objects which can then be moved, modified, and stored in magnetic tape from which they can be re-entered at a later date for further modification.

The system is fairly simple to use. Available commands are few and uncomplicated. Some can use it well the first time, but most require several sessions on the system to pass the "tolerance limit"--that point when the system becomes useful instead of merely tolerable.

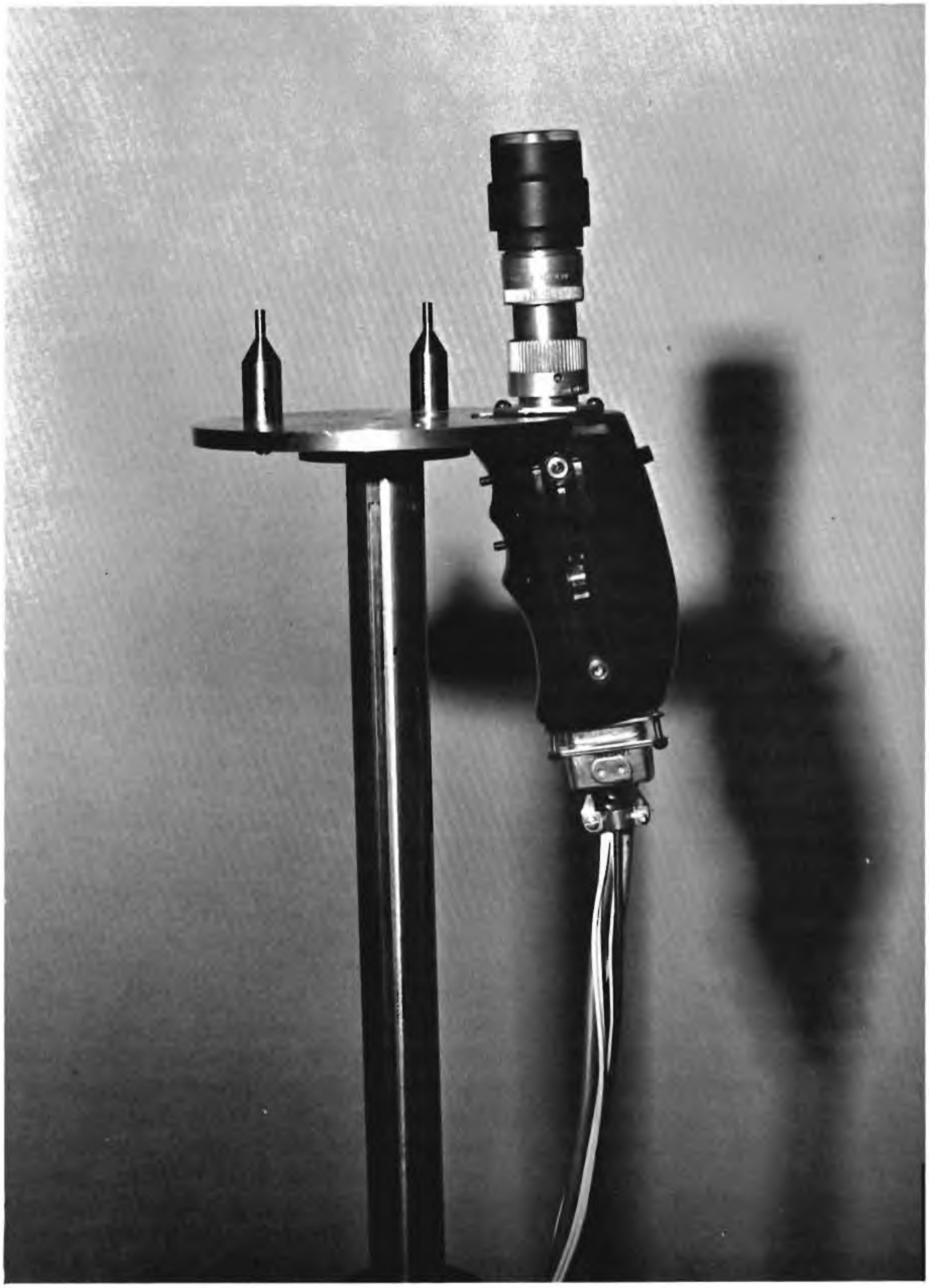


Fig. 3: Wand on its tripod "perch."
The ultrasonic tip is attached.

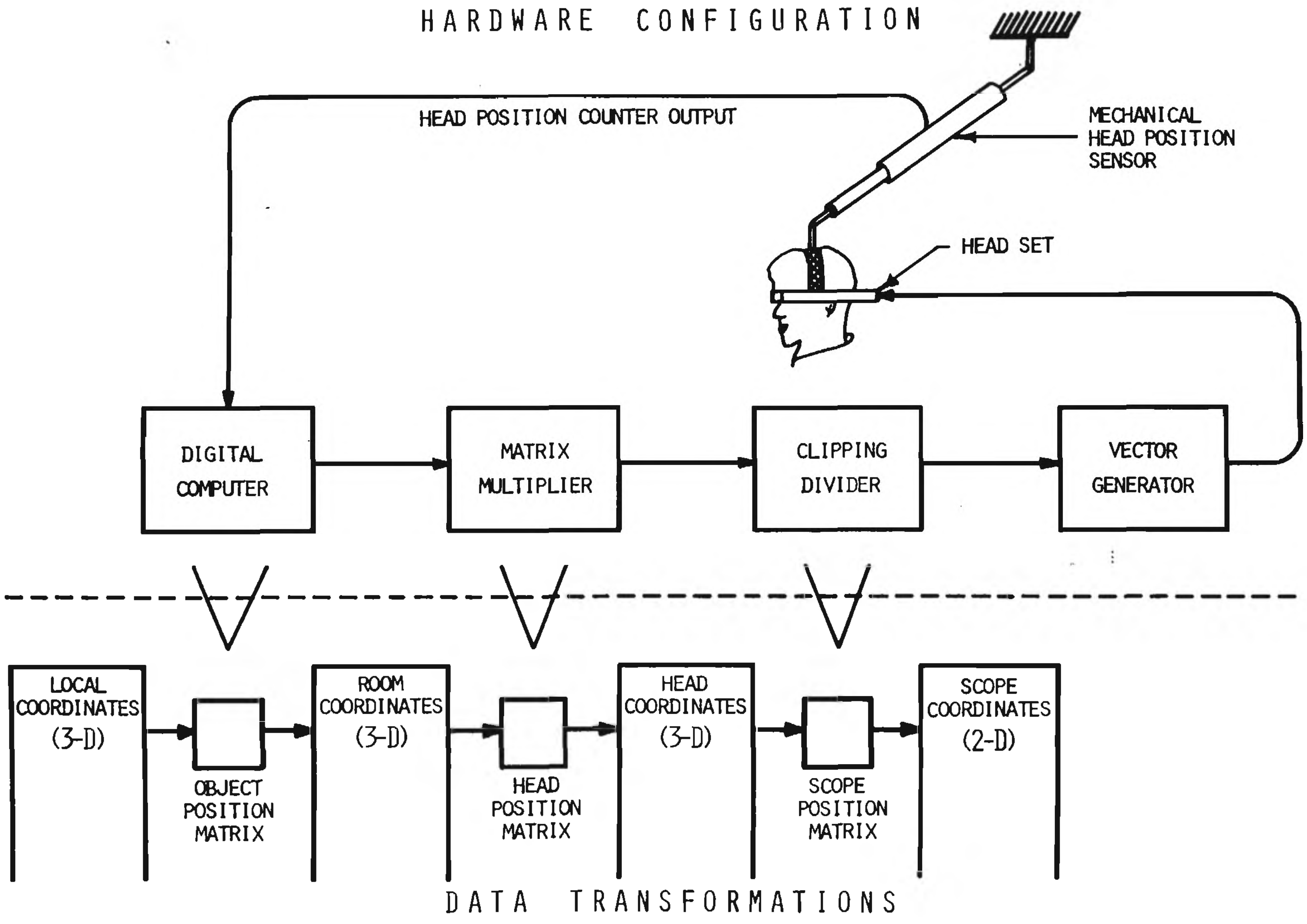


Fig. 4: Schematic showing flow of information and the corresponding data transformations in the head-mounted display system.

CHAPTER II
3-D ENVIRONMENT

The head-mounted display system which creates the illusory three-dimensional environment is composed of the head-mounted display and its supporting software. The special purpose display hardware and hand-optimized software produce a cyclic flow of information from the observer to the computer and back to the observer as seen in Fig. 4. Real time flow, 20 frames per second, of this information allows flicker-free viewing.

Synthetic objects seen through the head-mounted display appear to remain stationary as the observer moves. This stationarity results from continually changing the projection of the objects in a manner which compensates for head motion. Thus, the head-mounted display system is not only capable of displaying a frame every 20th of a second, but also of changing the perspective of each successive frame according to the changing position of the observer's head.

Head-Mounted Display Hardware

The head-mounted display was designed and built under the direction of Dr. Ivan E. Sutherland at Harvard College. It was completed and tested at Harvard in August 1968, just a few days before Dr. Sutherland brought it to the University of Utah. Though some portions of the hardware were immediately put into use at Utah, the display itself was not again used for viewing synthetic objects until the first day of 1970.

Initially the display hardware consisted of six separate subsystems: a head set, a head position sensor, a general purpose computer, a matrix multiplier, a clipping divider, and a vector generator. The multiplier, divider, and generator have been removed temporarily for modification.* Although their functions are now being simulated with software, they will nevertheless be discussed in this section.

Head Set.--The head set is a cathode-ray tube device which presents the synthetic objects to an observer. It is worn much like a pair of spectacles (Fig. 2). Each temple piece of the spectacles is a miniature CRT which is 6 inches long and has a 13/16 inch diameter screen. A 2-D projection drawn on the CRT face is reflected from an enclosed mirror into a series of lenses and finally to an eyepiece. Through the center of this clear glass eyepiece is a partially reflecting silver plane which directs the picture into the observer's eye. The observer thus sees a virtual image of the 2-D projection superimposed on his normal visual field. These synthetic objects appear to be in front of the observer and can be programmed to hang freely in space (Fig. 5,6) or to appear coincident with maps, walls, and other objects in the room.

The head set was designed to provide an observer with a stereoscopic view. With separate optics for each eye, stereo is possible by sending each eye a slightly different projection. Stereo pictures were

*The matrix multiplier became permanently inoperative after a system change in September 1970. In August 1971 the clipping divider and the vector generator interface were "taken down" for modification. Though progress in modification is slow, we still look with anticipation to the time when all the head-mounted display hardware will again be operational.

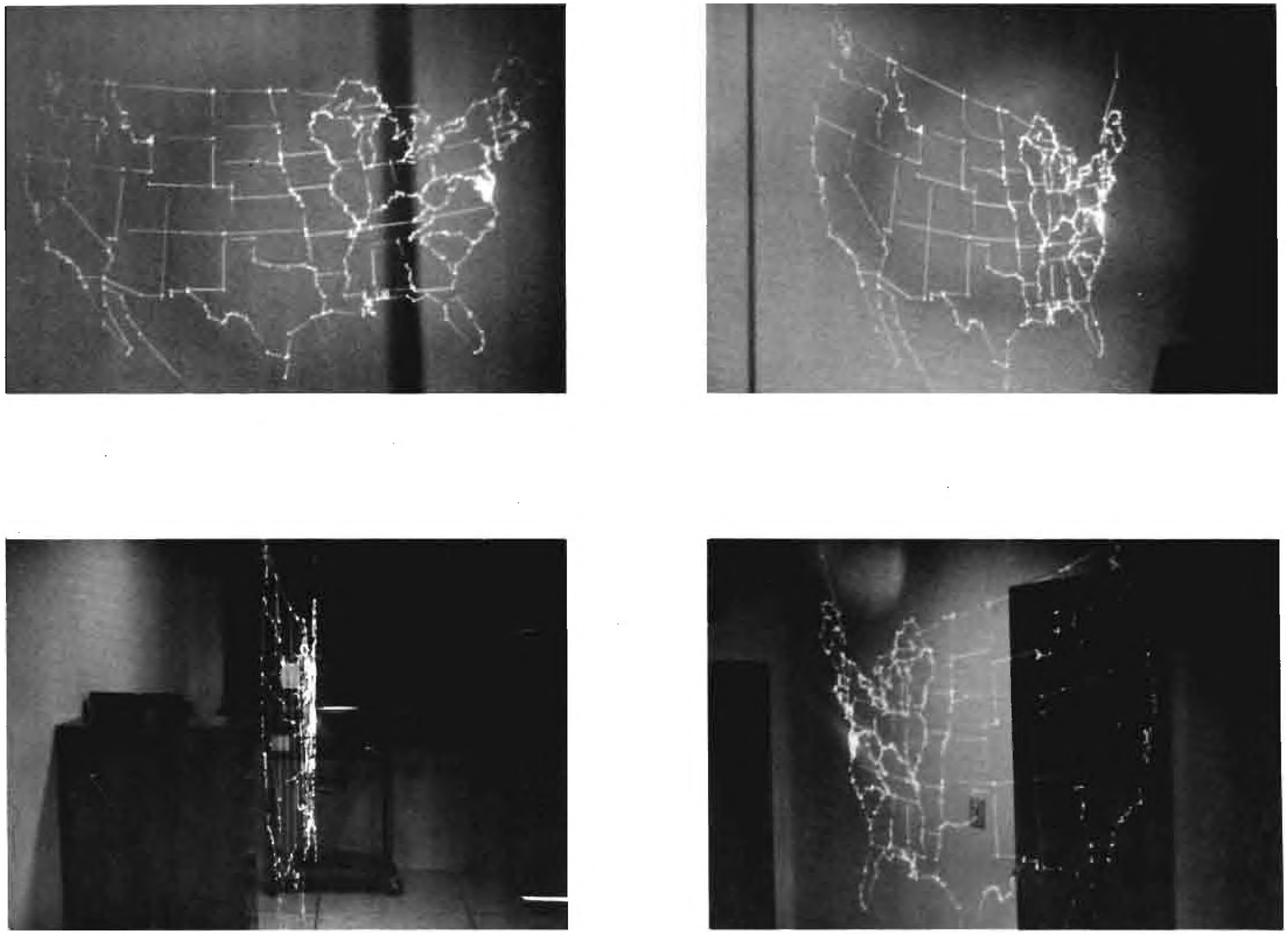


Fig. 5: USA map photographed from different angles through a head set eyepiece. The map consists of 649 line segments and was programmed to appear about three feet wide and two feet high.

seen at Harvard, but not dynamic stereo pictures. At Utah equipment limitations require the same view to be sent to both eyes.

Head Position Sensor.--The head position sensor is a simple mechanical device with the appearance of a two-section telescope (Fig. 7). The two telescoping tubes slide freely along their common axis and are fastened on one end through a universal joint to a swivel on the ceiling. The other end is connected through a second universal joint to the head set. Noncumulative-force springs support the shaft and head set so the observer is not burdened with their weight. A harness (Fig. 8) helps hold the head set in place on the observer's head.

Six rotary pulse generators monitor the six degrees of freedom of the observer's head. Five generators are mounted concentric to the universal joint axes and to the ceiling pivot axis; the sixth generator is used to indicate the extension of the telescoping tubes. The pulse generators supply 8000 pulses per revolution suggesting resolution in measurement of vertical translation of 0.00077 inch, horizontal translation of 0.048 to 0.067 inch depending on the shaft extension, and rotation of 2.7 minutes. Mechanical tolerances cause the overall accuracy of the head position sensor to be a great deal worse than the resolutions would indicate when considering the head's absolute position and orientation. However, when considering sustained motion in a given direction, the accuracy approaches the resolution.

The volume of head motion of the observer is limited by the head position sensor. An observer's head must always remain within the two foot high frustrum of a cone which is about eight feet high and six feet

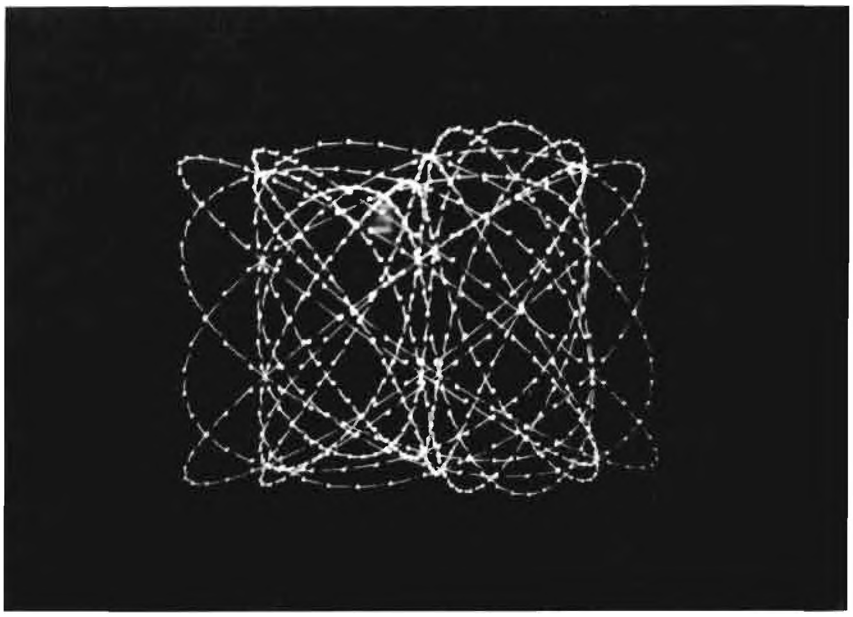


Fig. 6: A line segment 3:4:5 Lissajous' figure photographed through the head set. The "barbed wire" effect comes from a delay problem in the formerly used vector generator.

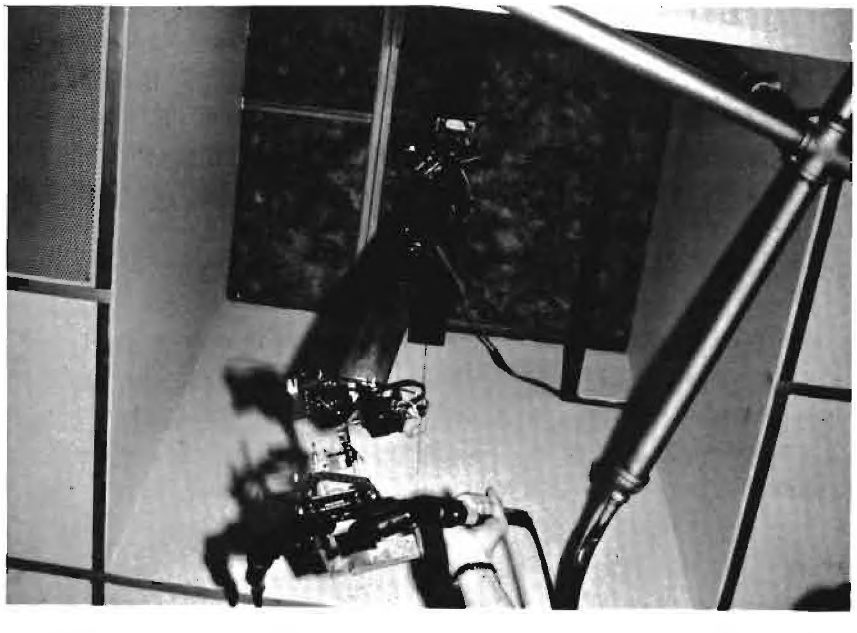


Fig. 7: Looking upward along the head position sensor shaft. The head set and a portion of the ultrasonic "shower stall" are also visible.

in diameter. Within this volume he can rotate up to two full turns in either direction from center. Normal neck motion is unrestricted with the exception that vertical tilt of the head is limited to a maximum of 30 degrees upward and 80 degrees downward.

General Purpose Computer.--A general purpose computer uses the output from the pulse generators on the head position sensor to create a head position matrix. Pulses from the six generators are continuously counted and stored in buffer registers. These registers are interrogated at the beginning of each frame by a Digital Equipment Corporation PDP-10 computer which creates from the six values a four-by-four homogeneous matrix [7,8]. This "head position matrix" (Fig. 4) defines the position and orientation of the observer's head with respect to a coordinate system attached to the head-mounted display room.* It is this matrix which appropriately changes the observer's view of synthetic objects as he moves about the room.

The PDP-10 is equipped with more than enough core to handle its other functions. Currently, in the "single user" mode the PDP-10 has 64K of one microsecond core memory. It is possible, though administratively difficult, to switch in an additional 96K of core from an adjoining time-sharing PDP-10. The other functions of the computer include interfacing with existing hardware and storing descriptions of the synthetic objects.

*The definition of head and room coordinate systems is relative. The values actually inserted into the head position matrix are those which describe where the room is in terms of the head coordinate system. It would be more correct to call this matrix the "inverse head position matrix" or the "room position matrix," but the title "head position matrix" will be used so as to remain in agreement with other publications [6,10].

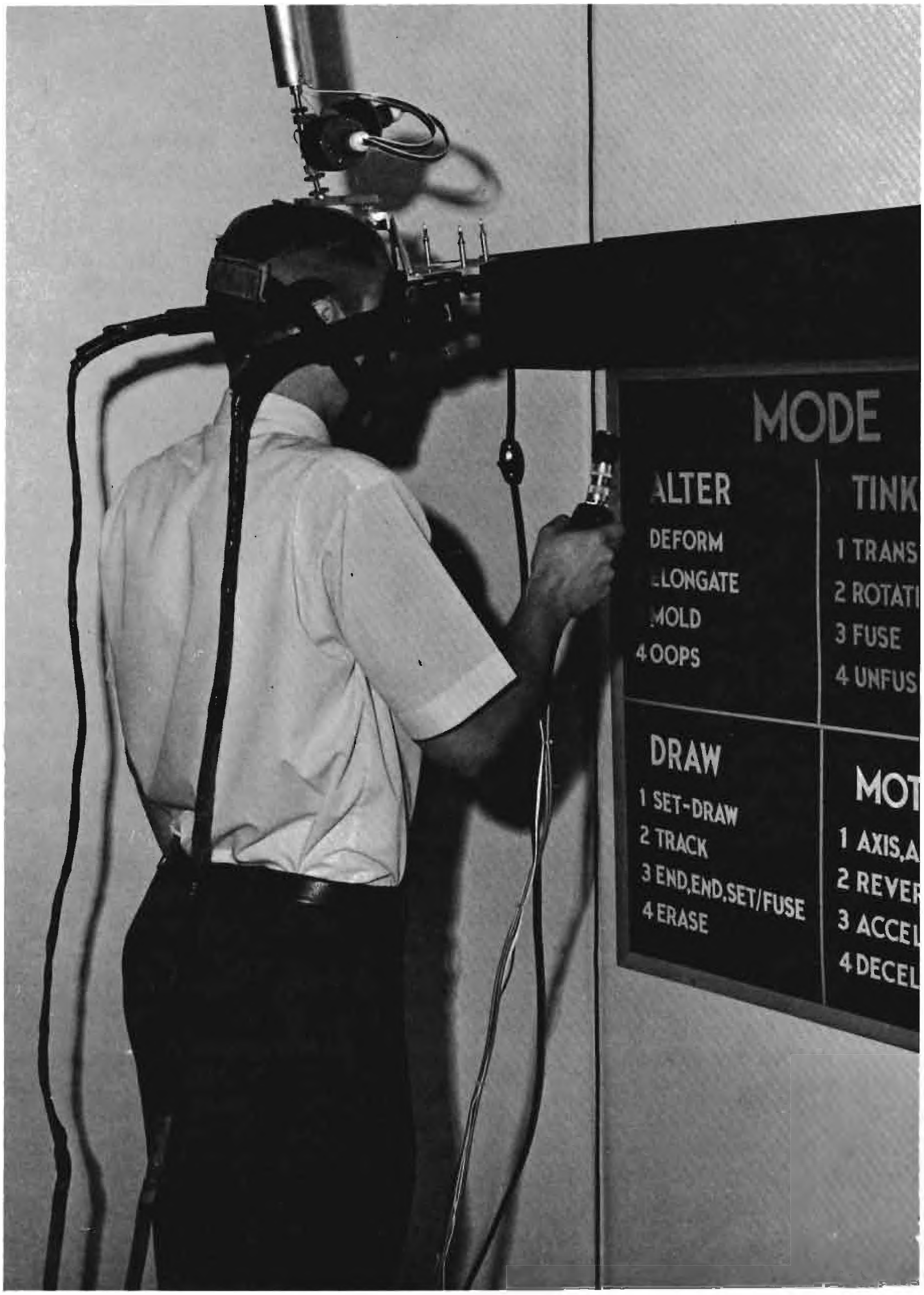


Fig. 8: Observer pointing with the wand at the wall chart.

Matrix Multiplier.--The matrix multiplier [2] is a high-speed device which transforms 3-D vectors from room to head coordinates (Fig. 4). It multiplies both endpoints of each vector by the head position matrix to perform the transformation. Before this multiplication, the head position matrix is multiplied on the right by a matrix which adjusts for the field of view of the head set optics. The hardware matrix multiplier was designed to multiply one endpoint by a four-by-four matrix in 5 microseconds. The software simulation takes about 200 microseconds.

Clipping Divider.--Head coordinate endpoints from the matrix multiplier are accepted in pairs by the clipping divider [9]. In the process of "clipping," the clipping divider eliminates those line segments and portions of line segments which are behind the observer or outside his field of view. The "dividing" function computes the perspective, as seen by the observer, of each visible line segment and by so doing transforms the 3-D object description into a 2-D description suitable for display on a flat screen.

Vector Generator.--The vector generator produces the analog signals necessary to draw lines on a CRT. These signals are proportional to the two-dimensional endpoint values the generator receives from the clipping divider. When all the head-mounted display hardware was in operation, the vector generator produced about 700 vectors without flicker (Fig. 5).

Currently a software vector generator simulator computes the coordinates of dots along visible line segments. These dots are displayed in rapid succession to simulate vectors (Fig. 12). Dot spacing may be specified by the user and is constant regardless of vector length.

Thus, a short vector will contain fewer dots than a long one. As an observer approaches a vector it appears to grow as dots are added to one end.

Even with all the software simulation and with the restriction of drawing only one dot at a time, the system is presently capable of drawing several screen-length vectors without flicker. The head set CRT phosphor persistence is such that a refresh rate of about 20 frames per second is necessary for synthetic objects to appear flicker-free. The number of flicker-free vectors is largely dependent upon the spacing of the dots which form the vectors. Twenty-four flicker-free vectors may be displayed with dot spacing of 100 scope units where 2048 is the screen length in scope units. Thirty-unit spacing of dots reduces the number of screen-length flicker-free vectors to about eight, but produces better looking vectors and is generally used (Fig. 12).

Head-Mounted Display Software

Head-mounted display software consists of a head position processor and a display file processor. The job of the head position processor is to compute the head position matrix from the information provided by the head position sensor. Of the 50 milliseconds allowed per frame for flicker-free viewing, only one millisecond is required by the head position processor to compute the head position matrix. Details of this computation are published [6].

The display file processor sees that the information in the display file is properly displayed. The display file contains a 3-D room coordinate description of the objects to be displayed. Figure 4 shows how the display file processor invokes the matrix multiplier and clipping divider to transform this description from room coordinates to head coordinates and finally to scope coordinates. For each frame the room coordinate description needs to be transformed only once by the head position matrix and only once by the scope position matrix. The addition of stereo would require one more transformation of the room coordinate description by the head position matrix. Similarly, multiple occurrences of a synthetic object on a CRT would require multiple transformations by the scope position matrix.

Most of the head-mounted display software was done in MACRO 10 assembly language. Two exceptions are the clipping divider simulator and the magnetic tape drive interface which were done in Fortran IV.

CHAPTER III
3-D INTERACTION

An observer within the 3-D environment of the head-mounted display system has at his disposal a wand system by which he can reach out and "touch" the synthetic objects he sees. Presently, interaction is possible only with specific vertices, usually line segment endpoints, of the synthetic objects. A wand for creating and interacting with synthetic objects visible only to one wearing the head set relates an aura of sorcery to bystanders and is the basis for the name of the combined head-mounted display and wand systems: Sorcerer's Apprentice.

The wand system contains a hand-held wand, an interpretive language for using the wand, and a unique data structure attached to the display file. All aspects of interaction with the wand are under control of the wand processor program.

Wand

Physically, the wand is an outgrowth of convenience and necessity (Fig. 3). The grip of the wand was taken from an inexpensive photographic flash attachment. It contains only four pushbuttons, a slide switch, and a small potentiometer for communicating with the computer. Four buttons were used simply because more would not conveniently fit. A cable carrying signals from the buttons, switch, and potentiometer and to the wand tips extends from the bottom of the wand.

Whereas both position and orientation of the head were monitored in the head-mounted display system, only the position of the wand is tracked. Thus, the wand tracking problem is really only a subset of the head tracking problem. For purposes of comparison* three methods of wand tracking were attempted. A ten-pin bayonet connector on top of the wand was designed to accept a tip from one of three tracking devices. The three tips (Fig. 9) include a continuous wave ultrasonic transducer, a mechanical or "puppet" adaptor, and a spark pen. Currently the ultrasonic and "puppet" tips, discussed below, are functional. The system for the spark pen, a modification of a commerically available pen [11], has been designed but only partly built.

The ultrasonic tracking device, a modification of a former head tracking system [12], uses an ultrasonic transmitter as the wand tip. The 37 KHZ sinusoidal output is picked up by four sensors mounted at the corners of a 45 inch square "shower stall" near the ceiling (Fig. 1). Phase shift of each of the four received signals relative to the transmitted signal is monitored by the computer. From these shifts and the initial length of the four vectors from the wand to the receivers, the wand processor program determines the wand's position with a typical resolution of 0.01 inch.

A mechanical tracking device uses the "puppet" tip which is attached to three 30 lb. test monofilament lines by a swivel. The other ends of these lines pass through teflon grommets, around encoder shafts to noncumulative-force springs. Three boxes (Fig. 10) containing the grommets, encoders, and springs are mounted on the ceiling in such a way that the small hole in each white grommet lies at the vertex of an 82.5 inch equilateral triangle. The encoders, which are rotary pulse

*See "Physical Restrictions," Chapter V.

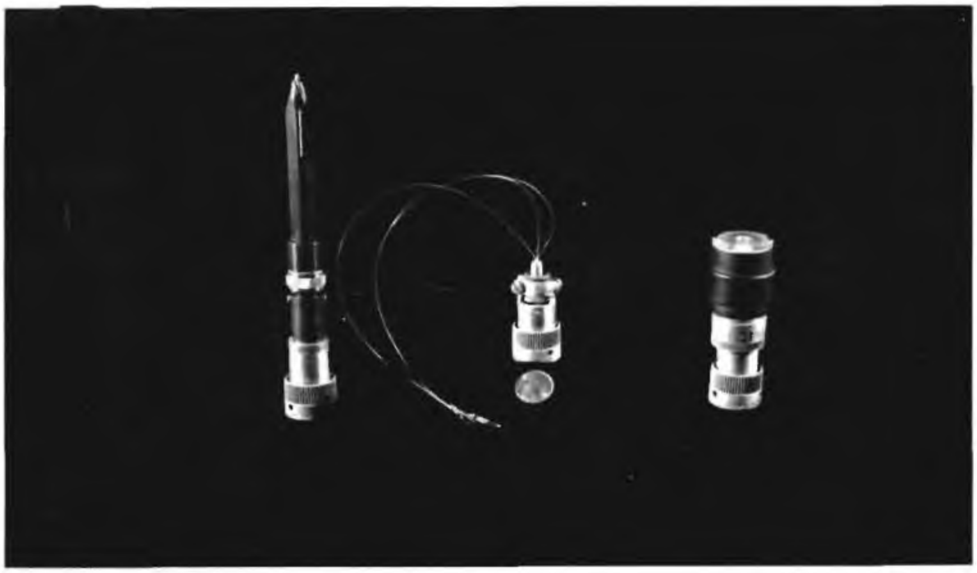


Fig. 9: Interchangable tips for tracking wand position. From left to right: sonic tip, "puppet" tip, and ultrasonic tip.

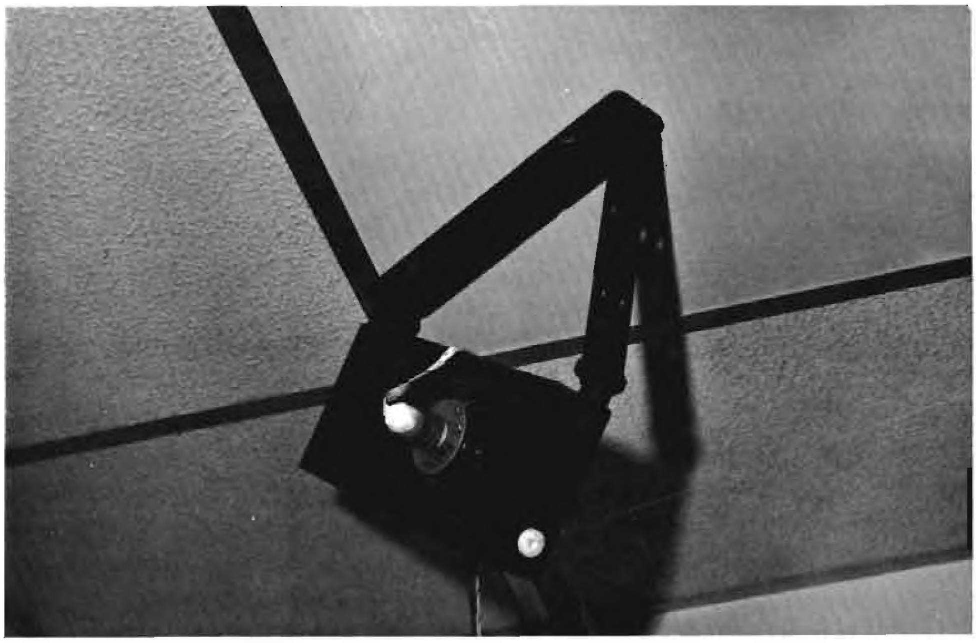


Fig. 10: One of three ceiling-mounted "puppet boxes." A monofilament line can be seen extending from a white teflon grommet. Protruding from one side of the box is a rotary pulse generator.

generators, produce 4000 pulses per 10.09 inches of line. If slipping and stretching of the lines could be prevented*, the accuracy of this method would fall in the range 0.0025 to 0.005 inch within the normal working volume available to an observer.

Wand Language

The wand language is built around a state table for which the current button positions serve as input [13]. Each state is linked to an action which is associated with the command specified by the person holding the wand. The organization gained by using a state table aids in changing the effect of a specific command or adding additional commands.

With only four buttons, the wand needs supplementation to provide enough commands for meaningful communication with the computer. Supplemental means of communication such as panel switches, portable keyboards, and touch tone keys seem unnecessarily complicated and burdensome. The limitation imposed by having only four buttons was finally alleviated by extending the interactive capabilities the wand has with synthetic objects to include inanimate objects, specifically, a wall chart.

The chart (Fig. 8,16) divides wand commands into four modes. Each mode, represented by a square on the chart, redefines the commands associated with the four buttons. These commands are printed on the chart for reference by the observer. An observer wearing the head set can see which mode is currently active by observing a synthetic cross

*See "Physical Restrictions," Chapter V.

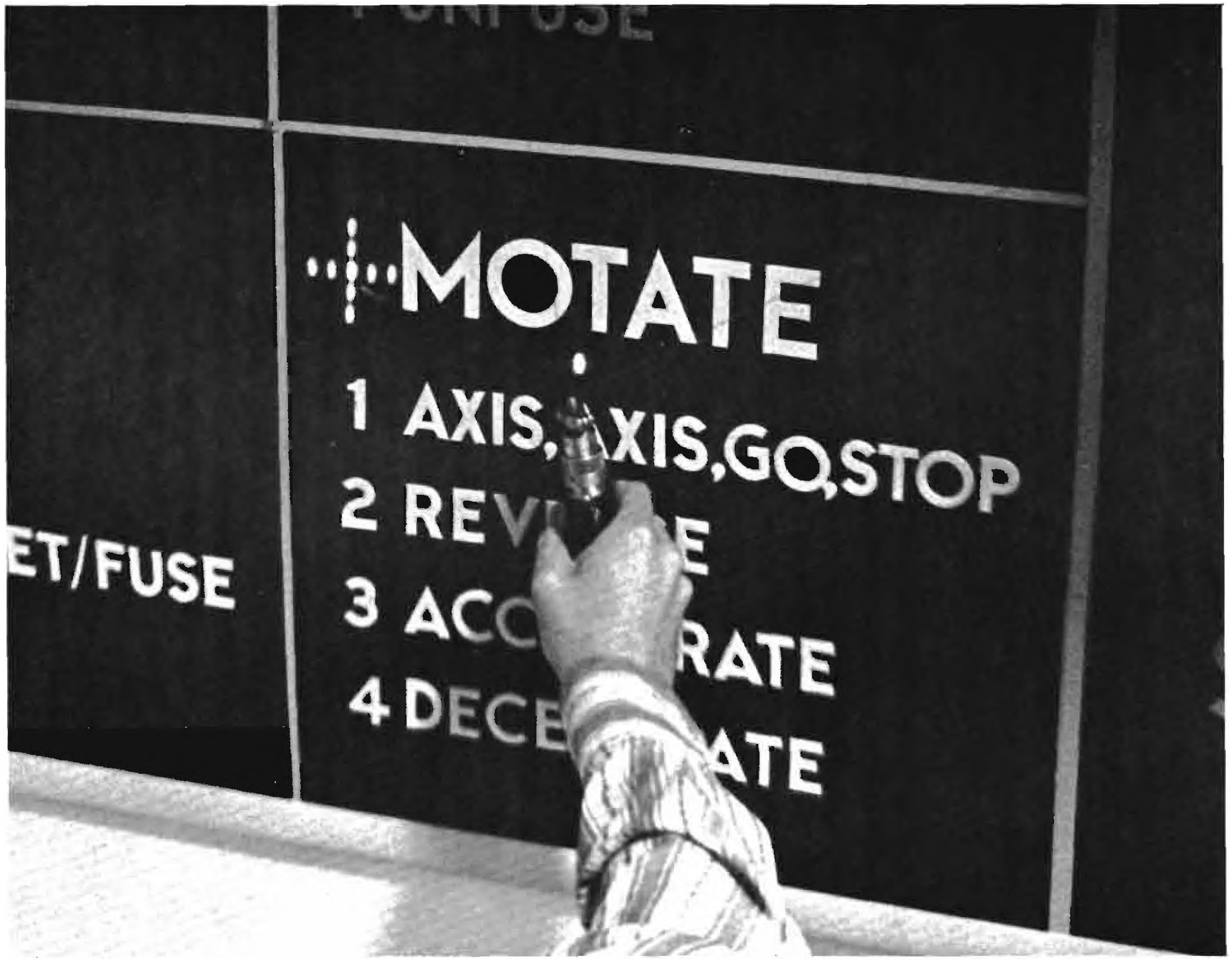


Fig. 11: Photograph of the wall chart taken through the head set. In addition to the observer's hand, the wand cursor and mode cross are visible.

in the appropriate mode square. In Figure 11, the MOTATE mode is active. In addition to button positions, the active mode serves as input to the wand language state table.

Interaction with the wall chart is the means by which an observer can change modes. He does this by simply pushing the slide switch up, "pointing" at the desired mode square, and pushing one of the buttons. "Pointing" is the act of aligning the wand cursor between the observer's right eye and the target (Fig. 8). The ability to interact with inanimate objects within the room is certainly extendible to more than four modes and certainly to more than just charts. Prior to such interaction one needs to know only the position and orientation of the head, the position of the cursor, and the position of the inanimate objects. One must also determine with which eye to align the wand.

Four modes currently appear on the chart. Our enthusiasm in defining the modes and commands exceeded the time available for implementation. Those modes and commands not presently operational appear in conjunction with the word "will" as though they are forthcoming, whereas, the certainty of their future implementation is unknown. Appendix A contains a more detailed explanation of the modes.

<u>ALTER</u> -- In this mode the observer is able to DEFORM synthetic objects by moving their vertices with the wand (Fig. 12). Eventually he will be able to ELONGATE or stretch objects and to MOLD them by moving one vertex with respect to an object's center and having all other vertices move proportionally with respect to the same center. In case a mistake is made, OOPS will recall a former image.

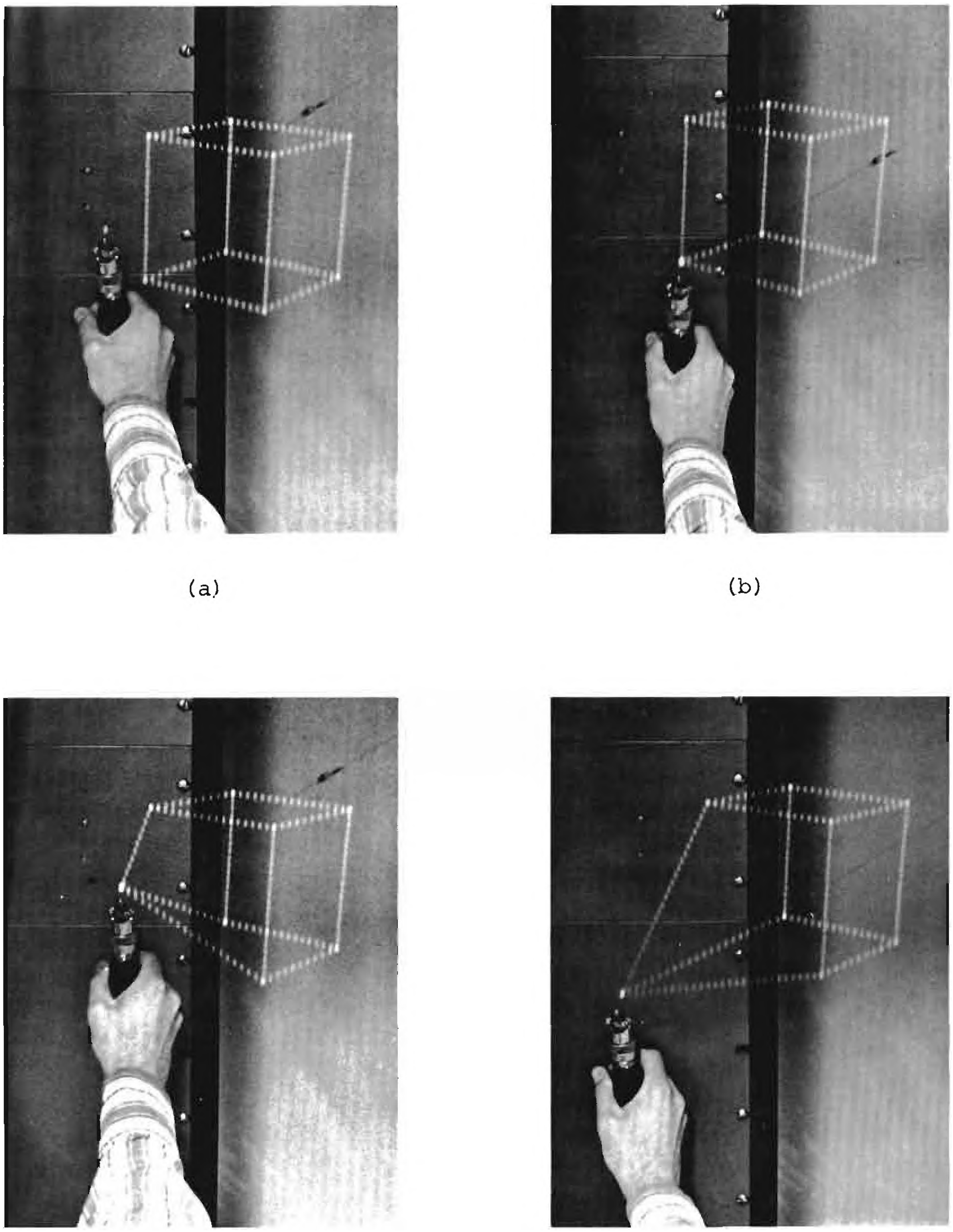


Fig. 12: Photographs taken through the head set showing effects of the DEFORM command. The observer is: (a) approaching the cube, (b) "touching" a vertex of the cube, (c,d) DEFORMing the cube.

DRAW -- This mode lets an observer DRAW synthetic objects (Fig. 13), add to existing objects, ERASE lines, and DRAW curves (Fig. 19). Eventually it will allow him to add vertices along existing line segments by making two line segments out of one long one. Lines which are drawn appear perfectly straight because of the use of "elastic line" drawing [14].

TINKER TOY -- Under this mode synthetic objects are rigid in form. An observer can move them along arbitrary 3-D paths; eventually he will be able to ROTATE them relative to an arbitrary origin, and to FUSE them together to create more complicated objects and structures.

MOTATE -- The name of this mode is an an acronym of "motor rotate" and the mode will be used to simulate sustained rotational motion. An observer will be able to INDICATE or define an axis and cause an object to rotate at a specifiable angular velocity about this axis. Rotation then will continue regardless of subsequent mode changes or commands until the object is specifically commanded by the observer to STOP, change velocities, or change direction. All the objects will be able to rotate simultaneously at different angular velocities and about different axes.

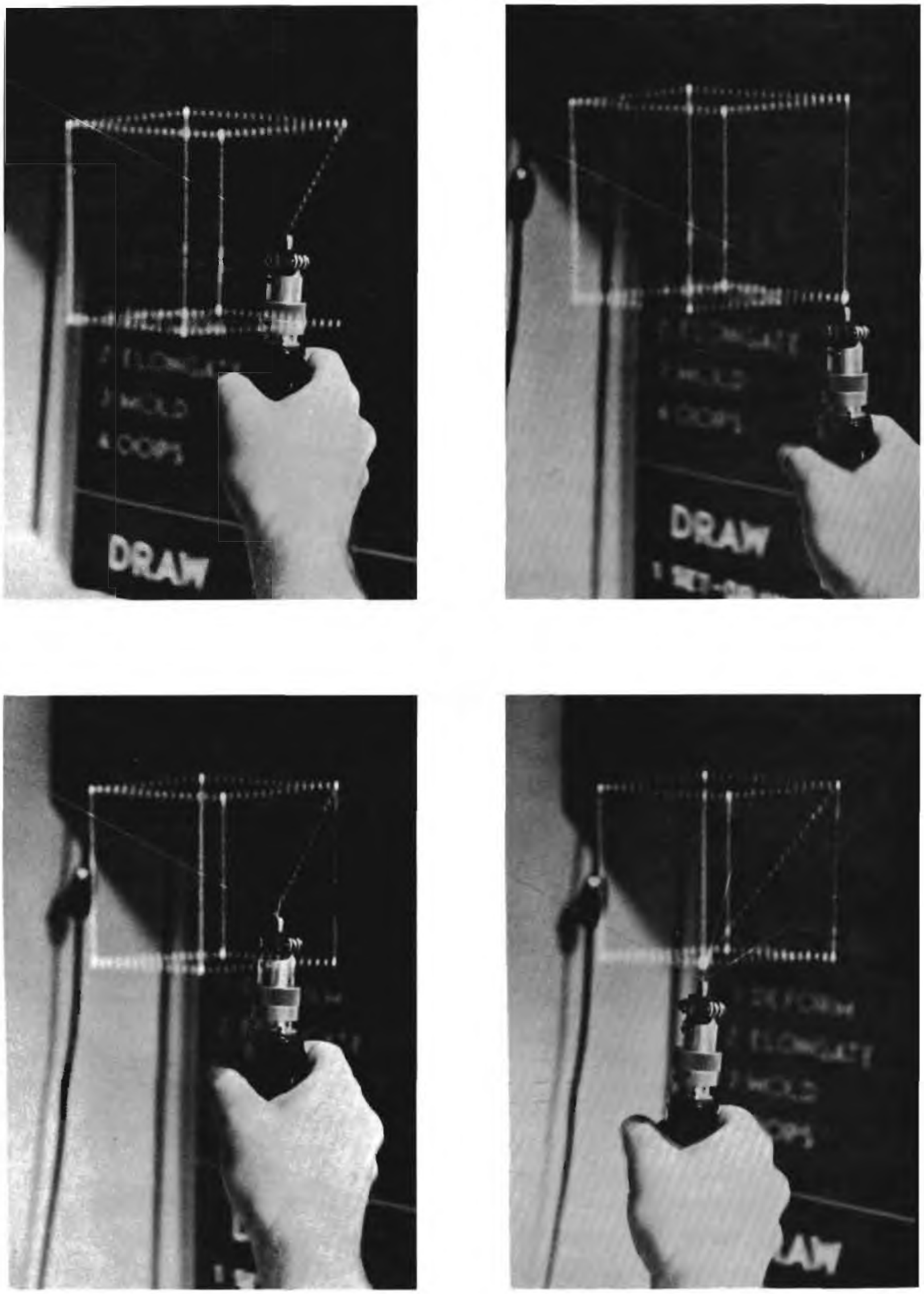


Fig. 13: Sequence showing effects of the DRAW command.

Some commands were thought to be of sufficient value that they weren't restricted to just one mode. These commands, called extensions, allow the observer effectively to extend his arm in any mode and interact with objects that are out of reach. The three extensions (Fig. 8) can be compared to the shift key on a typewriter. However, when the shift key (slide switch) is pushed, only the one potentially active extension is usable. Designation of which extension is potentially active is done in a manner similar to a mode change. Thus, there are always two synthetic crosses visible on the chart, one specifying the current mode and one specifying the potentially active extension.

Of the three extensions, ZOT, ZOOM and SCALE, only ZOT is presently implemented. After pushing the slide switch on the wand, the observer will be able to point at a synthetic object and scale it up or down if SCALE is the potentially active extension. If ZOOM is potentially active, the same action on the part of the observer will cause the synthetic object to zoom nearer to or farther from him along the path designated by his right eye and the wand cursor. The ZOT extension, similar to "cylinder mode" used in Johnson's system [3], attaches the cursor to any vertex whether near to or far from the observer whenever he points at the vertex and pushes a button. ZOT, therefore, lets an observer "touch" a vertex beyond his reach.

The wand contains a small potentiometer designed for use with extensions. The potentiometer, though not now functional, was added to allow the observer to indicate the sense and/or velocity of the SCALE, ZOOM, or ZOT. For instance, an observer will be able to point at a synthetic object and, by turning the potentiometer with his thumb in one direction, will make the object appear to grow. The further he turns the potentiometer, the faster it will grow. Turning the potentiometer in the other direction will similarly make the object shrink. The potentiometer will have a dead zone centered about its position at the time the slide switch was pushed. Until the potentiometer is turned past the dead zone in either direction, no growing or shrinking will occur; similarly, growth or shrinking will stop only when the dead zone is again reached. When ZOOMing, the potentiometer will indicate direction and velocity. For the ZOT extension, it will act as the means for the cursor leaving the wand and attaching to a vertex or leaving the vertex and assuming its normal position over the wand. Buttons currently serve to attach or retract the cursor.

Wand Data Structure

In Sorcerer's Apprentice, a dual-copy data structure is the key to real time interaction between the observer and the synthetic objects. This data structure* stores definitions of each synthetic object in both local coordinates and room coordinates (Fig. 4). Local coordinates are those whereby each object is defined relative to its own central, "left hand" coordinate system (Fig. 14). Local coordinate descriptions are necessary for easier implementation of such commands as SCALE and MOLD. The room coordinate description, on the other hand, facilitates the DEFORM and DRAW commands. The room coordinate description is set up in such a way that it also serves as the display file for the display

*See Appendix B for a detailed explanation of the dual-copy data structure.

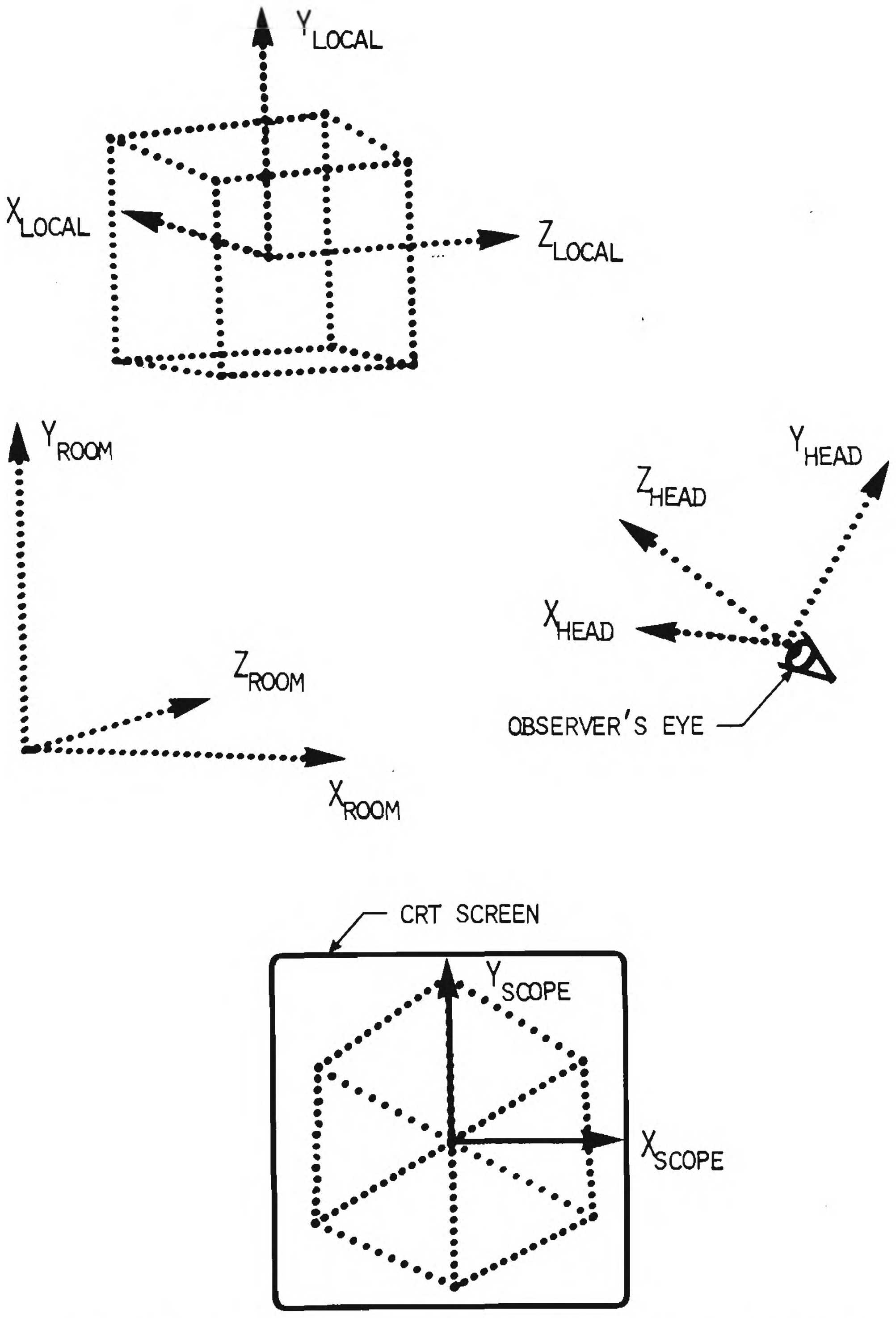


Fig. 14: Cube shown relative to local, room, head, and scope coordinate systems. This drawing was made by actually positioning the representative objects and photographing them; labels and the CRT view were added later. All photographed (dotted) vectors are two inches long.

file processor. Thus, an important feature about the dual-copy data structure is that it allows object description changes in either local or room coordinates with little cost in computing time. Another nice feature is that the format of the display file, presently conforming to the Evans and Sutherland LDS-1 hardware display processor [16], is independent of the data structure.

The problem with maintaining two descriptions of each object is keeping them in agreement. Going from local to room coordinates, this problem is solved using an object position matrix and a hash addressing scheme.* The object position matrix (Fig. 4,15) specifies the orientation of an object's local coordinate system with respect to the room coordinate system.

Hashing provides access to a rich set of pointers which link room coordinate endpoints having certain similar characteristics. Pointers chain together all those endpoints having the same position, having different positions but identical hash codes, belonging to one synthetic object, and in the process of being modified, e.g. by the ELONGATE command. A scatter index table [15] bridges the gulf between the random ordering of data characteristic of hashing and the sequential ordering of data used by the display file processor.

The agreement problem when going from room to local coordinates also has a two part solution. Pointers set up after the hashing operation link each room coordinate endpoint with its local coordinate counterpart. It is the inverse of the object position matrix which is used to transform data from room to local coordinates. Fortunately,

*See Appendix B for more detail about the hash addressing scheme.

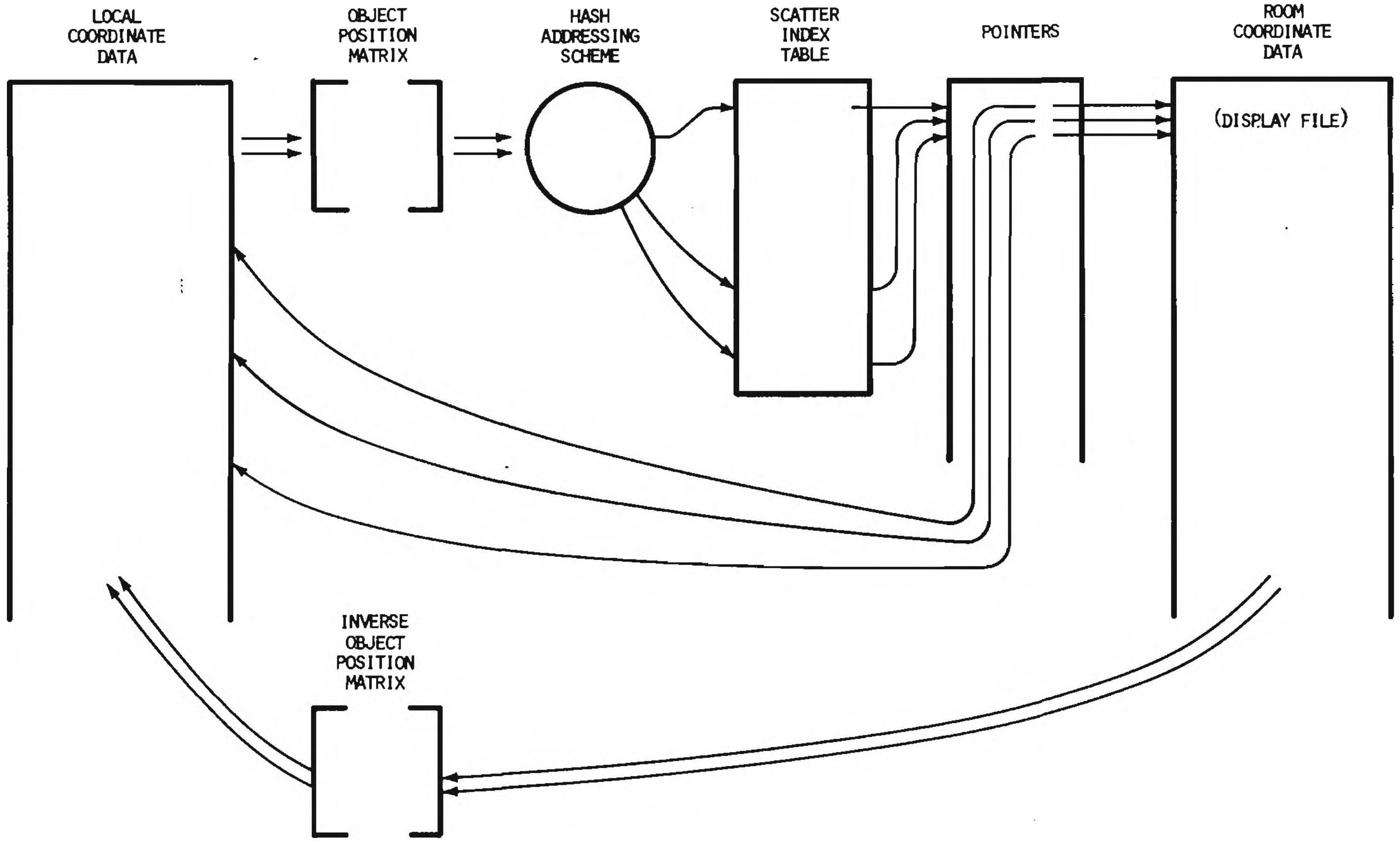


Fig. 15: Simplified diagram of dual-copy data structure.

the inverse of the rotation portion [7] of the object position matrix is equivalent to the transpose of that same portion and, therefore, little effort is required to form the inverse of the object position matrix.

CHAPTER IV
DEVELOPMENTAL PROBLEMS

Before the wand could be added to the head-mounted display system to form a working Sorcerer's Apprentice system, four major problems needed to be solved. Some of these problems were mentioned in earlier chapters but will be discussed again here for completeness. These four problems are: (1) the wand-computer communication problem, (2) the data structure problem, (3) the "no hardware" problem, and (4) the 3-D "touching" problem.

Wand-Computer Communication Problem

A problem, introduced under "wand language," was that of communication from wand to computer. One accustomed to manipulating 3-D objects with combinations of teletypes, console switches, light pens, joy sticks, 2-D tablets, and various buttons, knobs, potentiometers, and switches is fitted with a head set and given a wand in his right hand. How effectively can a wand with only four buttons, a switch, and a potentiometer replace a teletype or rows of knobs and switches?

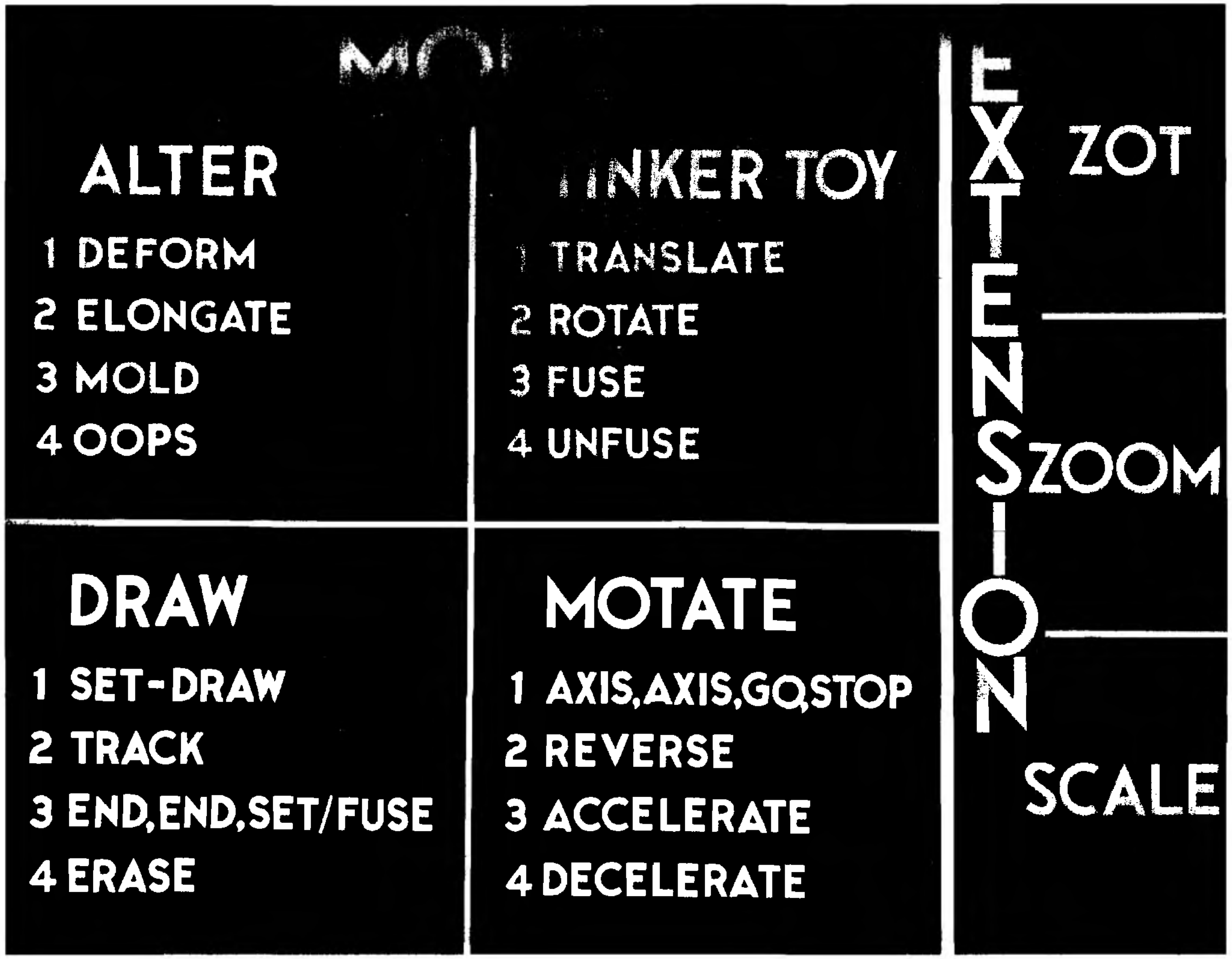
In the question, the inference that the wand replaces a teletype or rows of switches is somewhat misleading. First, the head-mounted display and not the wand replaces standard knob-controlled viewing techniques. Next, most cursor control functions are accomplished by the observer's natural 3-D pointing motion and have nothing to do with the number of buttons on the wand. However, since the wand is the only device by which the observer can send commands to the computer, the wand must be capable of sending more than just four commands. Wall chart interaction was used to multiply the effective number of wand buttons thereby eliminating the need for something like a portable keyboard.

Interaction with charts was chosen as a solution to the wand-computer communication problem because it is simple, expandable, and flexible. With a chart for reference, one needs to remember the function of only four buttons at once. The whole room could be filled with charts if desired. For flexibility, the chart could be replaced with a blackboard, chalk and eraser. With sufficient processing capability, a synthetic chart could appear on the wall. For this synthetic chart, the wand could be made to simulate the functions of chalk and eraser.

The chart in Figure 16 was made by using fluorescent letters on a black background. An ultraviolet light is mounted above the chart so the letters can be made to fluoresce under conditions of low ambient room light. These conditions are sometimes used to enhance the contrast between synthetic objects and furnishings within the surrounding room.

Not only does interaction with wall charts conveniently solve the problem of wand-computer communication, but it also demonstrates the capabilities of interacting with inanimate objects and of superimposing synthetic objects onto real ones. These two capabilities may prove to be among the most valuable contributions of Sorcerer's Apprentice.

Even the design of the wand helps somewhat with solving the wand-computer communication problem. The buttons offer a variety of inputs which are quick and require little motion on the part of the observer. The potentiometer, adjustable with the right thumb, fulfills the need


MO

ALTER
1 DEFORM
2 ELONGATE
3 MOLD
4 OOPS

TINKER TOY
1 TRANSLATE
2 ROTATE
3 FUSE
4 UNFUSE

DRAW
1 SET-DRAW
2 TRACK
3 END,END,SET/FUSE
4 ERASE

MOTATE
1 AXIS,AXIS,GO,STOP
2 REVERSE
3 ACCELERATE
4 DECELERATE

EXTENSION
ZOT
ZOOM
SCALE

Fig. 16: View of the wall chart showing the four modes and three extensions.

for a variable input to the computer, and the slide switch serves as a latching device which is ideal for extension and mode changes.

Data Structure Problem

A major problem in developing Sorcerer's Apprentice was that of finding a suitable data structure. After investigating many varied and exotic structures it became apparent that the crux of the problem was whether to keep in memory one or two copies of the data describing the synthetic objects. The choices were to keep a copy in room coordinates, in local coordinates, or in both. In making a decision it was noted that an object will shift while being scaled unless it is defined in local coordinates. If in room coordinates, the data would have to be transformed into local coordinates before scaling. However, when one is trying to "touch" an object, local coordinate descriptions are undesirable since the wand position is known only in room coordinates. In the case of "touching," the wand position would have to be transformed by the inverse "object position matrix" associated with an object and then checked in local coordinates to see if some "touching" criteria were met. If several synthetic objects were present in the display file, this transforming and checking would have to be done for each object. Therefore, to avoid needless transformations from one coordinate system to another, it would be nice to have both local and room coordinate descriptions of all synthetic objects.

For the purposes of Sorcerer's Apprentice the advantages of a dual-copy data structure outweighed the disadvantages. A data structure with just one copy of the data is easier to work with and requires less computer memory than the dual-copy structure. But with all the high speed, special purpose hardware now inoperative*, computing time for the many transformations required by the single-copy data structure could not be spared. In addition to reducing the number of time consuming transformations from local to room coordinates, the dual-copy data structure had the advantage of serving as the basis for solutions to the remaining two developmental problems, the "no hardware" problem and the 3-D "touching" problem.

With the decision to use the dual-copy data structure came the problem of keeping the two copies in agreement since they both would be subject to change directly from wand commands. Under "wand data structure," Chapter III, it was explained that a hash addressing scheme was employed as a solution to the agreement problem.** This scheme, detailed in Appendix B, not only greatly reduces the agreement problem, it also makes the touching problem trivial. The scheme also represents an unusual feature of Sorcerer's Apprentice, the successful application of hash addressing to geometrical data.

"No Hardware" Problem

When the high speed head-mounted display hardware was removed temporarily, a serious question arose: Can any progress be made in the Sorcerer's Apprentice experiment without real time displaying capabilities? Surprisingly, the question did not need to be answered; even with software

*See "Head-Mounted Display Hardware," Chapter II.

**See the following section for more about the agreement problem.

simulating the removed hardware, it is possible to display up to 24 flicker-free, screen-length vectors.* However, at this point it was even more critical that the wand system require as little processing time as possible.

The problem of reducing wand system processing time had a three part solution. First, as explained in the next section, the dual-copy data structure significantly reduces the time necessary to check "touching." This savings in time is made possible principally by the use of the hash addressing scheme. Second, during wand interaction, changes are made only in that copy of the data which is most easily altered. This, of course, is one of the advantages of the dual-copy data structure. Third, the state table** used in acting upon the data structure performs three actions for each push of a wand button. The "initial" and "terminal" actions handle transformations and complicated maneuvers. However, the "continual" action is very brief.

By looking more carefully at the problem of agreement between two copies of data one sees that the three action state table can be applied effectively to reduce that problem also. Agreement is necessary only before a wand command is executed. At that time the data structure, and therefore the state table, must be prepared for either a room coordinate or a local coordinate referencing command. However, during execution of a command such as DEFORM, where the command is continually executed as long as the button is depressed, it is necessary to change only the room coordinate copy of the data until the deformation is completed.

---

*Up to 70 vectors can be displayed with "tolerable" flicker. See "Vector Generator," Chapter II and "Interaction," Chapter IV.

**See "Wand Language," Chapter III.

Only after completion is it necessary to use transformations to bring the local coordinate copy up to date. Similarly, commands which reference the local coordinate copy need not disturb the room coordinate copy during sustained execution of the command. Application of the three action state table to the agreement problem is quite simple. For a given wand command the "initial" state table action prepares the appropriate copy of the data so the computations made by the "continual" action will be very few in number. The "terminal" action is the one which then brings the two copies into agreement by making use of the hash addressing links to transform any change made in one copy to the other. Since only the "initial" and "terminal" actions are involved with agreement of the two copies of data, the three-action state table prevents the agreement problem of the dual-copy data structure from slowing down the display rate.

Efficiency of the dual-copy data structure and three-action state table in reducing the agreement problem and, thus, the wand system processing time is remarkable. For example, with the DEFORM command, the "initial" and "terminal" actions each take typically 100 times more computer time than the "continual" action which affects only the room coordinate copy. The addition of the wand processor to the head-mounted display software slows the refresh rate down by only 1.0 percent. Continual execution of the DEFORM command slows the refresh rate down by an additional 1.2 percent.

3-D "Touching" Problem

Without the electronic cue available from a light pen on a 2-D display, the problem of "touching" displayed objects becomes difficult.

The addition of another dimension further complicates the problem. With a dual-copy data structure we have the advantage of working in room coordinates, but even with room coordinate descriptions of all objects, how can the computer determine when the wand is "touching" an object?

Before the "touching" problem can be solved, it is necessary to define exactly what is meant by "touching." A "sensitive cube" was defined to be centered about the wand cursor and to move with the cursor in such a way that the cube is always aligned with the room coordinate system. Any point of a synthetic object which falls within this "sensitive cube" meets the "touching" criteria. However, if several points meet the "touching" criteria simultaneously, only one of them may actually be considered as "touched" by the wand. Currently only endpoints of line segments can be "touched" in this manner. Figure 17 shows a 2-D representation of the "sensitive cube" and endpoints meeting and not meeting the "touching" criteria. This "sensitive cube" may be varied in size by the observer while using Sorcerer's Apprentice. One of the implementation goals was to let the potentiometer be the means of variance.

An "obvious solution" to the "touching" problem would be to compare the position of the wand with that of every endpoint in the display file to see which endpoints are within the "sensitive cube." For a few simple objects or with the help of special comparison-making hardware [16], this solution may work. Otherwise, comparison time for a large number of endpoints could severely limit CRT refresh rate.

Fortunately a very efficient software solution was found using properties of the hashing scheme and pointers (Fig. 15). The X, Y,

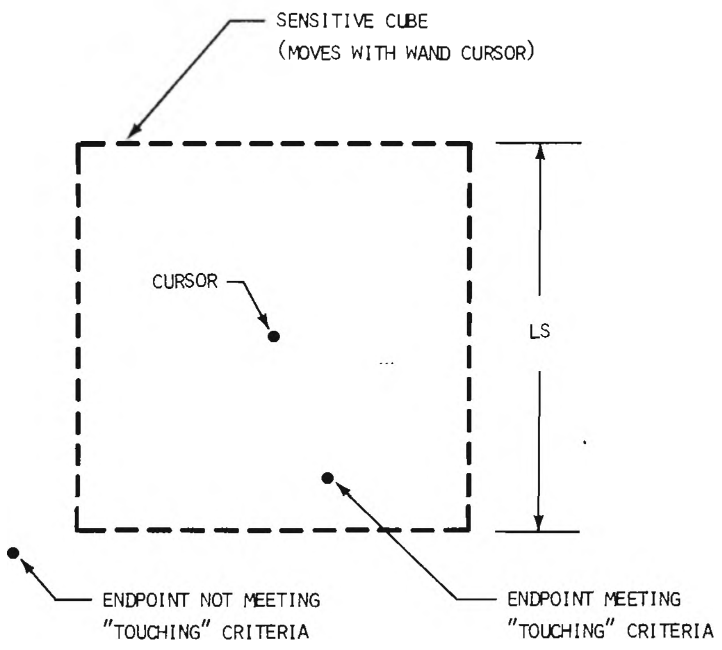


Fig. 17: Two-dimensional representation of wand cursor and "sensitive cube."

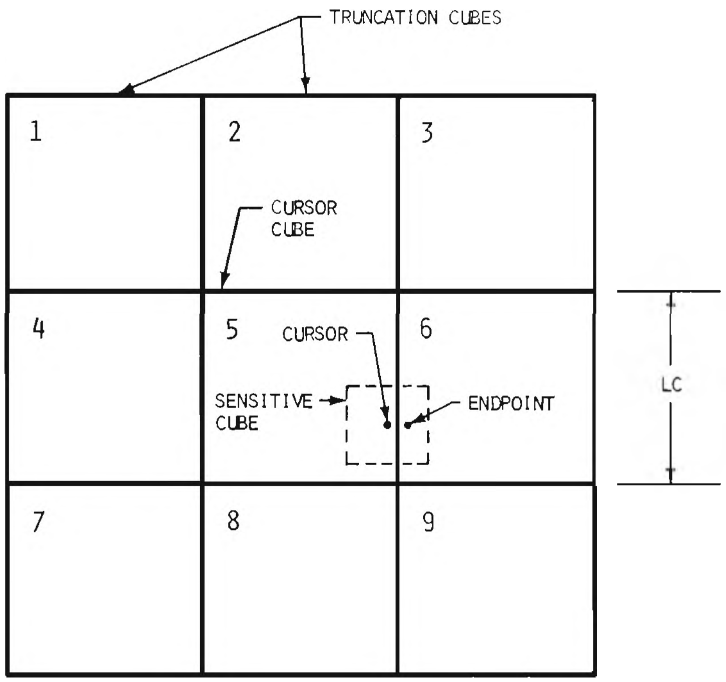


Fig. 18: Two-dimensional representation of wand cursor and truncation cubes.

and Z values of wand position as well as of all object endpoints are truncated; this truncation divides space into small stationary "truncation cubes" and gives the same hash code to all points within a cube. Thus by checking the wand position against those points linked by "same hash" pointers, one is really checking only those endpoints which lie within the truncation cube currently containing the wand cursor or within other truncation cubes having the same hash code as the cursor cube.* The truncation cube currently containing the wand cursor is called the "cursor cube;" those truncation cubes having the same hash code as the cursor cube are called "same hash cubes."

The concept of cursor and truncation cubes may be cleared up by the 2-D representation in Figure 18. This figure shows a few of the truncation cubes which fill space around the observer. The cursor cube, as is the case with all truncation cubes, is immovable. However, as the cursor is moved from cube to cube, different truncation cubes will be designated as the cursor cube. In Figure 18, cube 5 is the cursor cube; if the cursor were moved into cube 6 that cube would then become the cursor cube. Truncation cube size, dependent on the number of bits truncated, is fixed before any object is entered into Sorcerer's Apprentice. The number of "same hash cubes" depends on the length of

*Cubes having the same hash code as the cursor cube form varying geometric patterns in space depending on how the truncation is done. For the case where X, Y, and Z are truncated and then summed, we have cubes forming diagonal patterns throughout space. For the case where X, Y, and Z are summed and then truncated, we don't get cubes at all; instead we get diagonal slabs. This latter method was found to make the act of "touching" a little easier and is currently used. Nevertheless, for the sake of simplicity, reference will still be made to individual truncation and cursor "cubes" in the text.

the scatter index table.* Most "same hash cubes" contain no endpoints and therefore will not impede the checking process.

Tests show that the truncation scheme makes far fewer checks than the "obvious solution" to determine if the wand is "touching" an endpoint. The "obvious solution" requires that the "touching" criteria be applied to each endpoint in the display file. The truncation scheme can reduce the checks to only a handful and often to none. Presently with a truncation of 9 bits out of 36, corresponding to a cursor cube length of one inch, and with an index table size of 1001, an average of only 33 endpoints are checked each frame when 1000 endpoints of data are used.

There is a minor problem inherent to the truncation scheme. It is that the "touching" criteria can be successfully met only when an endpoint is within the cursor cube or an adjacent "same hash cube." Thus, the endpoint in Figure 18 may avoid detection of being touched even though it is definitely within the sensitive cube. This problem can be totally removed simply by using the "touching" criteria on all endpoints within all truncation cubes enveloping the cursor cube. Thus by checking points within 27 cubes, one could be certain of finding any endpoint within a distance from the cursor equal to the length of the cursor cube. In the 2-D situation of Figure 18, this would correspond to checking points within all 9 cubes shown. For this solution to work, it is necessary that $LS \leq LC$ where LS is the "sensitive cube" length and LC is the cursor cube length. In practice, "touching" works best when $LS < LC$; a ratio of 1:4 has been found to work nicely.

*See "Wand Data Structure," Chapter III.

Also, in practice, only the six truncation cubes having surfaces common with the cursor cube have been checked in addition to the cursor cube. It was very recently discovered, however, that an observer can sense very little difference between a program checking just the cursor cube and one checking seven cubes. In the test resulting in 33 checks per 1000 endpoints, all points within the seven cubes were being checked. Certainly the number of checks would have been lower if only the cursor cube had been considered.

CHAPTER V
USING SORCERER'S APPRENTICE

Finding solutions to the communication, data structure, "no hardware," and "touching" problems was necessary before the first version of Sorcerer's Apprentice could be made operational. There are, however, numerous other problems, surprises, and breakthroughs which continue to appear during implementation and subsequent use of the system. The author hopes that the many faults pointed out will not downgrade the system in the mind of the reader. These faults are mentioned merely as guideposts for those developing similar systems.

Stereo

Without stereo and with the same view sent to both of the observer's eyes, the ability of the observer to judge depth is very poor. For example, an observer must do a lot of head moving to determine the shape and position of an unfamiliar synthetic object. A cube is easily recognizable from nearly any view, but when one corner of the cube is DEFORMED (Fig. 12) the observer must move around quite a bit to learn the new position of the corner.

The ability of the head-mounted display to superimpose synthetic objects onto real ones partially counters the depth judgement problem. A stationary observer looking only at a synthetic object cannot tell whether the object is small and near or large and far away. When seeing a two inch cube, most observers initially assume it to be large and far away. Even after moving around the cube, one or two observers still refused to believe the cube was small but instead believed it was moving as they moved. Only after a real object, such as the wand, was held near the cube would they admit that, indeed, it appeared small and stationary. For most novice observers this ability to compare synthetic objects with real objects seems to be an extremely helpful feature of the Sorcerer's Apprentice system.

Even with the problem of depth judgement, many observers discovered that for familiar shaped synthetic objects motion tends to supplant stereo. That is, most people who saw a cube through the head set for the first time did not realize they were seeing the same image with both eyes -- they actually thought they were viewing a stereo image.*

A related surprise is that even without stereo the head-mounted display removes the "inside-out" optical illusion common to 2-D line drawings. It was thought that if motion and perspective could be added to these drawings, the optical illusion could be eliminated. But watching a moving cube on a 2-D CRT monitor proved differently. Through the head set, however, when the motion was that of the observer, the same cube appeared with no "inside-out" illusion. Furthermore, when one CRT in the head set was turned off, many observers still felt they were seeing the cube with both eyes -- in stereo!

Undoubtedly stereo viewing would help with the problems of judging distance and recognizing unfamiliar shapes, however, stereo does have some disadvantages. One disadvantage, in addition to increased costs, is that stereo requires twice the computation and display time for

*This thought was undoubtedly weighted by their observation that the head set was designed for stereoscopic viewing.

every function from the matrix multiplier to the head set. Right now, the addition of stereo would reduce the number of flicker-free vectors to almost one half.

Visual Feedback

Visual feedback plays an important role when interacting with synthetic objects. As used here, visual feedback is the information gleaned by an eye which can be used to confirm or reject an event or situation. For example, it is visual feedback which imparts realism to synthetic objects when they appear on, near, or inside real objects. In this case the eye confirms the stationarity of a synthetic object by comparing it to a real object.

Two other visual cues have been programmed to aid an observer in "touching" a synthetic object. Whenever the vertex of an object is found within the "sensitive cube" (Fig. 17) centered about the wand cursor, the cursor appears to "jump" to the vertex as though drawn by a magnet. The moving wand no longer influences the cursor until the "sensitive cube" passes by the vertex at which time the cursor "jumps" back and continues moving with the wand. The second cue appears after the cursor has "jumped" to a vertex. This vertex is then displayed as a bright spot (Fig. 12). Without these two cues, it is very difficult for an observer to know when he has "touched" an object, and drawing a line from an existing vertex is nearly impossible.

Some visual feedback has a negative effect. Because of error in determining wand position or head position and orientation, for instance, the wand cursor does not always stay directly above the wand.

This discrepancy in the relative positions of the wand and cursor is distracting, however, only if the variation is more than a couple of inches. Even when the variation is great, an observer can reduce the negative effect by lowering a blinder which covers the head set eyepieces. This blinder, visible in Figure 2, blocks the observer's view of the surrounding room and allows him to see only the synthetic objects. Thus, the blinder offers the advantage of reducing visual problems of erroneous wand tracking and the disadvantage of not allowing visual superposition of real and synthetic objects. Veteran users of Sorcerer's Apprentice prefer the blinder down when working with familiar shaped objects.

Optics

Much of the reality of the head-mounted display illusion stems from the quality of the head set optics. These optics present to the observer a remarkably good view of what is actually drawn on the miniature CRT screens. Because the small cathode-ray tubes have a one mil spot size, each CRT is able to display with about the same resolution obtainable by a standard 20 mil spot on a 15 inch screen. Nearly the whole picture is visible to an observer as long as the pupils of his eyes are within the "exit pupil" which is a small conical volume extending toward the observer from each eyepiece.

The head set optics do impose several limitations, one of which is a restriction of the observer's field of view. To obtain a reasonably large exit pupil it was necessary to limit the observer's field of view of synthetic objects to about 40 degrees both horizontally and vertically. His field of view of objects within the room is limited by the shape of the head set to about 100 degrees vertically but is essentially unrestricted horizontally. This restricted field of view is not a problem when looking at a specific object but it does require the observer to turn more than he might otherwise in order to get a panoramic view.

The exit pupil is sufficiently long that those wearing eyeglasses may leave them on while using Sorcerer's Apprentice. However, if glasses are left on, the head set harness cannot conveniently be used. The author prefers to heave his eyeglasses on and hold the head set in place with his free hand. Currently there is no way of adjusting the optics to compensate for visual defects of the observer.

Two things need to be made clear concerning the focal length of the head set optics. First, a single eye looking through a single eyepiece must focus near infinity to see a sharp image. This focal length was demonstrated by placing a camera behind an eyepiece and focusing on a synthetic object. Second, as an observer is looking at a synthetic object with both eyes, the distance from his eyes to the point of convergence of his eyes is always three feet. This distance will be referred to as the interocular convergence length.

An interocular convergence length of three feet means that only when the wand is held at that distance from the observer, will he see both the wand and the synthetic object as single images. When the wand is held closer, the observer will look at it and see a double image of the synthetic object or will look at the object and see a double image of the wand. The author can tolerate a slight double image, but when the wand is within about one foot of his eye, he must close one eye or lower the blinder.

A focal length near infinity causes problems with an observer and with a camera. The problem to the observer is that from a standpoint of focus, synthetic objects never appear real. When he looks at real objects from a distance of three feet, his focal length is also at three feet, but when he adjusts his eyes so that a synthetic object appears as a single image, it is still out of focus -- similar to the appearance of a real object seen at a distance of three feet by a nearsighted person without eyeglasses. It turns out, however, that this is more of an aid than a problem because dots out of focus blend better to simulate solid lines. In fact, the dots which are currently used to simulate vectors are electronically defocused to enhance the simulation. Similarly, the dots appear out of focus to a camera which is focused on objects within the room (Fig. 12).

Another distracting feature about the optics system is that with an interocular convergence length of three feet, synthetic objects do not blur like real objects when looked at with just one eye from very close range. In fact, one is able to simulate the effect of piercing his eyeball with a synthetic line segment going directly away from him. This piercing effect is fun to observe and could possibly be a great advantage for getting closeup views of small detail in synthetic objects except for the practical limitations imposed by the resolution of the head position sensor and the observer's own inability to remain perfectly stationary. Under the present system, as an observer is attempting to pierce his eye with a line or trying to put his eye right on an endpoint, any slight motion of his head causes the synthetic image to jump from one side of his field of view to another. Possibly a future version of Sorcerer's Apprentice would allow an observer to use his own motion to get near some detailed portion of a synthetic object and then allow him to switch to potentiometer-controlled motion for a closer look at the detail.

Most of the annoying problems with optics could be overcome by using an optics system with separately adjustable focal and interocular convergence lengths. For most use of the Sorcerer's Apprentice system, even a fixed interocular convergence length of about 12 to 18 inches would help greatly.

Physical Restrictions

The most noticeable physical restrictions of Sorcerer's Apprentice are those imposed by the head position sensor. Although any normal motion of the head is possible within the restricted volume described by the head position sensor, off-axis extension of the sensor pulls against the observer's head. A head harness (Fig. 2) is often used to counteract the problem of pull by distributing it over a larger area of the user's head. Unfortunately, when adjusted to maintain rigid contact between head and head set, the harness is painfully tight and tedious to adjust. Without it, though, one's free arm gets very tired while holding the head set in position. Most observers do not last longer than about ten minutes without the harness. Mounting the head set on something like a flight helmet would probably aid immensely. Presently both harness and free hand are needed for extended use of the system; for quick looks and demonstrations of Sorcerer's Apprentice to visitors, the harness is seldom used.

We are hoping that one day the mechanical head position sensor will be replaced by something less restrictive, say an optical device using flashing lights and a scanning system like the "Burton Box" being implemented at the University of Utah [17]. An earlier attempt at Utah with an ultrasonic head position sensor [12,18] proved it to be unsatisfactory because of the sensitivity of the 40 KHZ signals to external noise and because of its inability to maintain consistent tracking. The mechanical device now in use is very difficult to adjust such that it gives the exact position and orientation of the head. Right now a cube will shift by two inches when viewed first from one side and then from the opposite side.

In spite of its inaccuracies and restrictions, the mechanical head position sensor has three distinct advantages. First, it is easy to maintain. Second, the error in its measurement does not grow in time and could be greatly reduced by adding more matrices to the eleven which are now combined to describe the transformation from room to head coordinates [6]. Third, the greatest advantage to the mechanical device is that it works now and has been working dependably for the past four years.

After comparing the "puppet" and ultrasonic tracking systems, one finds the majority of advantages in the "puppet" device. First, when using the "puppet" device the wand and cord weight, about 13 oz total, are counterbalanced by the spring-loaded "puppet" lines. Within about a two foot diameter sphere, the lines will hold the wand stationary when released from the observer's hand. Second, the "puppet" device will track properly after one of the lines has been bumped. The ultrasonic device, like its head position sensing counterpart, tends to lose tracking when anything passes between the transducer and the receivers. Loss of tracking is corrected by re-referencing the wand on a tripod set into three floor sockets (Fig. 1). Third, the "puppet" system promises to be more accurate once the problems of stretching and slipping of the lines are overcome. Even now, these problems are negligible for slow, steady, wand movement. Fourth, the "puppet" will track wand movement anywhere within the observer's reach whereas the ultrasonic tip must always remain in the vicinity under its "shower stall." The spring-loaded "puppet" lines do have a disadvantage in their tendency to influence one's hand motion along the direction of the line. Also, the lines are subject to slipping and stretching and are occasionally bumped by the head position sensor. With the blinder down, however, an observer is often unaware of such a bump. Still, with the "puppet" wand one is uninhibited within the majority of reachable volume. All in all, the "puppet" device is more reliable and more accurate than the ultrasonic device and is preferred by most.

In wand tracking, too, the "Burton Box" or a similar system could alleviate some problems. The box is designed to track the position of 64 lights at the rate of 15 times per second. Therefore, three light could serve for determining head position and orientation and a fourth for determining wand position. Eventually each finger might be equipped with a light to allow an observer to "grab" a synthetic object and manipulate it with his hand.

Interaction

Drawing 3-D pictures is an unusual experience. Some things which are easy on 2-D systems are very difficult using Sorcerer's Apprentice and vice-versa. For example, it is very difficult to draw free-hand planar figures with the wand. Such drawing could be aided by adding a command that would automatically force designated lines to be coplanar with both a reference line and a reference dot.

Difficulties in "touching" a specific vertex seem to be reduced by ZOTting instead. Even with visual feedback cues, it is somewhat difficult to "touch" a specific vertex because of one's inability to judge depth. Depth cues from an observer's own motion are ineffective because he remains stationary while "touching." Stereo might help but according to Noll [5], the act of "touching" is difficult even with stereo. Of course, Noll's system lacked the superposition of the control and observation volumes found in Sorcerer's Apprentice. However, the ZOT extension offers a practical alternative to the difficulties encountered in trying to "touch" without depth judgement. ZOT works by pointing at a vertex and letting the program find its depth. For comparative purposes, three students who had never before looked through the head set were asked to DEFORM a cube to a dot first without and then with the aid of ZOT. The unaided deformation was completed in about 10 minutes by only one student; the other two gave up after that length of time. The aided deformation took between two and five minutes. The same task took the author 45 and 80 seconds, respectively. The additional time taken by the author using ZOT can be explained by the additional wand switch manipulation required and by the author's unfamiliarity with the newly implimented ZOT extension. For "touching" vertices of unknown location, ZOT is extremely useful to both new and experienced observers.

With the DEFORM command it is very easy to change the shape of a cube but it is very difficult to change it to some other meaningful shape. To move the vertex of a cube and then put it back where it was is next to impossible. Also, as in the case of the planar figures above, it is virtually impossible to DRAW any perfect geometric shape. If regular shapes are desired, there are several methods through which they may be obtained. First, Sorcerer's Apprentice might be expanded so that a cube could be made to appear by pointing to a wall chart picture of a cube. Second, geometric constraints [14] might be used whereby one could indicate the approximate size of a cube and invoke a command to draw the precise object. Third, one could use panel switches or a teletype to create known shapes, though this method removes the freedom of an observer to interact from wherever he might be.

Some useful tasks can be done with relative ease using Sorcerer's Apprentice. One such task is to DRAW lines between existing vertices, e.g. to DRAW the diagonal of a cube. Filling space with a 3-D grid of visible, immovable points which become bright when "touched" would allow an observer to DRAW a cube, a building, or a piece of furniture having predominantly rectangular features. One observer was able to DRAW a perfect cube in this manner in only 75 seconds. Of course, it is a trivial matter to trace 2-D maps or photographs with the wand and tracing many 3-D objects is nearly as easy. For example, a chair was traced in less than one minute. Also, simulating the intersection of two arteries [19] took a total of five to six hours over a two week period on a 2-D system but probably could be done in less than 10 minutes using a Sorcerer's Apprentice with, say, 200 flicker-free line capability. The two arteries were represented on a display by two square mesh sheets rolled into cylinders and were placed at an angle of 60 degrees to each other. The problem was to DEFORM the second so that it fit snugly against the first and to ERASE part of the first so there would be a clear passage through it into the second.

Although 24 was mentioned as the maximum number of flicker-free vectors, one can draw up to 70 screen-length vectors at 7 frames per second, which is the rate at which flicker becomes intolerable to the author. An even greater number of short vectors can be displayed in the same time. With the head-mounted display hardware in operation, the number of flicker-free vectors should be well over 600.

One's ability and therefore, to some extent, one's desire to interact with 3-D objects in a 3-D environment grows with repeated use of Sorcerer's Apprentice. Many first time users of the system feel disappointed or discouraged, though the large majority are awed and quite favorably impressed. One thing common to nearly all informed "first timers" is the slight timidity that comes from knowing of the 1,800 volts only a fraction of an inch from their temples while wearing the head set.

CHAPTER VI
FUTURE OF SORCERER'S APPRENTICE

It is intriguing yet somewhat difficult to imagine what it would be like to use a 600 flicker-free line Sorcerer's Apprentice judging from the 24 line version. Most use of the current version has been for brief demonstrations and debugging. With 24 lines there is just enough information to evaluate the data structure and get some ideas concerning wand interaction by trying out a few commands.

Presently the magnetic tape drive interfacing routines along with the facilities to DRAW, ERASE, DEFORM, TRANSLATE, TRACK, ZOT, and change modes are implemented. These commands appear very useful for such operations as altering the simulated intersection of two arteries. Some of the most useful commands not yet implemented appear to be the ZOOM and SCALE extensions, MOLD, FUSE, ROTATE, and MOTATE, and the ability to create basic 3-D shapes or "primitives" [20] by pointing to their pictures on wall charts.

By way of application, machinists, engineers, architects, medical researchers, and anyone working with 3-D objects and concepts are potential users of the Sorcerer's Apprentice system. They could use it to create, analyze, and modify (Fig. 12,13) these objects in real time and without the drudgery of updating hard copy. One medical researcher displayed an electrocardiogram as a function of time and was thrilled to walk around the resulting spiral, viewing it from many angles. Another researcher used the head set to simulate the "view" of a blind person "seeing" with the aid of an artificial eye now under development [21].

Since the head-mounted display system can simulate a very realistic 3-D environment, it would seem the perfect tool for human viewing of changing 3-D situations such as air traffic control [22] might require. The wand could be used to point at an aircraft, simulated by a triangle tracing a 3-D path in the controller's environment, in order to establish communication with that aircraft. On an aircraft carrier the 3-D environment could be a simulation of the airspace over the carrier. Through the head set, the officer in charge could see representations of all aircraft in the vicinity of the carrier. With the wand he could "pull" on a synthetic aircraft causing a message to be electronically relayed to the pilot as a suggested change of course.

Superposition of real and synthetic objects provides the tool for adding synthetic lines to such things as walls, maps, desk tops, schematics and typewriter keys [2]. With 3-D superposition, one could check to see how closely the computer model of an object fits the actual object or how closely an object was manufactured to computer specifications.

Graphical art for magazines and technical publications is another area wherein Sorcerer's Apprentice may be useful. For instance, Figure 14 was done by photographing a situation from a monitor CRT while an observer using the head set picked the appropriate view. The objects were positioned by the input program with the room origin at (0,0,0), the cube origin at (0,4,4), and the head origin at (4,0,8) inches. After the photographing of the cube and three coordinate systems, the observer walked over and looked down the Z-axis of the head coordinate system at the cube. The insert shown in a simulated CRT is exactly what he saw after the coordinate systems had been ERASED. Thus, with only the effort of walking a few steps, all the synthetic vectors -- each of which is two inches in length -- appear on a drawing in their true position and perspective.

There are several nontechnical applications for which Sorcerer's Apprentice seems particularly well suited. An obvious one is three-dimensional, wire frame art. Also, with lights on one's fingers, the "Burton Box" could help provide the means for modeling with synthetic clay or sculpting a piece of synthetic marble. Psychologists could vary the parameters of head position sensing and wand tracking thus causing objects to get smaller as one approached them. Such variations [6] are easy to make and could be used to see how people react to certain controllable phenomena involving motion and visual perception.

Wire frame drawings are also adequate for viewing 3-D Lissajous' figures (Fig. 6), mathematical equations, molecules, nervous systems, and for doing flight simulations and basic structural design and modification, but for some applications it would be nice to improve upon the quality of these drawings. It would probably be advantageous to add an optional algorithm [23] for removing lines which would be hidden from view if the synthetic objects were solids. To do this would require modifying the current line-segment oriented data structure to a polygon oriented structure. This hidden-line capability would allow a city planner or highway department engineer to view a layout as it would appear from any desired vantage point and to modify the plan if necessary. With the recent development of a real time hidden surface remover and continuous-tone picture generator [24], it is possible that a future version of Sorcerer's Apprentice will not be restricted to wire frame drawings. It would then be even more useful for visualizing and interacting with solid synthetic objects and structures.

In addition to including a hidden line algorithm and a continuous-tone picture generator in Sorcerer's Apprentice, there are many avenues for future development and research. Development could be made along the lines of a variable optics system, a helmet-mounted display, a more randomly distributed hash code scheme, and a working potentiometer. Future research should be done in the areas of holographic background, new commands, new constraints, interacting with wall pictures of primitives, and interacting with line segments at points other than just the vertices.

Getting back to the present, the first steps for an improved Sorcerer's Apprentice are to replace the temporarily removed special purpose hardware, to activate the stereo features, to change the head set focal lengths, and to find a less restrictive, more accurate head position sensor. Nevertheless, even without modification and with the problems and restrictions mentioned in the previous chapter, the illusion presented by the head-mounted display is very realistic and interaction with the wand is very natural. The Sorcerer's Apprentice system stands as a unique tool with exciting potential for 3-D interaction in a 3-D environment.

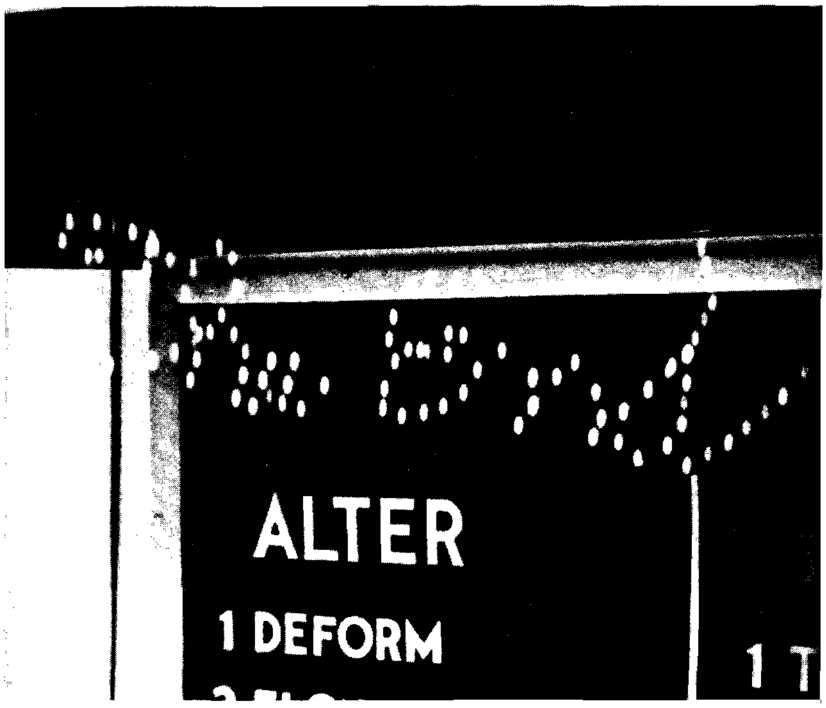
CHAPTER VII
CONTRIBUTIONS OF SORCERER'S APPRENTICE

With the major developmental problems besetting the Sorcerer's Apprentice system now solved, the system has several attractive attributes found in few 3-D graphic systems. From the head-mounted display system come the abilities to superimpose 3-D synthetic objects onto a real 3-D environment and to change one's view of such a synthetic object by one's own motion rather than by artificial means. Those attributes resulting from the addition of the wand system include the ability to create and modify 3-D structures directly by natural pointing motions rather than indirectly by pushing buttons or twisting knobs and the ability to interact with both real and synthetic objects in a 3-D environment.

Several hidden, though no less important, contributions need to be mentioned. The first is the application of hash addressing to geometrical data. From this application comes an easy and concise solution to the problem of "touching" a 3-D synthetic object with a wand. The hashing also offers an alternate software solution to the 2-D windowing problem [9] which confronts many computer-aided circuit board layout people. Another contribution is the development of a fairly extensive and expandable interactive graphics language which operates from the inputs of only four buttons, a slide switch, and a potentiometer. The conciseness of the language results from interaction with an inanimate wall chart. A dual-copy data structure and a three step state table help supply the mechanism for real time interaction using this language. A lesser contribution is the software package with the appropriate bells, whistles and mathematics to operate the head-mounted display.

Miscellaneous contributions and confirmations to the pool of general knowledge include the following: (1) a very realistic 3-D display can be made without stereo, (2) without stereo, motion and perspective added to 2-D wire frame drawings resolves the "inside-out" optical illusion only if the motion is that of the observer, (3) it is possible to have limited real time interaction with 3-D synthetic objects with only software.

A significant contribution of any first-generation system is the information gleaned from its development and use which is applicable to later generation systems. It is the author's hope that in this respect, also, will the Sorcerer's Apprentice experiment be useful.



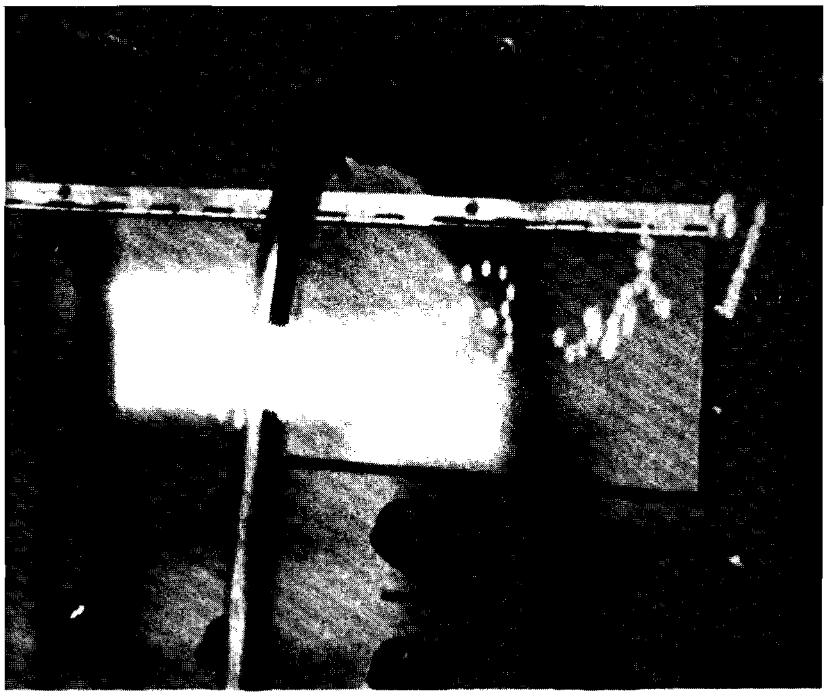


Fig. 19: Example of the TRACK command.

REFERENCES

[1] Riley, Wallace B. "Through a Glass, Lightly," *Electronics*, vol. 42, no. 25, 1969, p. 174.

[2] Sutherland, Ivan E. *A Head-Mounted Three-Dimensional Display*. Proceedings of the Fall Joint Computer Conference, vol. 33, pp. 757-764, 1968.

[3] Johnson, Timothy E. *Sketchpad III, Three Dimensional Graphical Communication with a Digital Computer*. S.M. Thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, Cambridge, Massachusetts, May 1963.

[4] *3 Axis Hand Control, Model 437*. Data Sheet, Measurement Systems, Inc., Norwalk, Conn.

[5] Noll, A. Michael. *Man-Machine Tactile Communication*. Ph.D. Dissertation, Department of Electrical Engineering, Polytechnic Institute of Brooklyn, June 1971.

[6] Vickers, D. L. *Head-Mounted Display Terminal*. Proceedings of the IEEE 1970 International Computer Group Conference, Washington, D.C. pp. 270-277, June 1970.

[7] Roberts, L. G. "Homogeneous Matrix Representation and Manipulation of N-Dimensional Constructs," *The Computer Display Review*, Adams Associates, May 1965.

[8] Sutherland, Ivan E., and Newman, William M. *Computer Graphics Technology*. Course notes for CS 200, University of Utah, Salt Lake City, Utah, Fall 1968.

[9] Sproull, R. F., and I. E. Sutherland. *Clipping Divider*. Proceedings of the Fall Joint Computer Conference, vol. 33, pp. 765-776, 1968.

[10] Vickers, D. L. *Sorcerer's Apprentice: Head-Mounted Display and Wand*. Proceedings of the First National Conference on Remotely Manned Systems, California Institute of Technology, Pasadena, California, September 1972.

[11] *GRAF/PEN Operation Manual*. Science Accessories Corporation, Southport, Connecticut.

[12] Baumgart, Bruce. *An Ultrasonic Head Position Sensor*. B.S. Thesis, Department of Applied Mathematics, Harvard College, 1968.

[13] Naur, Peter. "The Design of the Gier Algol Compiler," *BIT*, vol. 3, p. 160, 1963.

[14] Sutherland, Ivan E. *Sketchpad: A Man-Machine Graphical Communication System*. Technical Report No. 296, Lincoln Laboratory, Massachusetts Institute of Technology, January 1963.

[15] Morris, Robert. "Scatter Storage Techniques," *Communications of the ACM*, vol. 11, pp. 38-44, 1968.

[16] *Line Drawing System, Model 1*. System Reference Manual, Evans and Sutherland Computer Corporation, Salt Lake City, Utah, January 1970.

[17] Burton, Robert P. *Real-Time Determination of Multiple Three-Dimensional Positions*. Ph.D. Dissertation, Department of Electrical Engineering, University of Utah, Salt Lake City, Utah, June 1973.

[18] Sutherland, Ivan E. *Hardware for a Three-Dimensional Display*. Final Report on Contract XG-2972 Between the United States Government and the President and Fellows of Harvard College, August 1968.

[19] Greenfield, H., Vickers, D., Sutherland, I., Kolff, W., and Reemtsma, K. "Moving Computer Graphic Images Seen From Inside the Vascular System," *Transactions of the American Society of Artificial Internal Organs*, vol. 17, pp. 381-385, 1971.

[20] Romney, Gordon W. *Computer Assisted Assembly and Rendering of Solids*. Ph.D. Dissertation, Computer Science, University of Utah, Salt Lake City, Utah, August 1969.

[21] Mladejovsky, Michael J. Ph.D. Dissertation, Department of Electrical Engineering, University of Utah, Salt Lake City, Utah, June 1973.

[22] Vickers,Donald L. "Head-Mounted Display," *Utechnic*, University of Utah Student Engineering Magazine, Salt Lake City, Utah, vol. 10, no. 4, pp. 14-17, 1970.

[23] Archuleta, Michael J. *Hidden Surface Line Drawing Algorithm.* Technical Report UTEC-CSc-72-121, Computer Science, University of Utah, Salt Lake City, Utah, June 1972.

[24] Watkins, Gary S. *A Real Time Visible Surface Algorithm.* Ph.D. Dissertation, University of Utah, Salt Lake City, Utah, June 1970.

APPENDIX A
DETAILED LIST OF WAND COMMANDS

Communication from an observer to the computer via the wand is limited to four pushbuttons, a slide switch, and a potentiometer. To augment communication, modes are used. The commands invoked by pushing the four buttons are redefined by each mode.

When the slide switch is pushed up one can either use an extension or change modes. There is the possibility of confusion when one wants to change modes and synthetic objects exist between him and the chart. This confusion is resolved by restricting mode changes to the use of button B1 and extension control to the use of the potentiometer. The rest of the buttons become impotent during the time the slide switch is up. When the slide switch is pushed down, the mode in use prior to pushing it up will again be in effect. With the switch down, the potentiometer can control only the size of the wand "sensitive cube."

The commands invoked by pushing buttons B1 through B4 are described below. The activating button is indicated in parenthesis following the name of the command. Those that have actually been implemented at this time are indicated by an asterisk.

Alter Mode

This mode is used for changing the shapes of synthetic objects.

DEFORM* (B1) -- DEFORM has no effect if a point on a synthetic object is outside the "sensitivity cube." Otherwise, the coordinates of the first point found within the cube are replaced by the coordinates of the wand cursor as long as B1 is pushed. In other words, the vertex indicated is "dragged" by the wand and all lines initially attached to the vertex remain attached (Fig. 12).

ELONGATE (B2) -- This is similar to DEFORM except that a vertex will follow the cursor only along a certain major local coordinate axis of an object. Not just the point within the "sensitivity cube" is moved, but also every point which lies at least as far from the origin along the indicated axis as the "touched" point. Thus, a cube could be ELONGATEd in the direction of its positive X axis by "touching" one point having a positive X coordinate and moving the wand outward along the X axis. Once the group of vertices to be moved has been established by outward movement along an axis, the group will follow the component of wand motion along that axis until B2 is released. Thus, to "squash" a cube, one needs to "touch" a vertex, move outward along an axis to establish the axis and the group of vertices, and then reverse motion along that axis. A cube may be "turned inside-out" in this manner. Because of the nature of this command, objects will always maintain perfect geometrical shape, i.e., rectangualr parallelepipeds will always remain rectangular parallelepipeds. Once the axis and group of vertices are established, it is not necessary that the initially "touched" vertex remain within the "sensitive cube."

MOLD (B3) -- If a vertex is within the "sensitive cube" when B3 is pushed, this vertex will be "dragged" as with the DEFORM command and all the other vertices of that object will move relative to the object's origin in the same proportion. Thus, if the wand moves the "touched" vertex directly away from the origin, the object is scaled up; if the wand moves away from the object's origin in a path parallel to an axis, the object is lengthened both in the direction of the wand motion and in the opposite direction. The vertex initially "touched" with the cursor will remain coincident at all times until B3 is released.

OOPS (B4) -- As B1, B2, and B3 are pushed, a copy is made of the position, orientation, and shape of all objects. The OOPS button will cause the program to discard the most recent alteration and to display the copy saved before the alteration. Thus, if one DEFORMed an object by mistake, he could push B4 and return to the state just prior to the pushing of the DEFORM button, B1.

Draw Mode

This mode is for DRAWing and adding to 3-D objects.

SET-DRAW* (B1) -- Depression of B1 creates an "elastic line" of which the stationary end is given the cursor coordinates at the time of depression. The other end of the "elastic line" is the current cursor position. Thus, the line follows the cursor until B1 is released, at which time the line becomes fixed in space (Fig. 13). Connected lines are drawn by "touching" an existing vertex before pressing B1. Unattached lines or groups of lines are considered as new objects. However, any new line or group of lines which are DRAWn "touching" an existing object become part of that object unless specifically designated otherwise by pressing B4. Thus, two cubes connected by a line could be three separate objects, two separate objects, or all one object.

TRACK* (B2) -- This is a continuous form of SET-DRAW wherein a line may be drawn each time a frame is displayed. The line is drawn from the position of the cursor at the end of a frame or a specified number of frames to the position of the previous line. Thus, as long as B2 is depressed, an observer can DRAW 3-D curves made up of straight line segments (Fig. 19). The length of the lines is governed by the speed of the observer's motion and the refresh rate of the display.

END, END, SET/FUSE (B3) -- A line segment is specified by "touching" each end of it and pressing B3. After a line is thus specified, the line itself can be "touched" and by pressing B3 again a new vertex will appear on the line. If B1 is pressed after the specified line has been "touched," the observer can DRAW from that point. The lines DRAWn will become part of the object containing the specified line segment.

ERASE* (B4) -- By successively "touching" each end of a line segment and pressing B4, a line segment will disappear. By holding B4 down and carefully retracing a curve created by TRACK, the whole curve will disappear. B4 also serves as an object terminator when pushed immediately following a DRAW command.

Tinker Toy Mode

This mode is for manipulating and connecting existing objects.

TRANSLATE* (B1) -- If B1 is pressed while one is "touching" a vertex, the object or objects to which the vertex belongs follow the cursor. When B1 is released, the object again remains stationary. One can use this to station objects in the room or to connect objects together. Two objects thus connected, however, remain structurally separate and can be disconnected simply by TRANSLATEing one away from the other.

ROTATE (B2) -- ROTATE is a two-step command. The first depression of B2 defines an origin about which rotation occurs. If a vertex is within the "sensitive cube" as B2 is pressed a second time, a vector is created from that vertex to the previously defined origin. As long as B2 is held down, that vector will be rotated each frame to coincide with the vector from the wand cursor to the origin. Each time the vector is rotated, all the other vectors from the origin to all the other vertices of the indicated object will be rotated an equal amount.

FUSE (B3) -- Once objects are connected by TRANSLATE, DRAW, DEFORM, etc., the vertices common to two or more objects may be "touched" and FUSEd by depressing B3. At this time a reversible structural change in local coordinates occurs such that the objects behave as one. The new local coordinate origin is an average of previous local origins.

UNFUSE (B4) -- After "touching" the FUSEd vertices mentioned above, one can cause the objects to assume their separate identity by pushing B4. UNFUSE has no effect when applied to a vertex which was not previously FUSEd.

Motate Mode

This mode is for imparting a continuous motion to objects.

AXIS, AXIS, GO, STOP (B1) -- This four-step command will allow an observer to define an arbitrary axis of rotation with two pushes of B1. The third push is made when "touching" an object destined to rotate about this axis. Any rotating object may be stopped by "touching" its axis and depressing B1 a fourth time. Stopping in this manner cannot occur after the first or second step of this sequence but can occur at any other time.

REVERSE (B2) -- By "touching" an end of its axis of rotation, any object can be made to reverse its direction of rotation with no change in its final angular velocity. Pushing B2 without "touching" the axis will cause a reversal in the direction of the object whose axis was most recently "touched."

ACCELERATE (B3) -- In a manner similar to B2, B3 interacts with rotating objects to increase their angular velocity. Velocity will increase at a steady rate while B3 is depressed and will be maintained at a constant level upon release of B3.

DECELERATE (B4) -- This works identically to B3 with the obvious exception that it reduced angular velocity.

Extensions allow interaction by pointing instead of just by "touching." The three extensions now appearing on the wall chart (Fig. 16) are explained below.

ZOT* -- This allows an observer to point to a vertex and cause the cursor to attach itself to that vertex regardless of how far away the vertex is. The potentiometer determines whether the cursor is to leave the vicinity of the wand and move to a vertex or whether it is to leave the vertex and attach once more to the wand. Wand motion is not amplified when the cursor is attached to a vertex. That is, if an observer were DEFORMing an object five feet away, a one inch movement of the wand would cause a deformation of only one inch to the object.

ZOOM -- With the slide switch up, an observer can point to any vertex of an object and cause the object to move toward or away from him with the potentiometer. The motion is along the vector from the right eye to the cursor. Once the object has been indicated by pointing at a vertex, it is not necessary to continue pointing at that vertex or even at the object.

SCALE -- Very similarly to ZOOM, SCALE allows one to indicate an object by pointing and to increase or decrease its size by turning the potentiometer.

APPENDIX B
SORCERER'S APPRENTICE DATA STRUCTURE

The dual-copy data structure used in Sorcerer's Apprentice contains two copies of each object to be displayed, one in local coordinates and one in room coordinates (Fig. 14,15). The local coordinate copy contains the topology and the room coordinate copy serves as a display file. The two copies are linked in the local to room direction by the object position matrix which defines the object's position with repsect to the room and by a hash addressing scheme. In the other direction the link is through pointers and through the inverse object position matrix (Fig. 20).

In local coordinates, each object is listed in an object table (Fig. 21). Objects are identified by a number which also serves as an index to this table. For each object, the table contains the address of an object block.

Object blocks contain that information pertinent to a specific object. This includes the rotational and translational information used in the object position matrix and the scaling information which, for convenience of the MOLD command, is separate for X, Y, and Z. These blocks also contain pointers to the first and last elements in a chain of room coordinate point blocks and a pointer to the first element of the local line block chain. These two chains link all points and lines belonging to an object. For convenience of the FUSE command, forward and backward pointers exist which link FUSEd objects; for later UNFUSEing, the location of an object's center relative to the center of the conglomerate object is saved.

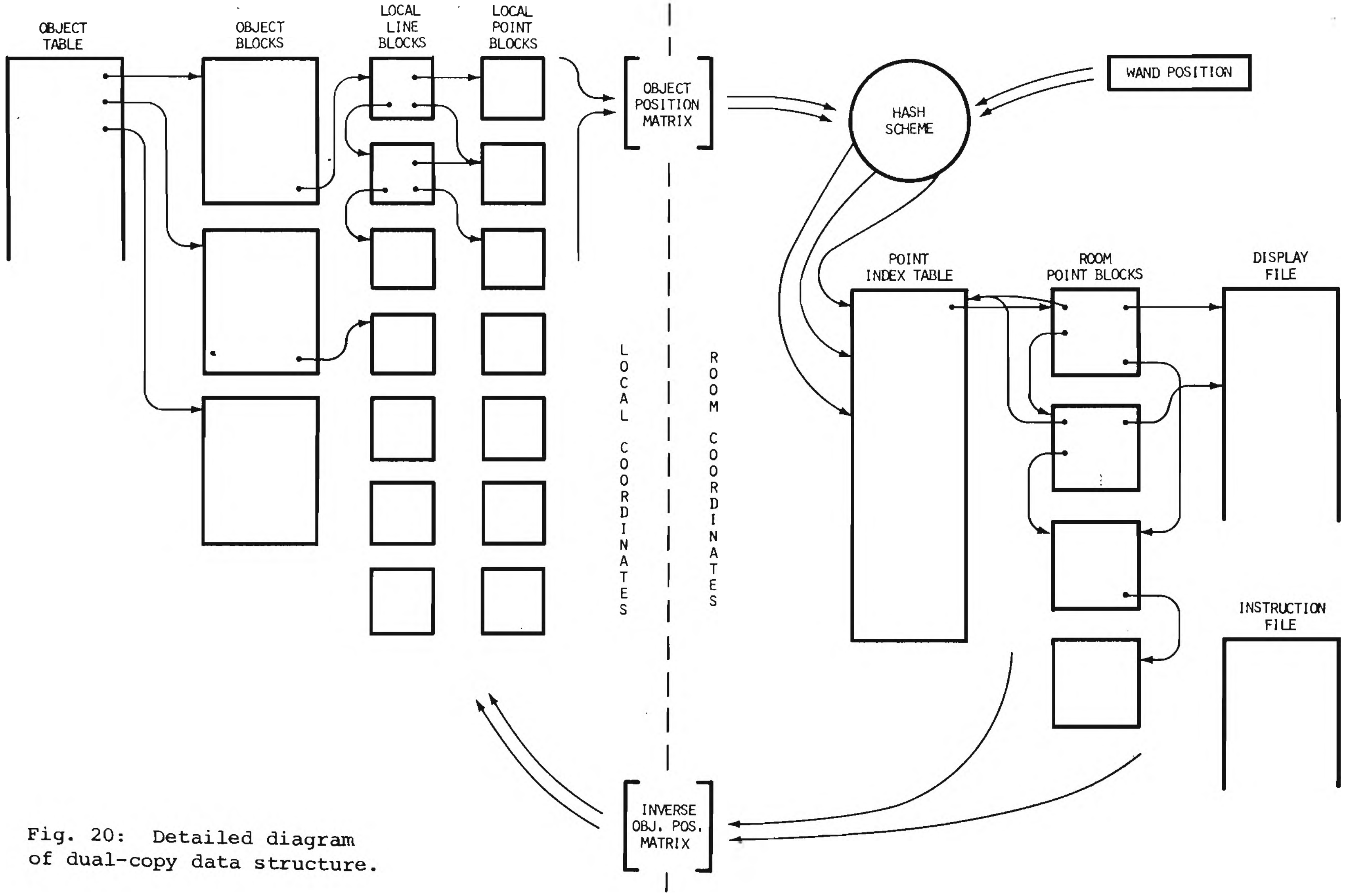


Fig. 20: Detailed diagram of dual-copy data structure.

Each unique line and each unique point of a object are stored in local coordinate line and point blocks (Fig. 21). Local line blocks contain pointers to the local point blocks where the X, Y, and Z values are actually stored. They also point in chain fashion to each other and have an "ERASEd?" marker. This marker is merely an indication that a line segment has been ERASEd and a cue for the program to eliminate the line the next time the object is multiplied by its object position matrix.

Transformation form local to room coordinates is done by multiplying the local [X, Y, Z] vector by the object position matrix. The resulting room coordinate X, Y, and Z values are summed, truncated, and then divided by the point index table length. The remainder is the hash code and is used as an index to the point index table (Fig. 22). This table, referred to as the "scatter index table" in an article by Morris [15], directs the random hash code to a sequentially formed chain of room coordinate point blocks.

Room point blocks, unlike local point blocks, contain pointers to the room coordinate X, Y, and Z values rather than containing the actual values. The values are stored in the display file. Another difference is that room point blocks are not necessarily unique for a given object. That is, if a point, e.g., the vertex of a cube, is attached to three line segments in an object, it will have three separate room point blocks but only one local point block. Room point blocks contain implicit line information in that the blocks are always created in line segment pairs. This manner of creation was used so the display file would meet the requirements of the display file processor which is line segment oriented.

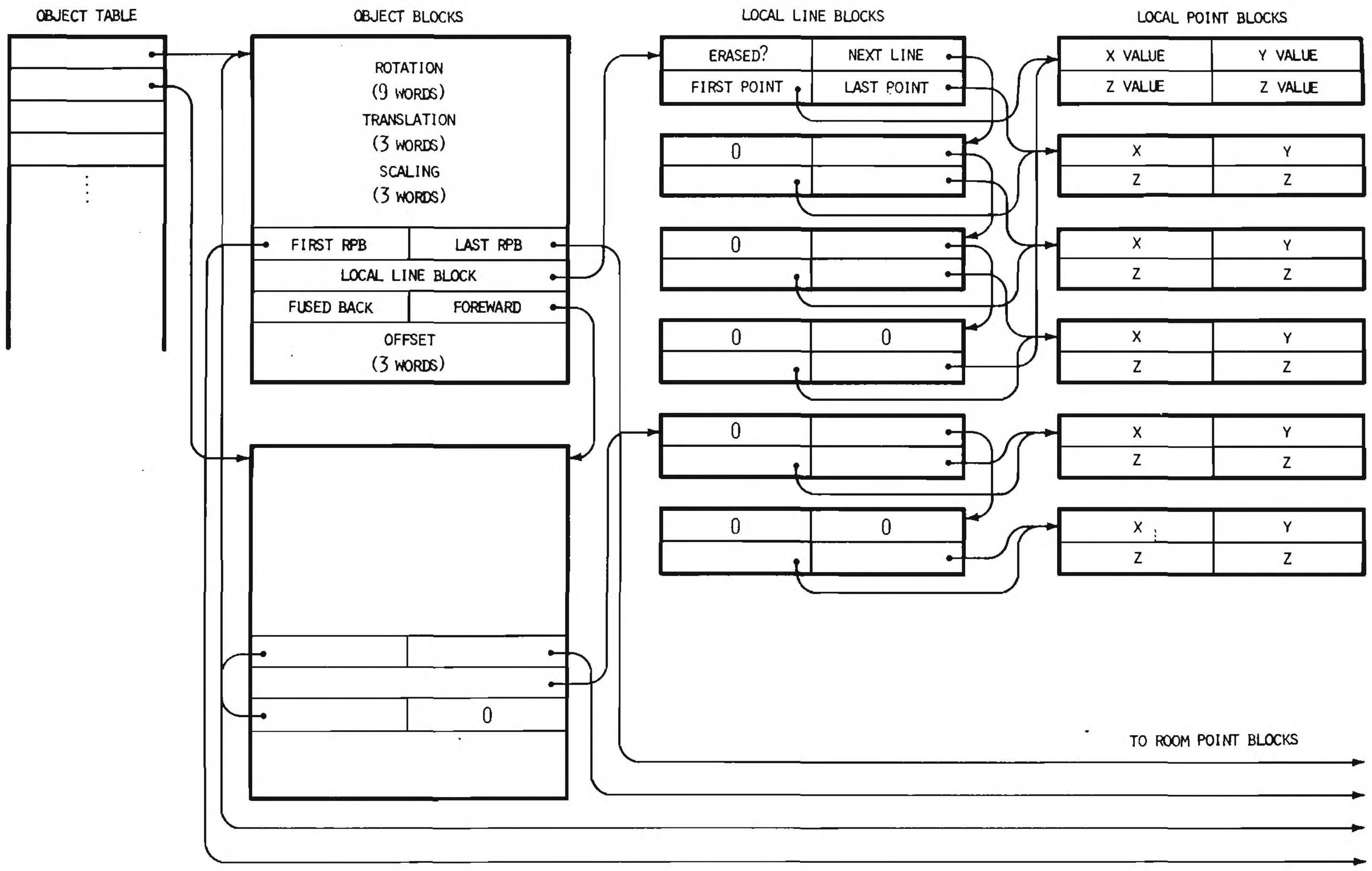


Fig. 21: Detail of local coordinate portion of data structure. The first of the two fused objects is a closed quadrilateral; the second is two separate dots.

Room point blocks contain the many pointers necessary to make the dual-copy data structure fast enough for real time applications. Referring again to Figure 22, the DF pointer points to the actual X, Y, and Z data in the display file. The PIT pointer contains the hash code of the point and therefore is a back pointer to the point index table. This pointer is used to get to the top of a chain of points having the same hash code. All points having the same value are linked, as are all those having the same hash code. A convenient way to visualize this is to think of a chain of "same hash" blocks, each having the possibility of being the first element of a "same value" chain. Other pointers link all points within a given object and all points currently being ELONGATEd. The ELONGATE pointers, of course, exist only during the act of elongating and are otherwise empty. An object number identifies the object to which the point belongs. Two final pointers, the LLB and LPB pointers, link a point in room coordinates with its local line block and local point block counterparts. These also serve to identify a specific point during the "touch" operation and to guarantee that a point transformed from room to local coordinates will map to itself when the transformation from local to room coordinates is performed.

The instruction file contains information used by the display file processor in displaying the line segments defined in the display file. Instruction file commands come in pairs, the first telling how many line segments to draw and the second telling where data for the line segment series begins. Thus, it is possible to display two line segments with one end point in common by referencing only three consecutive points in the display file and one instruction pair in

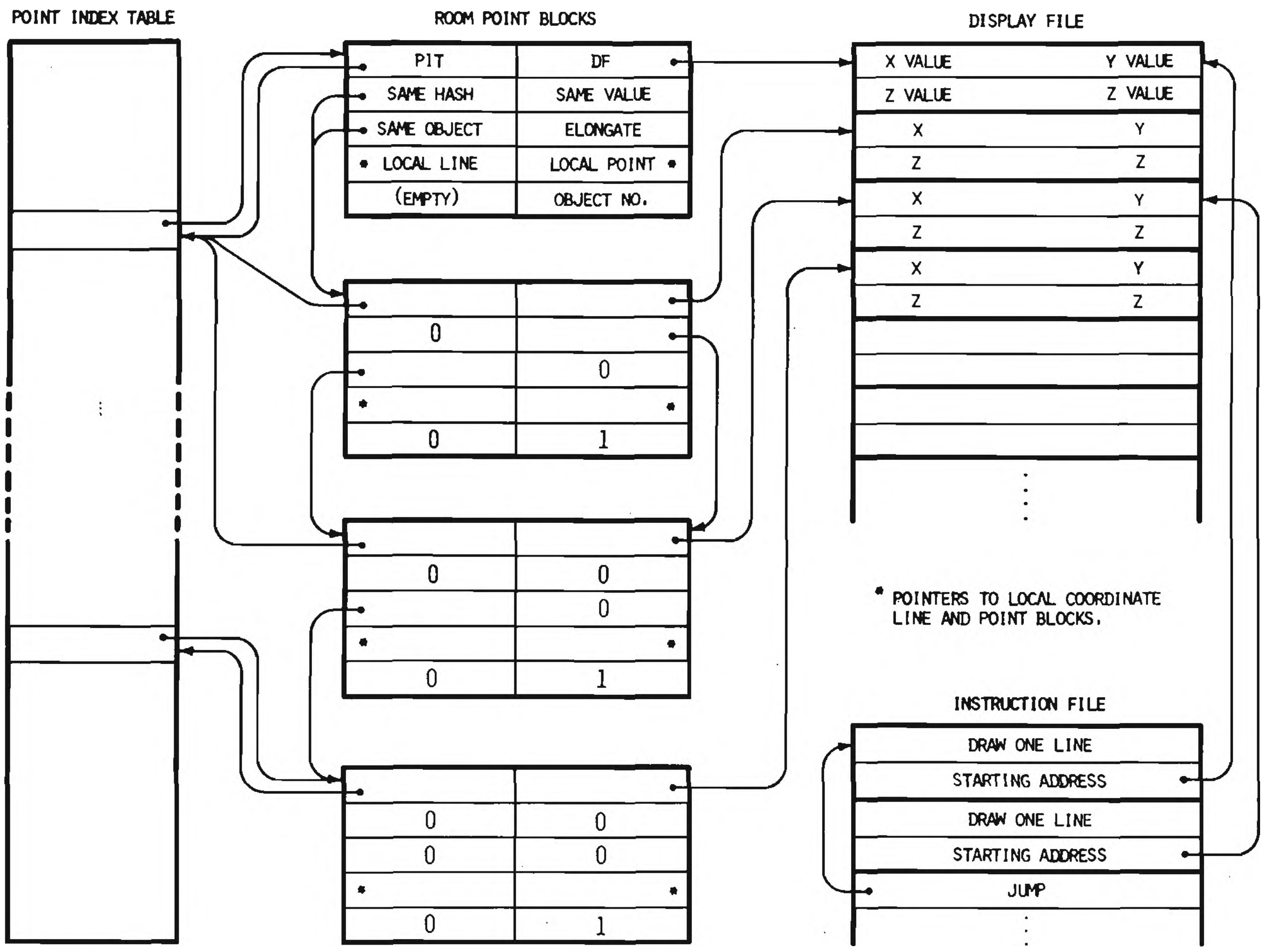


Fig. 22: Detail of room coordinate portion of data structure. The pointers show a possible configuration for displaying two connected line segments, both belonging to object number one.

the instruction file. However, for the purposes of Sorcerer's Apprentice, each instruction file pair will allow only one line segment to be drawn, causing the display file processor to reference four display file points and two instruction file pairs for the two line segments just mentioned. This less efficient method of displaying was chosen because it makes trivial the action of the ERASE command. Erasing is done by having the first of the ERASEd instruction pair "jump" control to the next instruction pair and by marking the "ERASEd?" box of the appropriate local point block (Fig. 21).

Storage requirements for the dual-copy data structure are high. In addition to the number of 36 bit words for the object table, the object blocks, and the point index table, each line segment requires from 18 to 24 words. Also, since the object table, display file, and instruction file data need to be consecutively arranged, these three files are predefined and not taken from a free storage structure as are the various object, line, and point blocks. However, with the high-speed hardware not functioning, the computing system has far more memory than display speed. If memory size was a problem, a "garbage collecting" scheme could be added to make use of the space now made available by ERASEing a line.

ACKNOWLEDGMENTS

Public credit should be given to the many associated with the Sorcerer's Apprentice project. Those most involved were Chris Cutler and Marlin Shelley whose ideas and assistance in both software and hardware have been invaluable. Mike Archuleta wrote the clipping divider simulator, Mike Mladejovsky designed the "puppet" tracking system, and Mike Milochik helped with the photographs. Jim Callan of Evans and Sutherland Computer Corporation provided the data for the USA map.

Sincere thanks goes to Dr. Ivan Sutherland who helped formulate the Sorcerer's Apprentice concept and provided encouragement and guidance along the way.