Final Report

HARDWARE FOR A THREE-DIMENSIONAL DISPLAY

Contract XG-2972

Between the United States Government

and

The President and Fellows of

Harvard University

Submitted to

Norman Prince

Contract Officer

Associate Professor Ivan. E. Sutherland

Principal Investigator

August 1968

Final Report

HARDWARE FOR A THREE-DIMENSIONAL DISPLAY

Harvard University

Associate Professor Ivan E. Sutherland

Principal Investigator

August 1968

INDEX

INTRODUCTION

This report on hardware for a three-dimensional display is composed of three classes of material: tutorial papers, student theses and logic diagrams. These items have been assembled to provide the necessary knowledge to understand the digital hardware designed, assembled, and checked out by Harvard personnel from the summer of 1967 to the end of August, 1968.

Part I is this introduction which explains the makeup of the report.

Part II, A Head Mounted Three-Dimensional Display, is a general introduction and first results from the total operating system that has been constructed. Part II should be understood before attempting the rest of the report. The other sections of the report describe pieces of the system involved in the overall project.

Part III, An Ultrasonic Head Position Sensor, is a student thesis. It explains the results obtained so far with an alternate method of determining location and orientation of physical objects in space. While further work, particularly on the redundant mathematics involved, is needed to make this method attractive in general, limited success with this method by itself is reported here. More work is contemplated in order to be able to use this method in the total system. As noted in Part II, the current system uses a mechanical head position sensor.

Part IV, A Clipping Divider is a paper prepared for presentation. It is a general introduction to the design and operation of the clipper hardware. This is the device that removes from the screen of a computer driven scope the lines that are out of the user's field of view. Significant accomplishments have been to remove these lines dynamically (in real time) in significant quantity to allow relatively complex pictures to be viewed with the aid of the three-dimensional system.

The appendices contain detailed supporting information. First, there is the original thesis describing the clipping divider. Then there are actual logic diagrams that describe the matrix multiplier and the clipper.

No programming for the total system is presented. The system as described in Part II was operating from a PDP-1 at Harvard for only a short period in August 1968. The system will again be assembled at the University of Utah in the fall of 1968 using a PDP-9 for a local, fast-core processor, and a Univac 1108 for backup systems and storage.

Part II: A Head-Mounted Three-Dimensional Display

The work reported here was performed at Harvard University by Associate Professor Ivan E. Sutherland as Principal Investigator. Agency support was used to purchase hardware of the clipping divider, the matrix multiplier, and their several interfaces with the other units. Additional support was obtained from the Advanced Research Projects Agency (ARPA) of the Department of Defense under contract SD 265, from the Office of Naval Research under contract ONR 1866(16), and from a long standing agreement between Bell Telephone Laboratories and the Harvard Computation Laboratory. The early work at the MIT Lincoln Laboratory was also supported by ARPA.

The material in Part II has been accepted for Professor Sutherland's presentation at the Fall Joint Computer Conference.

Final Report Aug. 1968

ABSTRACT

This paper describes a display which gives its user the illusion that the objects shown are three-dimensional and surround him in space. The user wears special spectacles which contain miniature cathode ray tubes. The position and orientation of the user's head are measured. Special-purpose high-speed digital hardware generates a perspective view which changes as the user moves his head exactly as would the view of a real three-dimensional object. The digital hardware can perform calculations fast enough to recompute and redisplay up to 3000 lines thirty times per second. Different perspective images may be presented to each eye for stereo viewing.

INTRODUCTION

The fundamental idea behind the three-dimensional display is to present the user with a perspective image which changes as he moves. The retinal image of the real objects which we see is, after all, only two-dimensional. Thus if we can place suitable two-dimensional images on the observer's retinas, we can create the illusion that he is seeing a three-dimensional object. Although stereo presentation is important to the three-dimensional illusion, it is less important than the change that takes place in the image when the observer moves his head. The image presented by the three-dimensional display must change in exactly the way that the image of a real object would change for similar motions of the user's head. Psychologists have long known that moving perspective images appear strikingly three-dimenisonal even without stereo presentation; the three-dimensional display described in this paper depends heavily on this "kinetic depth effect" (1).

In this project we are not making any effort to measure rotation of the eyeball. Because it is very difficult to measure eye rotation, we are fortunate that the perspective picture presented need not be changed as the user moves his eyes to concentrate on whatever part of the picture he chooses. The perspective picture presented need only be changed when he moves his head. In fact, we measure only the position and orientation of the optical system fastened to the user's head. Because the optical system determines the virtual screen position and the user's point of view, the position and orientation of the optical system define which perspective view is appropriate.

Our objective in this project has been to surround the user with displayed three-dimensional information. Because we use a homogeneous coordinate representation (2,3), we can display objects which appear to be close to the user or which appear to be infinitely far away. We can display objects beside the user or behind him which will become visible to him if he turns around. The user is able to move his head three feet off axis in any direction to get a better view of nearby objects. He can turn completely around and can tilt his head up or down thirty or forty degrees. The objects displayed appear to hang in the space all around the user.

The desire to surround a user with information has forced us to solve the "windowing" problem. The "clipping divider" hardware we have built eliminates those portions of lines behind the observer or outside of his field of view. It also performs the division necessary to obtain a true perspective view. The clipping divider can perform the clipping computations for any line in about 10 microseconds, or about as fast as a modern high-performance display can paint lines on a CRT. The clipping divider is described in detail in a separate paper (4) in this issue. Because the clipping divider permits dynamic perspective display of three-dimensional drawings and arbitrary magnification of two-dimensional drawings, we feel that it is the most significant result of this research to date.

In order to make truly realistic pictures of solid three-dimensional objects, it is necessary to solve the "hidden line problem". Although it is easy to compute the perspective positions of all parts of a complex object, it is difficult to compute which portions of one object are hidden by another object. Of the software solutions now available, (2,5-10) only the MAGI (9) and the Warnock (10) approaches seem to have potential as eventual real-time solutions for reasonably complex situations; the time required by the other methods appears to grow with the square of situation complexity. The only existing real-time solution to the hidden line problem is a very expensive special-purpose computer at NASA Houston (11) which can display only relatively simple objects. We have concluded that showing "opaque" objects with hidden lines removed is beyond our present capability. The three-dimensional objects shown by our equipment are transparent "wire frame" line drawings.

OPERATION OF THE DISPLAY SYSTEM

In order to present changing perspective images to the user as he moves his head, we have assembled a wide variety of equipment shown in the diagram of Figure 1. Special spectacles containing two miniature cathode ray tubes are attached to the user's head. A fast, two-dimensional, analog line generator provides deflection signals to the miniature cathode ray tubes through transistorized deflection amplifiers. Either of two head position sensors, one mechanical and the other ultrasonic, is used to measure the position of the user's head.

As the observer moves his head, his point of view moves and rotates with respect to the room coordinate system. In order to convert from room coordinates to a coordinate system based on his point of view, a translation and a rotation are required. A computer uses the measured head position information to compute the elements of a rotation and translation matrix appropriate to each particular viewing position. Rather than changing the information in the computer memory as the user moves his head, we transform information from room coordinates to eye coordinates dynamically as it is displayed. A new rotation and translation matrix is loaded into the digital matrix multiplier once at the start of each picture repetition. As a part of the display process the endpoints of lines in the room coordinate system are fetched from memory and are individually transformed to the eye coordinate system by the matrix multiplier. These translated and rotated endpoints are passed via an intermediate buffer to the digital clipping divider. The clipping divider eliminates any information outside the user's field of view and computes the appropriate perspective image for the remaining data. The final outputs of the clipping divider are endpoints of two-dimensional lines specified in scope coordinates. The two-dimensional line specifications are passed to a buffered display interface which drives the analog line-drawing display.

We built the special-purpose digital matrix multiplier and clipping divider to compute the appropriate perspective image dynamically because no available general-purpose computer is fast enough to provide a flicker-free dynamic picture. Our equipment can provide for display of 3000 lines at 30 frames per second, which amounts to a little over 10 microseconds per line. Sequences of vectors which form "chains" in which the start of one vector is the same as the end of the previous one can be processed somewhat more efficiently than isolated lines. Assuming, however, two endpoints for every line, the matrix multiplier must provide coordinate transformation in about 5 microseconds per endpoint. Each matrix multiplication requires 16 accumulating multiplications; and therefore a throughput of about 3,000,000 multiplications per second. The clipping divider, which is separate and asynchronous, operates at about the same speed, processing two endpoints in slightly over 10 microseconds. Unlike the fixed time required for a matrix multiplication, however, the processing time required by the clipping divider depends on the data being processed. The time required by the analog line generator depends on the length of the line being drawn, the shortest requiring about 3 microseconds, the longest requiring about 36 microseconds and an average of about 10 microseconds.

The matrix multiplier, clipping divider, and line-generator are connected in a "pipe-line" arrangement. Data "streams" through the system in a carefully interlocked way. Each unit is an independently timed digital device which provides for its own input and output synchronization. Each unit examines an input flag which signals the arrival of data for it. This data is always held until the unit is ready to accept it. As the unit accepts a datum, it also reads a "directive" which tells it what to do with the datum. When the unit has accepted a datum, it clears its input flag. When it has completed its operation, it presents the answer on output lines and sets an output flag to signal that data is ready. In some cases the unit will commence the next task before its output datum has been taken. If so, it will pause in the new computation if it would have to destroy its output datum in order to proceed. Orderly flow of information through the system is ensured because the output flag of each unit serves as the input flag of the next. The average rate of the full system is approximately the average rate of the slowest unit. Which unit is slowest depends on the data being processed. The design average rate is about 10 microseconds per line.

The computer in this system is used only to process the head-position sensor information once per frame, and to contain and manipulate the three-dimensional drawing. No available general-purpose computer would be fast enough to become intimately involved in the perspective computations required for dynamic perspective display. A display channel processor serves to fetch from memory the drawing data required to recompute and refresh the CRT picture. The channel processor can be "configured" in many ways so that it is also possible to use the matrix multiplier and clipping divider independently.

For example, the matrix multiplier can be used in a direct memory-to-memory mode which adds appreciably to the arithmetic capability of the computer to which it is attached. For two-dimensional presentations it is also possible to bypass the matrix multiplier and provide direct input to the clipping divider and display. These facilities were essential for debugging the various units independently.

PRESENTING IMAGES TO THE USER

The special headset which the user of the three-dimensional display wears is shown in Figure 2. The optical system in this headset magnifies the pictures on each of two tiny cathode ray tubes to present a virtual image about eighteen inches in front of each of the user's eyes. Each virtual image is roughly the size of a conventional CRT display. The user has a 40 degree field of view of the synthetic information displayed on the miniature cathode ray tubes. Half-silvered mirrors in the prisms through which the user looks allow him to see both the images from the cathode ray tubes and objects in the room simultaneously. Thus displayed material can be made either to hang disembodied in space or to coincide with maps, desk tops, walls, or the keys of a typewriter.

The miniature cathode ray tubes mounted on the optical system form a picture about one half of an inch square. Because they have a nominal six tenths mil spot size, the resolution of the virtual image seen by the user is about equivalent to that available in standard large-tube displays. Each cathode ray tube is mounted in a metal can which is carefully grounded to protect the user from shorts in the high voltage system. Additional protection is provided by enclosing the high voltage wiring in a grounded shield.

The miniature cathode ray tubes have proven easy to drive. They use electrostatic deflection and focussing. Because their deflection plates require signals on the order of only 300 volts, the transistorized deflection amplifiers are of a relatively straightforward design. Complementary-symmetry emitter followers are used to drive four small coaxial cables from the amplifier to each cathode ray tube. Deflection and intensification signals for the miniature cathode ray tubes are derived from a commercial analog line-drawing display which can draw long lines in 36 microseconds (nominal) and short lines as fast as three microseconds (nominal).

The analog line generator accepts picture information in the coordinate system of the miniature cathode ray tubes. It is given two-dimensional scope coordinates for the endpoints of each line segment to be shown. It connects these endpoints with smooth, straight lines on the two-dimensional scope face. Thus the analog line-drawing display, transistorized deflection amplifiers, miniature cathode ray tubes, and head-mounted optical system together provide the ability to present the user with any two-dimensional line drawing.

HEAD POSITION SENSOR

The job of the head position sensor is to measure and report to the computer the position and orientation of the user's head. The head position sensor should provide the user reasonable freedom of motion. Eventually we would like to allow the user to walk freely about the room, but our initial equipment allows a working volume of head motion about six feet in diameter and three feet high. The user may move freely within this volume, may turn himself completely about, and may tilt his head up or down aproximately forty degrees. Beyond these limits, head position cannot be measured by the sensor. We suspect that it will be possible to extend the user's field of motion simply by transporting the upper part of the head position sensor on a ceiling trolley driven by servo or stepping motors. Since the position of the head with respect to the sensor is known, it would be fairly easy to keep the sensor approximately centered over the head.

The head position measurement should be made with good resolution. Our target is a resolution of 1/100 of an inch and one part in 10,000 of rotation. Resolution finer than that is not useful because the digital-to-analog conversion in the display system itself results in a digital "grain" of about that size.

The accuracy requirement of the head position sensor is harder to determine. Because the miniature cathode ray tubes and the head-mounted optical system together have a pin-cushion distortion of about three percent, information displayed to the user may appear to be as much as three tenths of an inch out of place. Our head position sensor, then, should have an accuracy on the order of one tenth of an inch, although useful performance may be obtained even with less accurate head-position information.

We have tried two methods of sensing head position. The first of these involves a mechanical arm hanging from the ceiling as shown in Figure 3. This arm is free to rotate about a vertical pivot in its ceiling mount. It has two universal joints, one at the top and one at the bottom, and a sliding center section to provide the six motions required to measure both translation and rotation. The position of each joint is measured and presented to the computer by a digital shaft position encoder.

The mechanical head position sensor is rather heavy and uncomfortable to use. The information derived from it, however, is easily converted into the form needed to generate the perspective transformation. We built it to have a sure method of measuring head position.

We have also constructed a continuous wave ultrasonic head position sensor shown in Figure 4. Three transmitters which transmit ultrasound at 37, 38.6, and 40.2 kHz are attached to the head-mounted optical system. Four receivers are mounted in a square array in the ceiling. Each receiver is connected to an amplifier and three filters as shown in Figure 5, so that phase changes in sound transmitted over twelve paths can be measured. The measured phase shift for each ultrasonic path can be read by the computer as a separate five-bit number. The computer counts major changes in phase to keep track of motions of more than one wavelength.

Unlike the Lincoln Wand (12) which is a pulsed ultrasonic system, our ultrasonic head position sensor is a continuous wave system. We chose to use continuous wave ultrasound rather than pulses because inexpensive narrow-band transducers are available and to avoid confusion from pulsed noise (such as typewriters produce) which had caused difficulty for the Lincoln Wand. The choice of continuous wave ultrasound, however, introduces ambiguity into the measurements. Although the ultrasonic head position sensor makes twelve measurements from which head-position information can be derived, there is a wave length ambiguity in each of the measurements. The measurements are made quite precisely within a wave, but do not tell which wave is being measured. Because the wavelength of sound at 40 kHz in air is about 1/3 of an inch, each of the twelve measurements is ambiguous at 1/3 inch intervals.Because the computer keeps track of complete changes in phase, the ambiguity in the measurements shows up as a constant error in the measured distance. This error can be thought of as the "initialization error" of the system. It is the difference between the computer's original guess of the initial path length and the true initial path length.

We believe that the initialization errors can be resolved by using the geometric redundancy inherent in making twelve measurements. We have gone to considerable effort to write programs for the ultrasonic head position sensor. These programs embody several techniques to resolve the measurement ambiguities. Although we have had some encouraging results, a full report on the ultrasonic head position sensor is not yet possible.

THE PERSPECTIVE TRANSFORMATION

Generating a perspective image of three dimensional information is relatively easy. Let us suppose that the information is represented in a coordinate system based on the observer's eye as shown in Figure 6. If the two-dimensional scope coordinates, X_s and Y_s , are thought of as extending from -1 to +1, simple geometric reasoning will show that the position at which a particular point should be displayed on the screen is related to its position in three-dimensional space by the simple relations:

$$X_s = \frac{x'}{z'} \cot \frac{\alpha}{2}$$

$$Y_s = \frac{y'}{z'} \cot \frac{\alpha}{2}$$

If an orthogonal projection is desired, it can be obtained by making the value of z constant. Because the perspective (or orthogonal) projection of a straight line in three-dimensional space is a straight line, division by the z coordinate need be performed only for the endpoints of the line. The two-dimensional analog line-generating equipment can fill in the center portion of a three-dimensional line by drawing a two-dimensional line. The digital perspective generator computes values only for the endpoint coordinates of a line.

The three-dimensional information to be presented by the three-dimensional display is stored in the computer in a fixed three-dimensional coordinate system. Because this coordinate system is based on the room around the user, we have chosen to call it the "room" coordinate system. The drawing data in the room coordinate system is represented in homogeneous coordinates. This means that each three-dimensional point or end of a three-dimensional line is stored as four separate numbers. The first three correspond to the ordinary X Y and Z coordinates of three-dimensional space. The fourth coordinate, usually called W, is a scale factor which tells how big a value of X Y or Z represents a unit distance. Far distant material may thus easily be represented by making the sacle factor, W, small. Infinitely distant points are represented by setting the scale factor, W, to zero, in which case the first three coordinates represent only the direction to the point. Nearby points are usually represented by setting the scale factor, W, to its largest possible value, in which case the other three coordinates are just the familiar fixed-point representations of X Y and Z.

THE MATRIX MULTIPLIER

We have designed and built a digital matrix multiplier to convert information dynamically from the fixed "room" coordinate system to the moving "eye" coordinate system. The matrix multiplier stores a four-by-four matrix of 18 bit fixed-point numbers. Because the drawing data is represented in homogeneous coordinates, the single four-by-four matrix multiplication provides for both translation and rotation (2). The matrix multiplier accepts the four 18 bit numbers which represent an endpoint, treating them as a four-component vector which it multiplies by the four-by-four matrix. The result is a four-component vector, each component of which is truncated to 20 bits. The matrix multiplier delivers this 80 bit answer to the clipping divider in approximately 5 microseconds. It therefore performs about three million scalar multiplications per second.

The matrix multiplier uses a separate multiplier module for each column. Each module contains an accumulator, a partial product register, storage for the four matrix elements in that column, and the multiplication logic. The entries of a row of the matrix serve simultaneously as four separate multiplicands. An individual component of the incoming vector serves as the common multiplier. The four multiplications for a single row are thus performed simultaneously. For additional speed, the bits of the multiplier are examined four at a time rather than individually to control multiple-input adding arrays.

THE CLIPPING OR WINDOWING TASK

The job of the clipping divider is to accept three-dimensional information in the eye coordinate system and convert it to appropriate two-dimensional endpoints for display. If both ends of the line are visible, the clipping divider needs merely to perform four divisions, one for each two-dimensional coordinate of each end of the line. Enough equipment has been provided in the clipping divider to perform these four divisions simultaneously.

If the endpoints of a line are not within the observer's field of view, the clipping divider must decide whether any portion of the line is within the field of view. If so, it must compute appropriate endpoints for that portion as illustrated in Figure 7. Lines outside the field of view or behind the user must be eliminated. Operation of the clipping divider is described in a separate paper (4) in this issue.

Like the matrix multiplier, the clipping divider is an independently-timed digital device which provides for its own input and output synchronization. It has an input and an output flag which provide for orderly flow of information through the clipping divider. If a line lies entirely outside the field of view, the clipping divider will accept a new input without ever raising its output flag. Thus only the visible portions of lines that are all or partly visible get through the clipping divider.

RESULTS

I did some preliminary three-dimensional display experiments during late 1966 and early 1967 at the MIT Lincoln Laboratory. We had a relatively crude optical system which presented information to only one of the observer's eyes. The ultrasonic head position sensor operated well enough to measure head position for a few minutes before cumulative errors were objectionable. The coordinate transformations and perspective computations were performed by software in the TX-2. The clipping operation was not provided: if any portion of a line was off the screen, the entire line disappeared.

Even with this relatively crude system, the three dimensional illusion was real. Users naturally moved to positions appropriate for the particular views they desired. For instance, the "size" of a displayed cube could be measured by noting how far the observer must move to line himself up with the left face or the right face of the cube.

Two peculiar and as yet unexplained phenomena occurred in the preliminary experiment. First, because the displayed information consisted of transparent "wire-frame" images, ambiguous interpretations were still possible. In one picture a small cube was placed above a larger one giving the appearance of a chimney on a house. From viewpoints below the roof where the "chimney" was seen from inside, some concentration was required to remember that the chimney was in fact further away than the building. Experience with physical objects insisted that if it was to be seen, the chimney must be in front.

A second peculiar phenominon occured during the display of the bond structure of cyclo-hexane as shown in Figure

8. Observers not familiar with the rippling hexagonal shape of this molecule misinterpreted its shape. Because their view of the object was limited to certain directions, they could not get the top view of the molecule, the view in which the hexagonal shape is most clearly presented. Observers familiar with molecular shapes, however, recognized the object as cyclo-hexane.

In more recent experiments with the improved optical system and vastly improved computation capability, two kinds of objects have been displayed. In one test, a "room" surrounding the user is displayed. The room is shown in Figure 9 as it would look from outside. The room has four walls marked N, S, E, and W, a ceiling marked C and a floor marked F. An observer fairly quickly accomodates to the idea of being inside the displayed room and can view whatever portion of the room he wishes by turning his head. In another test a small cube was displayed in the center of the user's operating area. The user can examine it from whatever side he desires.

The biggest surprise we have had to date is the favorable response of users to good stereo. the two-tube optical system presents indepent images to each eye. A mechanical adjustment is available to accomodate to the different pupil separations of different users. Software adjustments in our test programs also permit us to adjust the virtual eye separation used for the stereo computations. With these two adjustments it is quite easy to get very good stereo presentations. Observers capable of stereo vision uniformly remark on the realism of the resulting images.

ACKNOWLEDGEMENT

When I started work on the head-mounted display I had no idea how much effort would be involved. The project would have died many times but for the spirit of the many people who have become involved. The ultrasonic head-position sensor was designed and built at the MIT Lincoln Laboratory by Charles Seitz and Stylianos Pezaris and is available for our continued use through the cooperation of Lincoln Group 23. Seitz, as a Harvard employee, later designed the matrix multiplier. Robert Sproull, a most exceptionally capable Harvard Senior, simulated, designed most of, built parts of, and debugged the clipping divider. Two graduate students, Ted Lee and Dan Cohen have been an essential part of the project throughout. Our many arguements about perspective presentation, clipping, hidden-line algorithms, and other

subjects form one of the most exciting educational experiences I have had. Ted Lee's programs to display curved surfaces in stereo have been the basis for many experiments. Cohen's programs to exercise the entire system form the basis of the demonstrations we can make. I would also like to thank Quintin Foster who supervised construction and debugging of the equipment. And finally, Stewart Ogden, so called "project engineer", actually chief administrator, who defended us all from the pressures of paperwork so that something could be accomplished.

REFERENCES

1. Green, Bert F., Jr., "Figure Coherence in the Kinetic Depth Effect". Journal of Experimental Psychology, Vol. 62, No. 3, 272-282 (1961).

2. Roberts, Lawrence G., Machine Perception of Three-Dimensional Solids, MIT Lincoln Laboratory Technical Report No. 315 (May 22, 1963).

3. Roberts, Lawrence G., "Homogeneous Matrix Representation and Manipulation of N-Dimensional Constructs", The Computer Display Review, Adams Associates (May 1965).

4. Sproull, Robert F., and Ivan E. Sutherland, "A Clipping Divider", Proceedings of the Fall Joint Computer Conference, 1968 (this issue).

5. Cohen, Dan, A Program for Drawing Bodies With the Hidden Lines Removed, A term-project for course 6.539, MIT (Fall 1965).

6. Haynes, Herbert T., A Computer Method for Perspective Drawing, Master's Thesis, Texas A+M University (Aug. 1966).

7. Loutrel, Phillippe, A Solution to the "Hidden-Line" Problem for Computer-Drawn Polyhedra, New York University Technical Report 400-167 (Thesis), Bronx, New York (September 1967).

8. Appel, A., "The Notion of Quantitative Invisibility and the Machine Rendering of Solids", Proceedings of 22nd National Conference ACM, ACM Publication, p. 67, Thomson Book Company, Washington, D.C. (1967).

9. Mathematical Spplications Group, Inc. (MAGI), "3-D Simulated Graphics", Datamation, (February 1968).

10. Warnock, John E., A Hidden Line Algorithm for Halftone Picture Representation, University of Utah Technical Report 4-5 (May 1968).

11. Equipment installed at the Manned Space Craft Center at Houston, Texas. The project is under the direction of the General Electric Company Electronics Laboratory under NASA Contract No. NAS 9-3916.

12. Roberts, Lawrence G., The Lincoln Wand, MIT Lincoln Laboratory Report (June 1966).

13. Traub, Alan C., "Stereoscopic Display Using Rapid Varifocal Mirror Oscillations", Applied Optics, Vol. 6, number 6 (June 1967).

14. Vlahos, P., "The Three-Dimensional Display: Its Cues and Techniques", Journal of the Society for Information Display, Vol. 2, Number 6 (Nov./Dec. 1965).

15. Land, Richard, and Ivan Sutherland, "Real Time, Color, Stereo, Computer Displays", to be published in Applied Optics.

FIGURE CAPTIONS

Figure 1: The parts of the three-dimensional display system.

Figure 2: The head-mounted display optics with miniature CRT's.

Figure 3: The mechanical head position sensor in use.

Figure 4: The ultrasonic head position sensor in use.

Figure 5: The ultrasonic head position sensor logic.

Figure 6: The x', y', z' coordinate system based on the observer's eye position.

Figure 7: Clipping and perspective projection in three dimensions.

Figure 8: A computer-displayed perspective view of the cyclo-hexane molecule.

Figure 9: A computer-displayed perspective view of the "room" as seen from outside.

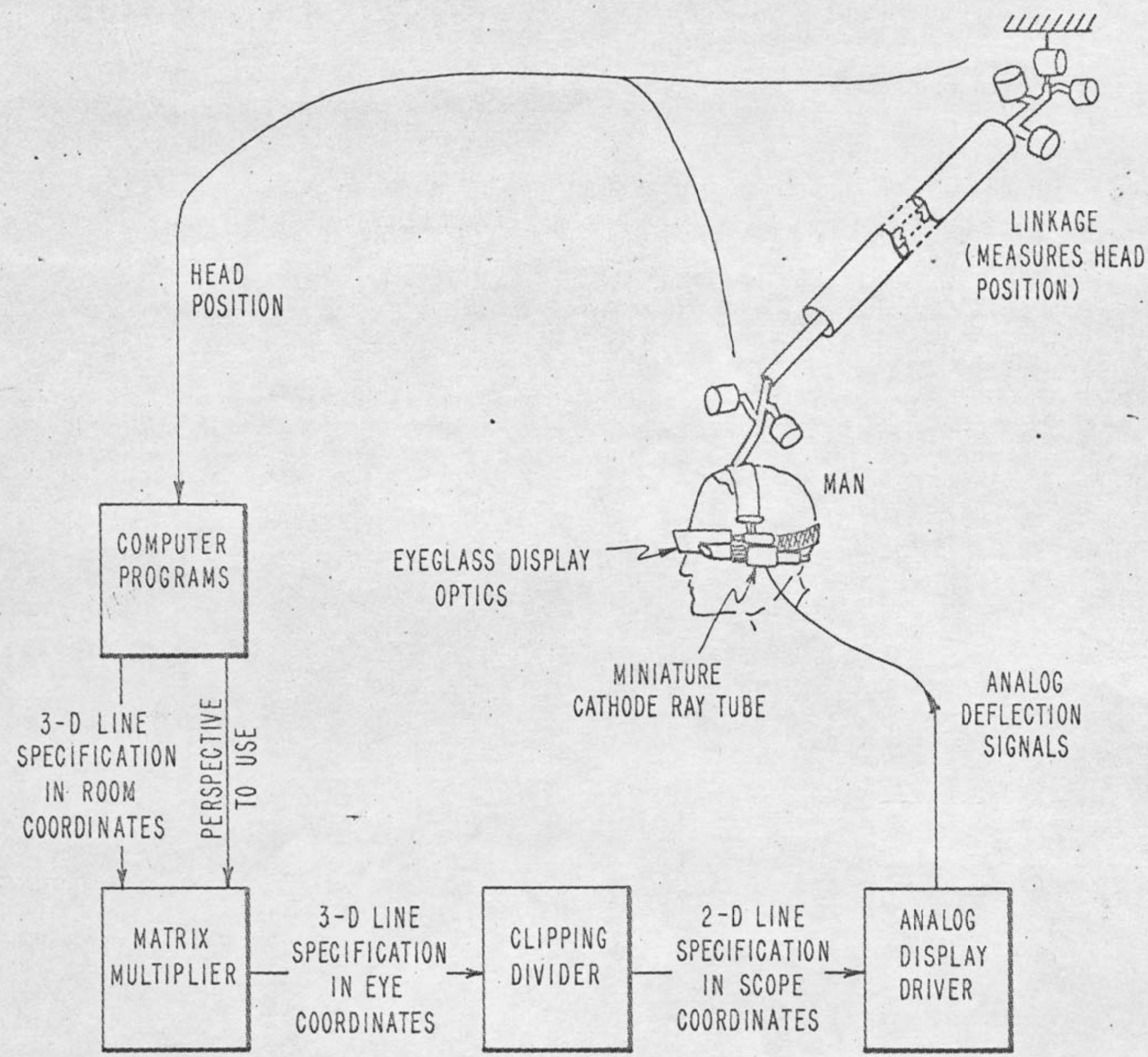


Figure 1: The parts of the three-dimensional display system.

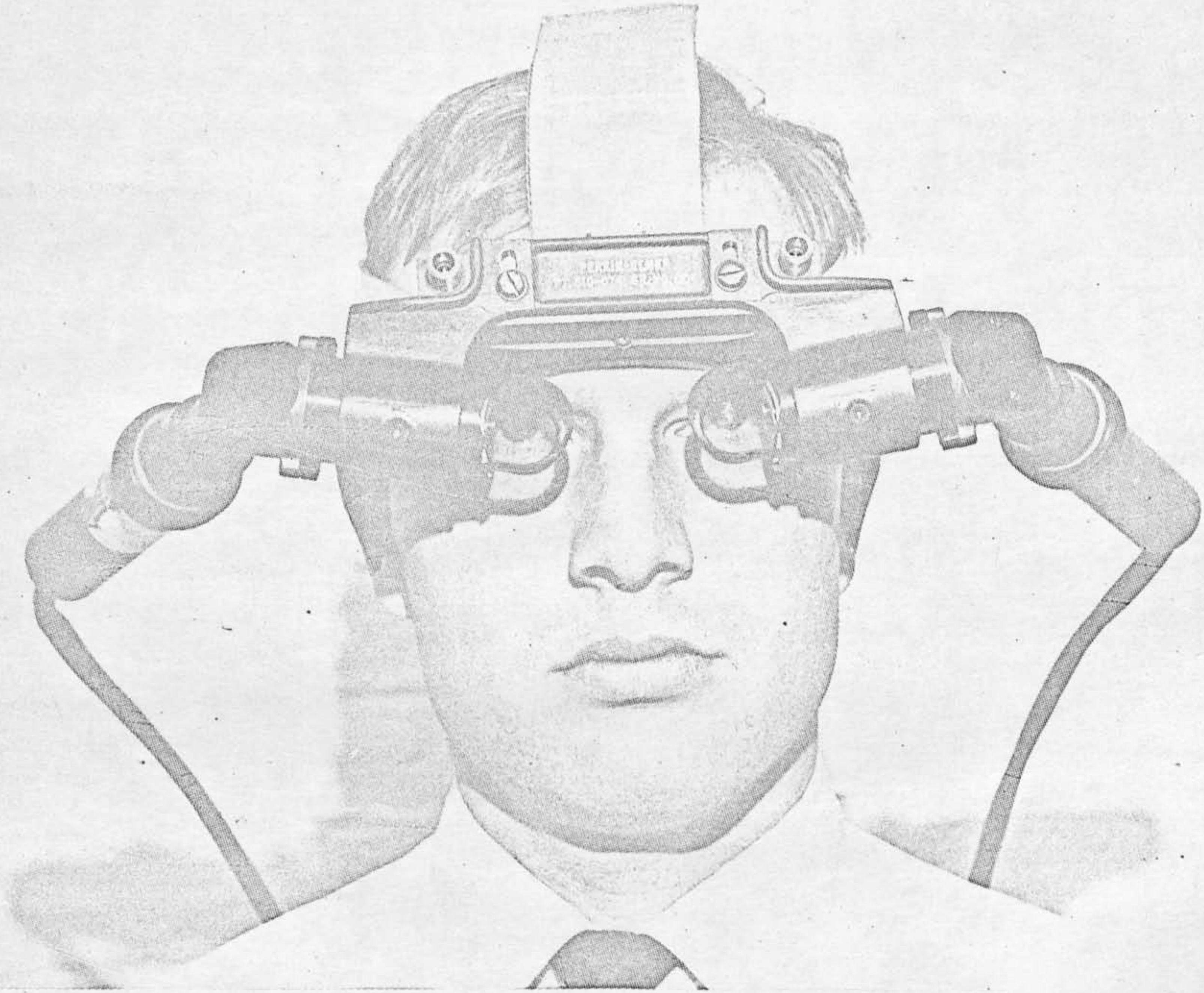


Figure 2: The head-mounted display optics with miniature CRT's.

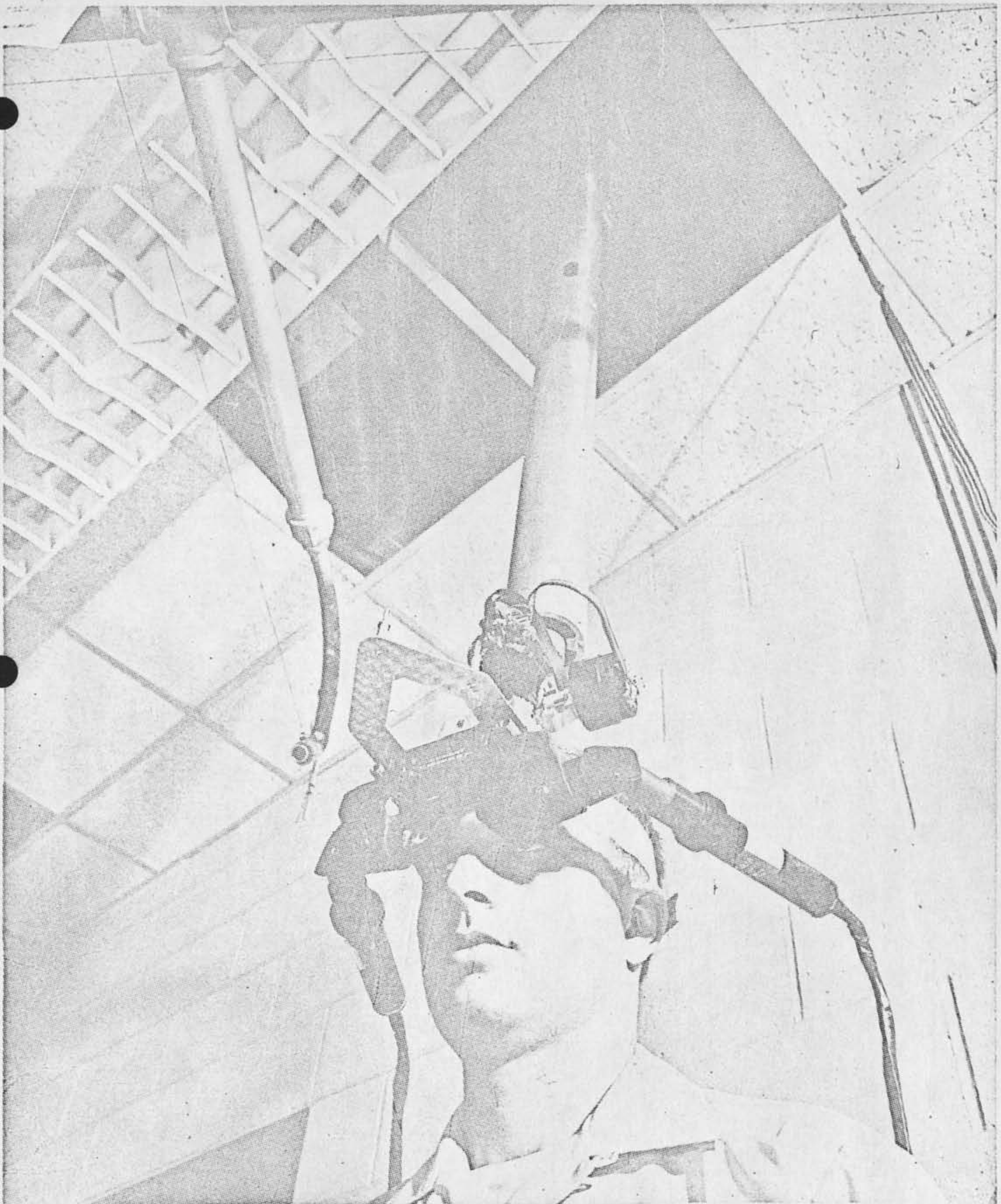


Figure 3: The mechanical head position sensor in use.

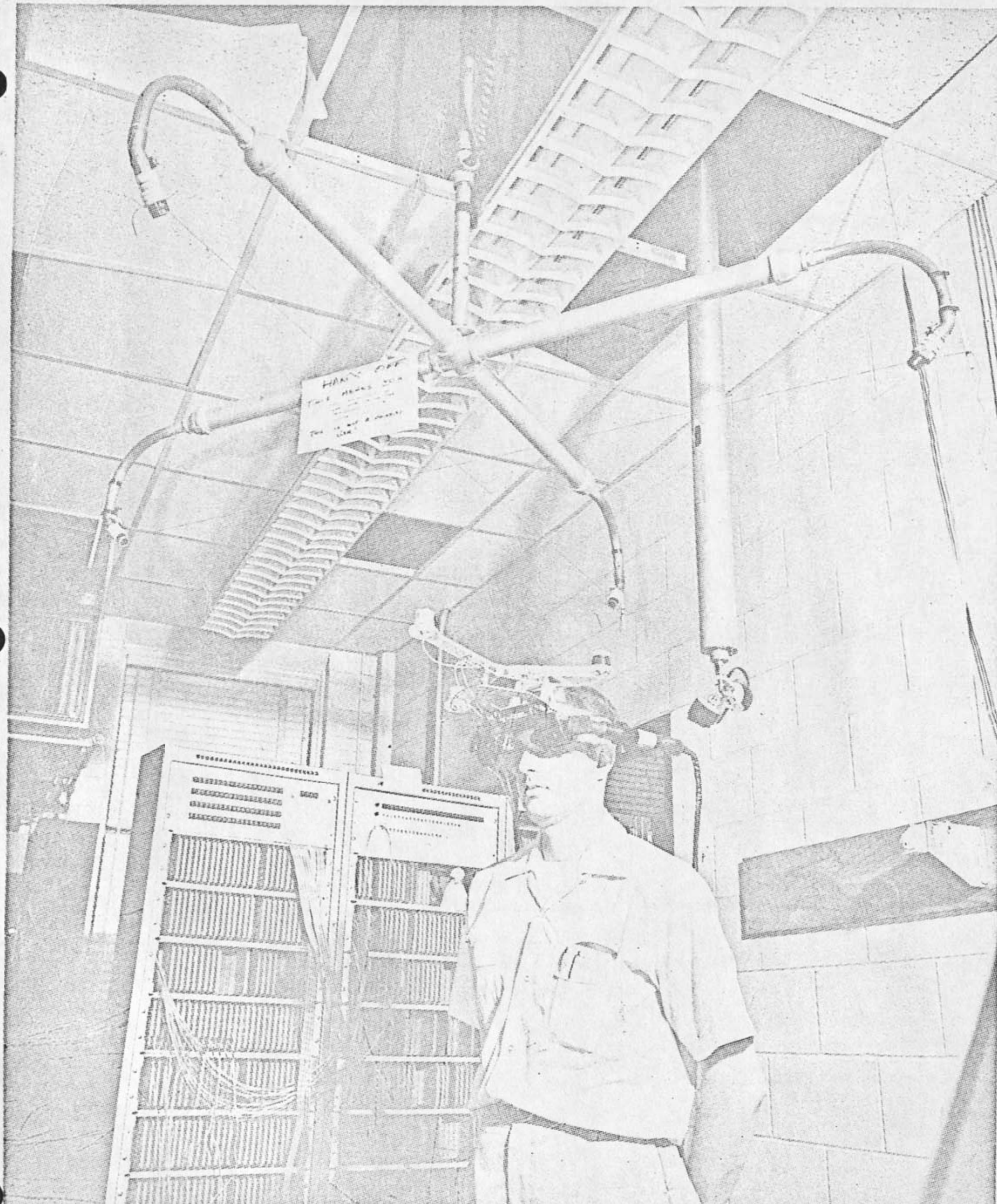


Figure 4: The ultrasonic head position sensor in use.

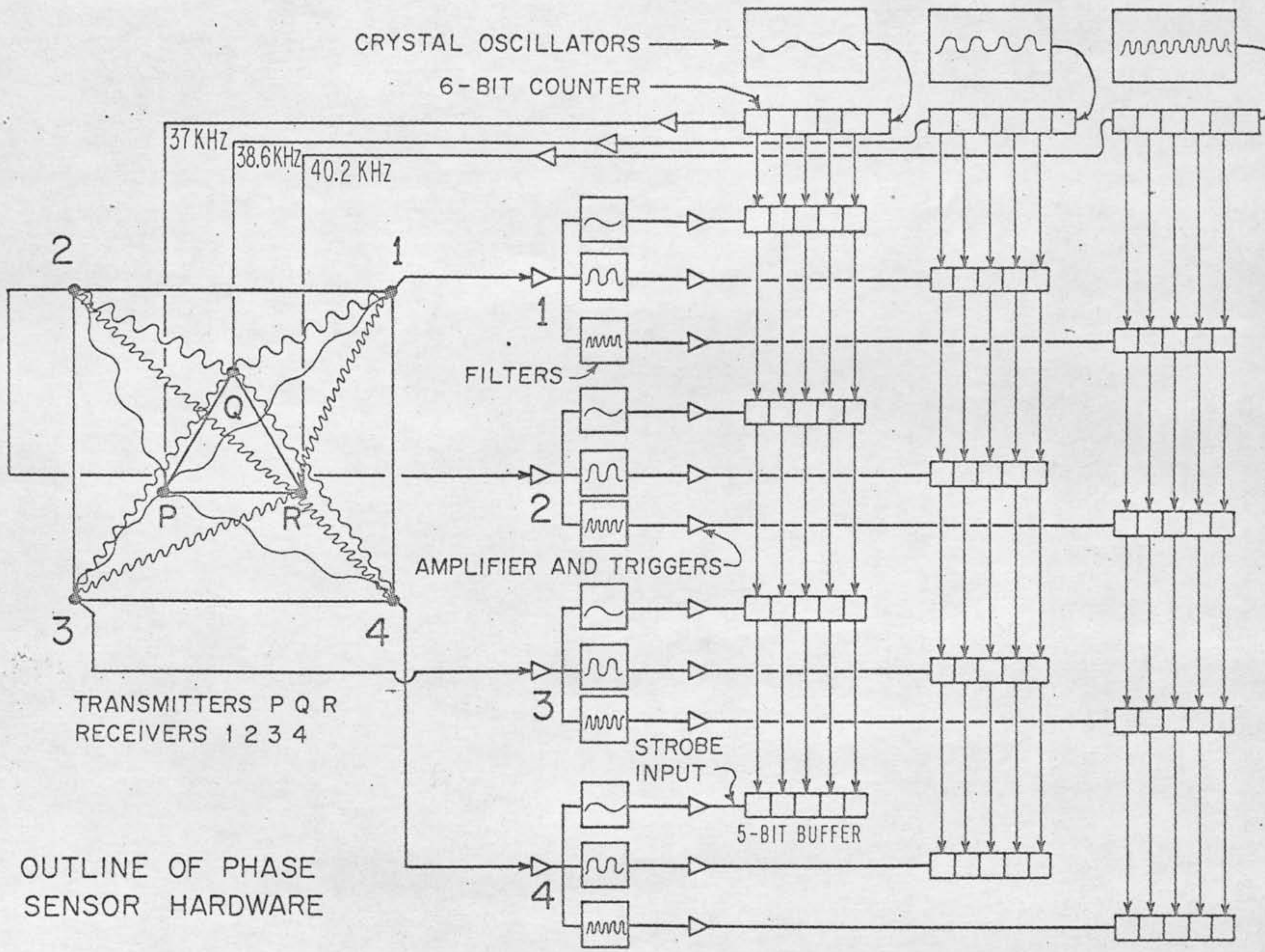


Figure 5: The ultrasonic head position sensor logic.

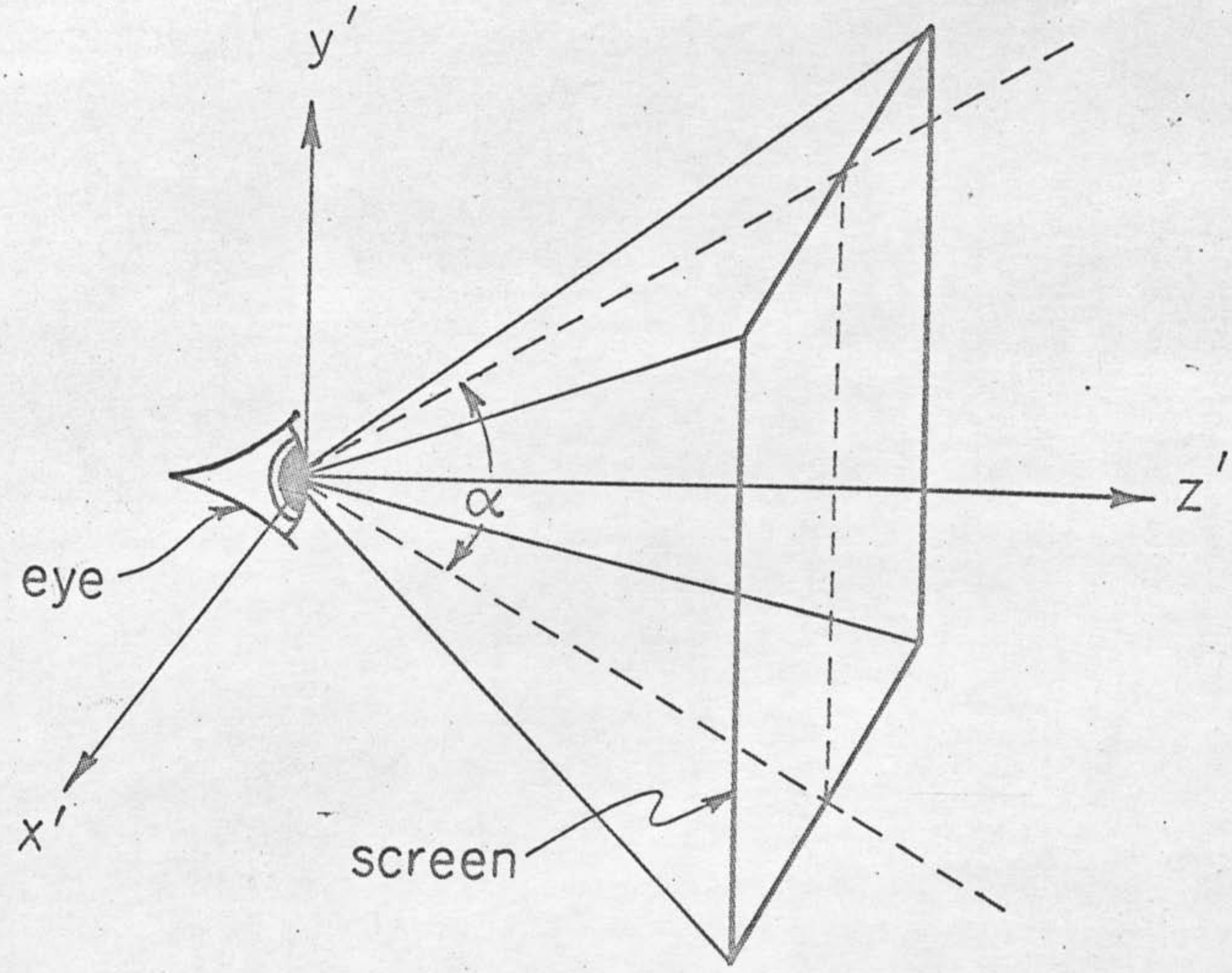
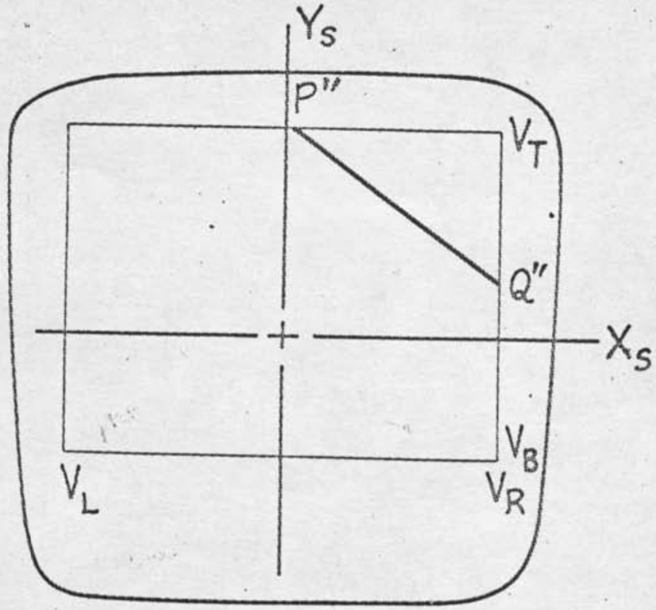


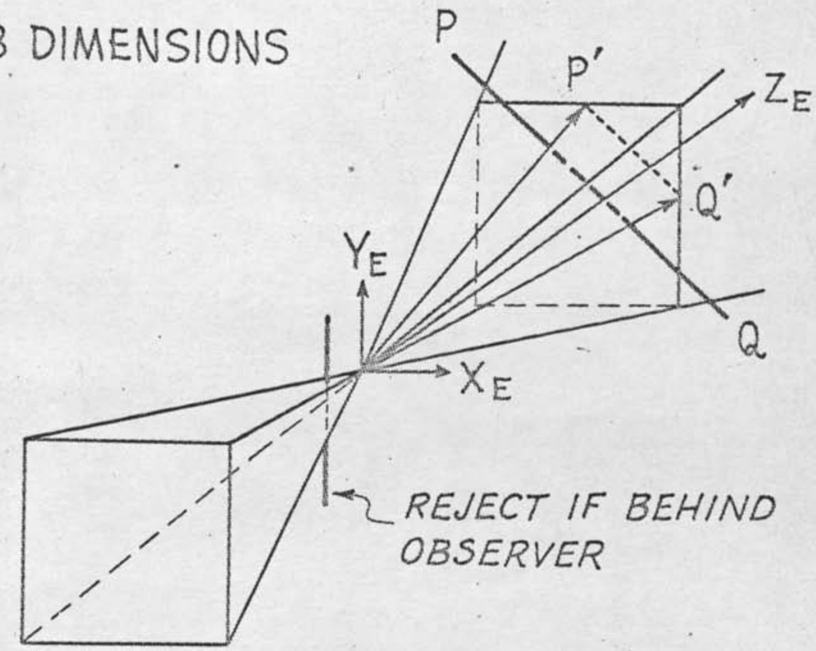
Figure 6: The x', y', z' coordinate system based on the observer's eye position.

Figure 7:

CLIPPING IN 3 DIMENSIONS



SCOPE COORDINATES



EYE COORDINATES

$$X_S = \frac{X_E}{Z_E} VS_x + VC_x$$

$$Y_S = \frac{Y_E}{Z_E} VS_y + VC_y$$

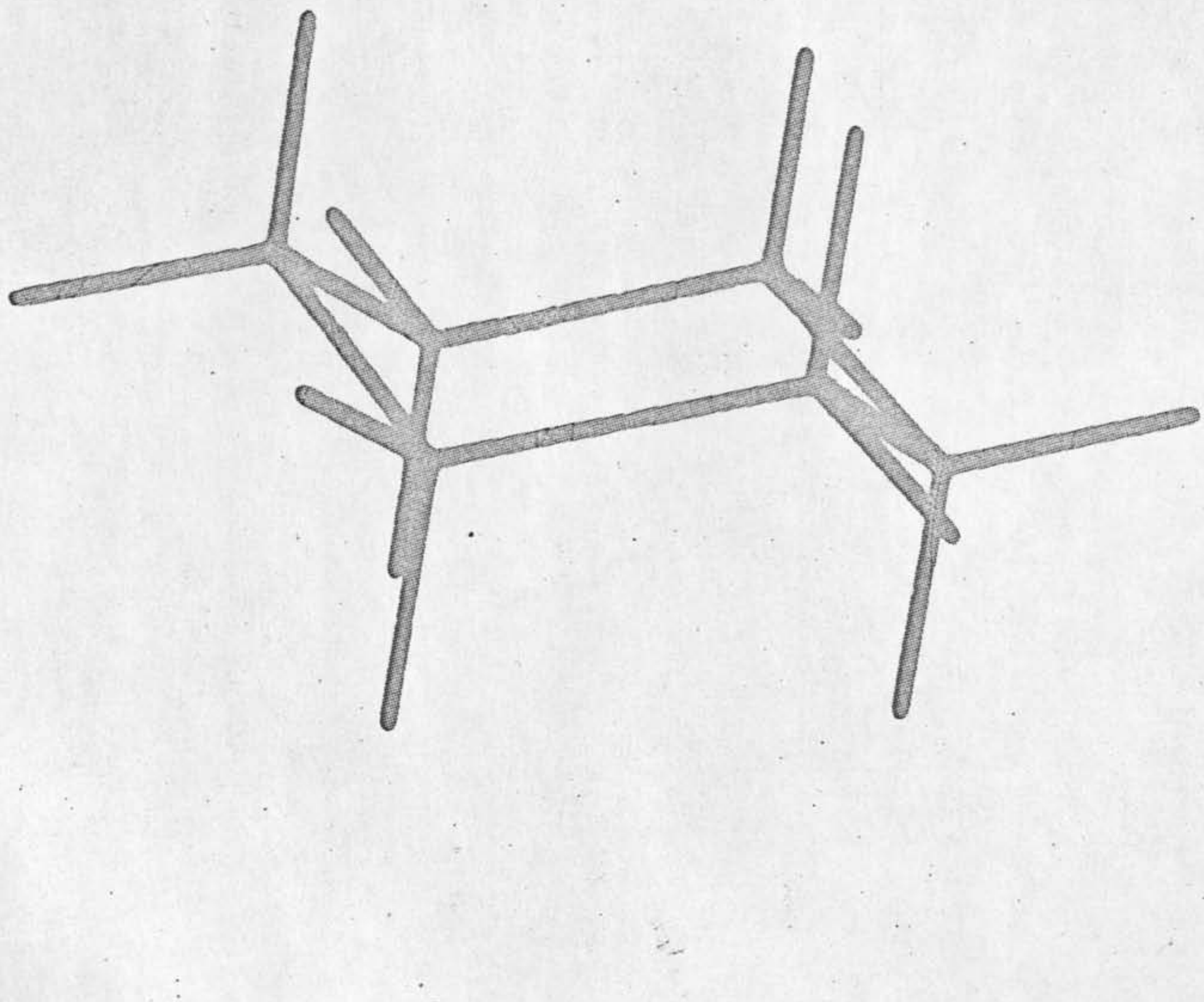


Figure 8: A computer-displayed perspective view of the cyclo-hexane molecule.

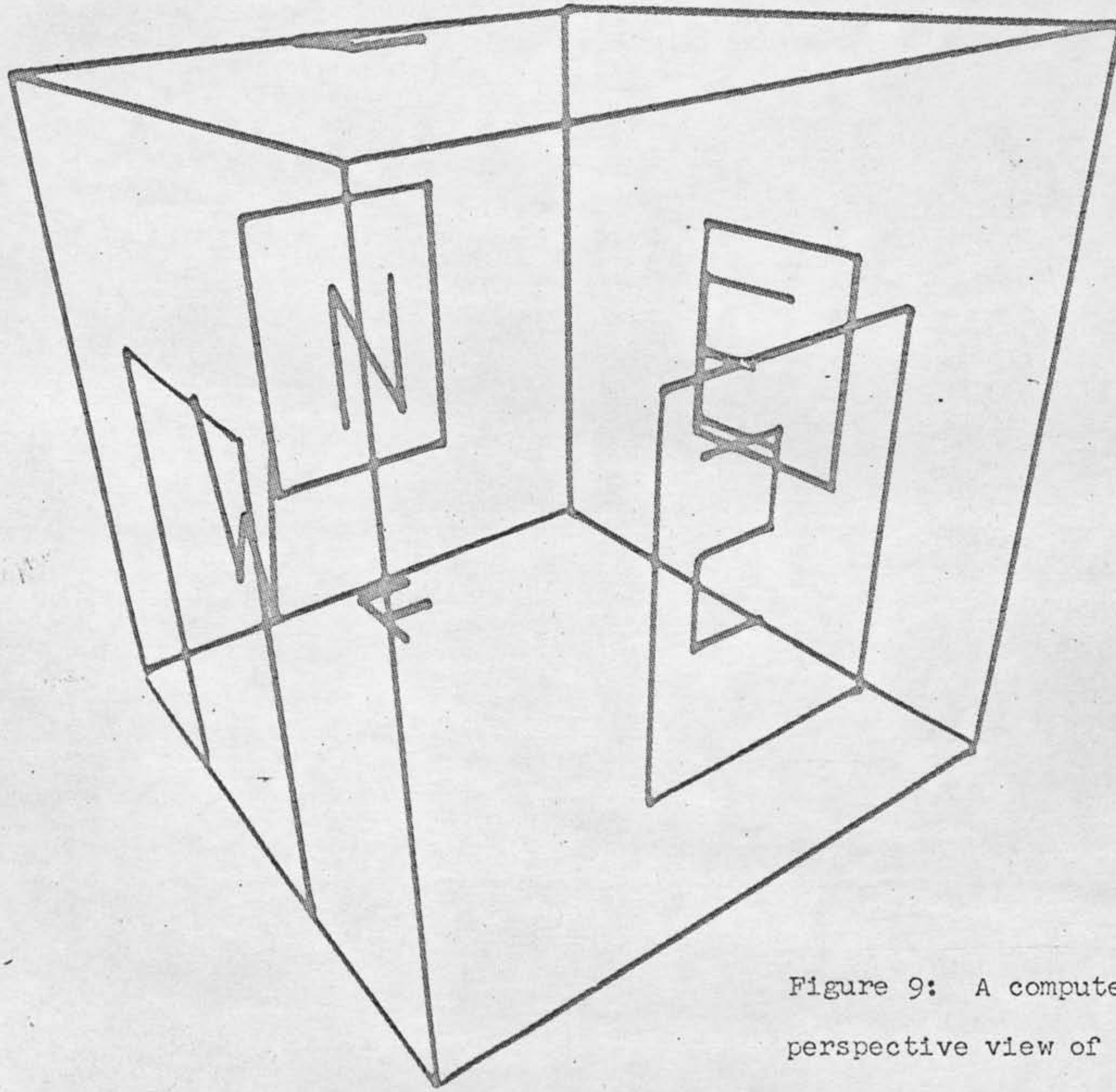


Figure 9: A computer-displayed perspective view of the "room" as seen from outside.

Part III: An Ultrasonic Head Position Sensor

The following is the thesis presented by Bruce Guenther Baumgart to the Committee on Applied Mathematics in partial fulfillment of the honors requirements for the degree of Bachelor of Arts. Hardware for the ultrasonic head position sensor was originally assembled at MIT Lincoln Laboratory under a contract with the Advanced Research Projects Agency. This equipment was used at Harvard under ARPA contract SD 265.

Final Report Aug. 1968

INTRODUCTION

This thesis describes the hardware, mathematical analysis and programming for an ultrasonic head position sensor. The head position sensor is part of a three dimensional display system. The illusion of viewing a three dimensional object will be generated for a single viewer looking through a special headset. The headset worn by the viewer will display a perspective view of the object on two minature CRTs. The particular view of the object displayed will correspond to what would really be seen from where the viewer's head actually is. Special hardware will compute the changing perspective as the observer moves his head. In order to know what perspective to produce, the display computer requires a special input device to supply the position and orientation of the viewer's head within an approximately 6' × 6' × 6' work area with a resolution of one hundredth of an inch.

HARDWARE

The head position is tracked by measuring only the phase of continuous wave ultrasound passing over twelve separate paths from the observer's head to fixed points in the room. Three ultrasonic transmitters are mounted on the viewer's headset, and four ultrasonic receivers are mounted on the ceiling. Sound goes from each transmitter to each receiver, thus there are twelve distances from which the head's position can be calculated. Each distance is spanned by an integral number of ultrasonic waves plus a phase or fraction of a wave. The present hardware arrangement can only report the fractions from which the number of waves and the corresponding distance must be deduced. The main topic of this thesis is how to deduce the distances to the head's position from the ultrasonic phase measurements.

The ultrasonic hardware was designed by I. E. Sutherland, in the summer of 1966, at Lincoln Laboratory, the digital equipment was built by Charles Seitz, the analog by Stellios Pezaris. The equipment was moved to Harvard in January 1968.

HARDWARE DETAILS

The three ultrasonic transmitters on the headset form an equilateral triangle of about 12 inches on a side. The four ultrasonic receivers on the ceiling form a square of about 46 inches on a side. Both the transmitters and receivers are barium titanate transducers manufactured by Fluid Data Corp. under the number BP-100.

The three transmitters are driven by the high order bit of a 6 bit counter at 37, 38.6 and 40.2 Khz; the low order bit of these 6 bit counters are driven by crystal oscillators at 2.386, 2.4704 and 2.5028 Mhz respectively. The four receivers are each connected to three filters, one filter for each frequency. The output of the filters is amplified and converted to a square wave which strobes the corresponding phase counter into a five bit buffer whenever an upward zero crossing is received. The five high order bits of the phase counter give the phase of the transmitted signal to one part in 32, about 1/100 of an inch of sound travel in air. The arrangement of the major components of the hardware is illustrated in figure 1 below.

The head sensor phase hardware is attached to a DEC PDP-1 computer which can read the 5-bit phase buffers. The phase-buffers are read into the PDP-1's IO register three at a time. * Full revolutions of phase, as measured, are counted by a simple program in the computer. This program samples the phase measurement every 0.01 seconds and notes any major differences from the previous samples. For example, if the new sample indicates a phase of 10 degrees and the previous phase for that channel was 350 degrees the program assumes that the 360 degree mark was passed, and adds one to the integer part of its distance record.

* ...by the instruction iot 41 of which the bits 9, 10, and 11 are microcoded: bit 9 disables the strobe lines when it is zero, and enables them when set; bits 10 and 11 select transmitters 1,2,3 or 4; the bit format of the data is the phase of transmitter P in bits 1 thru 5; the phase of transmitter Q in bits 7 thru 11; the phase of transmitter R in bits 13 thru 17; and bits 0, 6 and 12 are always zero. Thus, to read all the phases: iot 41 ...disable;iot 041;dio pqr1 ;iot 141 ; dio pqr2 iot 241 ; dio pqr3 ; iot 341 ; dio pqr4 ; iot 441 ...enable.

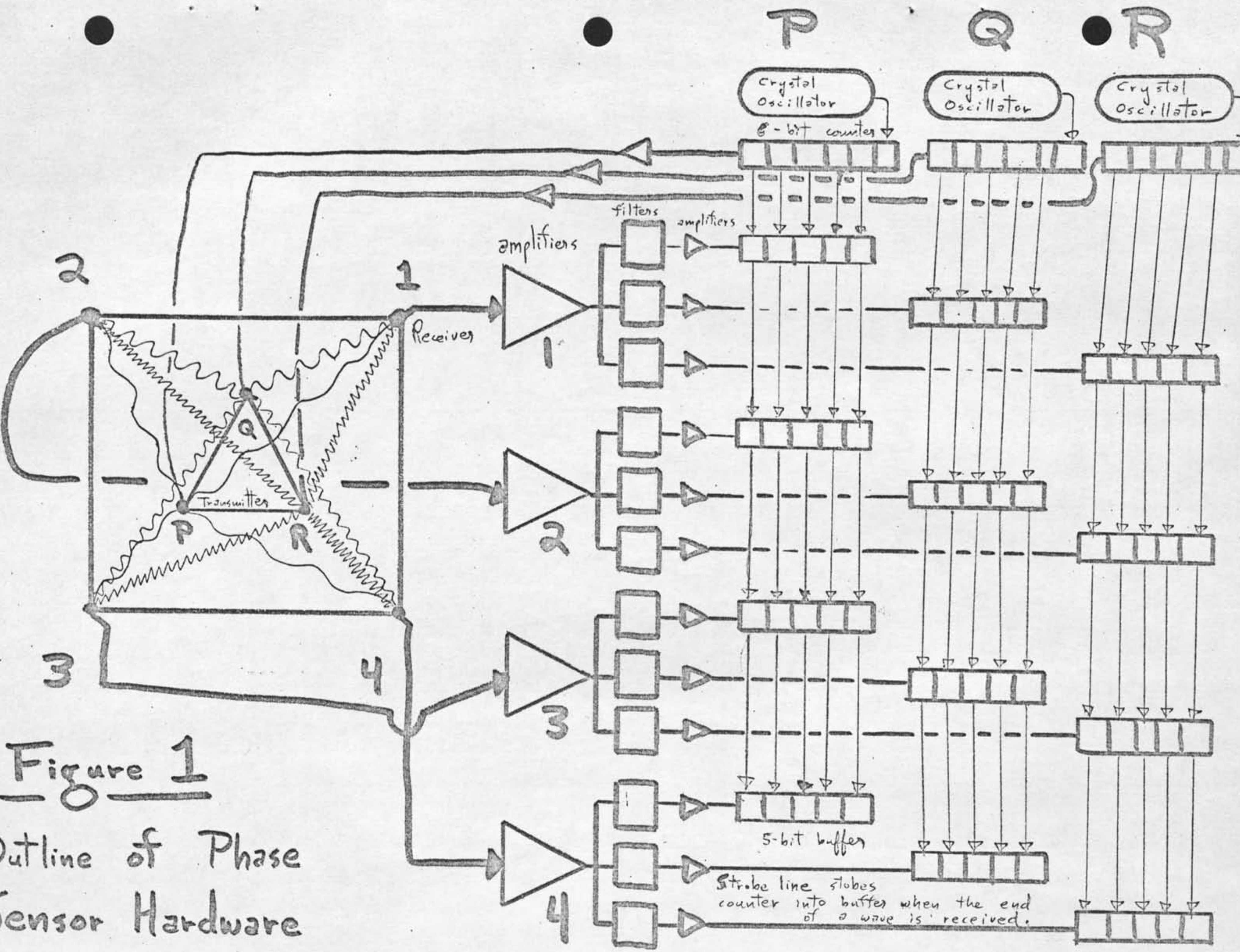


Figure 1

Outline of Phase Sensor Hardware

NOISE ERRORS

The accuracy of the ultrasonic head sensor phase system is limited by errors due to air movement, fast head motions and echoes. Because of the filters used in the receiving channels and the relatively high power of the transmitted sound, the ultrasonic head position sensor is entirely insensitive to room noise. Hand clapping, shouts, whistles, blower noise, typing, etc. have no effect on the system. However the system is quite sensitive to drafts in the room. The speed of sound in air is only about 1100 fps. Drafts of one foot per second are not uncommon even in rooms without ventilating blowers. Such drafts limit the accuracy of this ultrasonic system to about 0.1 per cent.

I have used a simple test program to observe directly the effects of wind. This test program displays the measured phases of all twelve channels as phasors on the computer's display screen. As the transmitters are moved the displayed phasors rotate, giving an instant check that all channels are operating. Thus, I have observed a jitter due to the air currents. Even when the room is still, a jitter of three or four parts in 32 of phase measurement for paths about 100 waves long is not uncommon. This corresponds to a wind of about one foot per second.

Another source of noise error is excessively fast head motions. The receiver filters have a bandwidth of about 100 cps, which is adequate for dopler shifts caused by head motions not greater than 2 or 3 feet/second - such a velocity is on the order of ordinary limb motions and should be a reasonible bound for tracking head motions. Frequency shifts greater than the bandwidth of the filters will result in loss of tracking.

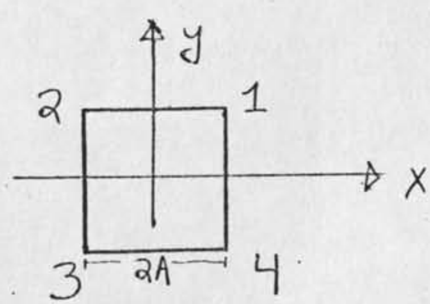
A third source of error arises from echoes of the ultrasonic waves off of surfaces and objects in the room. Although echoes are crucial, they are difficult to obverve repeatibily. I have tried two methods: First, I have set up a transmitter and a receiver so as to apparently echo off of a table. I noted the phasor positions, then I would either remove the table or cover it with a piece of material claimed to be ultrasonic absorbing. The ultrasonic absorber is a thick foam rubber mat with large triangular-prism shaped teeth on one side. Usually the phasors will change indicating perhaps the previous existence of an echo path or perhaps the perturbation of the air and apparatus caused by moving things around. Second, I have placed objects in the direct path between a transmitter and a receiver - this usually results in loss of tracking, but occassionally the phasors indicate another reading - however, I haven't been able to get the same second reading twice, although I place the same obstacle in the same location. Since isolating the apparatus and redundancy procedures avoid or are able to correct most possible echo errors - I haven't directly pursued this source of error quantitatively.

THE MAIN PROBLEM The main difficulty in using the ultrasonic phase head sensor is to deduce the absolute distances between the transmitters and the receivers from the changing phase measurements.

A direct approach would be to have a fixed stand from which the distances had been measured and use this referance location for initialization. A similar method is to touch each transmitter to each receiver and to have a set of switches to indicate to the computer which distance was then zero. I have used both of these methods with partial success, the only difficulty is that no ultrasonic path can become obstructed by the fixed stand or by initialization maneuvering. However, these methods have not been perfected because of the existence of purely geometric methods of deriving the distances from the phases.

THE GEOMETRIC SOLUTIONS The ambiguity in the distance measurements can be removed by using geometric redundancy. If the four ultrasonic receivers are arranged in a square with a side of length 2A and are assigned the following coordinates:

receiver 1 at $(A, A, 0)$
receiver 2 at $(-A, A, 0)$
receiver 3 at $(-A, -A, 0)$
receiver 4 at $(A, -A, 0)$



the actual distance from a transmitter at an arbitrary point (x, y, z) to each receiver is then:

(i.)

$$
\begin{aligned}
D_1^2 &= (x - A)^2 + (y - A)^2 + z^2 \\
D_2^2 &= (x + A)^2 + (y - A)^2 + z^2 \\
D_3^2 &= (x + A)^2 + (y + A)^2 + z^2 \\
D_4^2 &= (x - A)^2 + (y + A)^2 + z^2
\end{aligned}
$$

expanding and letting $l^2 = x^2 + y^2 + z^2$:

(ii.)

$$
\begin{aligned}
D_1^2 &= l^2 + 2A^2 - 2Ax - 2Ay \\
D_2^2 &= l^2 + 2A^2 + 2Ax - 2Ay \\
D_3^2 &= l^2 + 2A^2 + 2Ax + 2Ay \\
D_4^2 &= l^2 + 2A^2 - 2Ax + 2Ay
\end{aligned}
$$

solving for x, y, and l in terms of the distances squared:

(iii.)

$$
\begin{aligned}
x &= (D_2^2 - D_1^2) / 4A \\
x &= (D_3^2 - D_4^2) / 4A \\
y &= (D_4^2 - D_1^2) / 4A \\
y &= (D_3^2 - D_2^2) / 4A \\
l^2 &= (D_1^2 + D_3^2 - 4A^2) / 2 \\
l^2 &= (D_2^2 + D_4^2 - 4A^2) / 2
\end{aligned}
$$

a result which illustrates some of the redundancies. For, example note that by selecting three of the six equations, the coordinates of a transmitter can be obtained without relying on any one given receiver. Futhermore, by taking the difference of any of the two paired expressions above, and factoring out the constant, we arrive at the following consistency relationship between the four distance measurements:

(iv.)

$$
0 = D_1^2 - D_2^2 + D_3^2 - D_4^2
$$

If the distance measurements are inconsistent, the consistency relation will not in general hold and its non-zero value is then a measure of error, which we have called the error M, and define:

(v.)

$$M = Dm_1^2 - Dm_2^2 + Dm_3^2 - Dm_4^2$$

Where the Dm's are the inconsistent measurements, which will differ from the real distances by some errors:

(vi.)

$$\begin{aligned} D_1 &= Dm_1 + E_1 \\ D_2 &= Dm_2 + E_2 \\ D_3 &= Dm_3 + E_3 \\ D_4 &= Dm_4 + E_4 \end{aligned}$$

...the D's are the actual distances, the Dm's are the measured distances derived from the current phase reading and the accumulated count of all phase overflows minus phase underflows, and the E's are the errors for transmitters 1, 2, 3, and 4 respectively. It is important to note that the computer tracking program will maintain the same fixed errors between the D's and the Dm's even though the distances change. The errors remain fixed, because the tracking program accurately measures the actual differences in distance.

We can demand that the D's be consistent at five locii according to equation (iv.) and thus solve for the E's and the sum of their squares, by substituting equations (vi.) into (iv.):

$$(Dm_1 + E_1)^2 - (Dm_2 + E_2)^2 + (Dm_3 + E_3)^2 - (Dm_4 + E_4)^2 = 0$$

and expanding:

(vii.)

$$2\,Dm_1(E_1) - 2\,Dm_2(E_2) + 2\,Dm_3(E_3) - 2\,Dm_4(E_4) + (E_1^2 - E_2^2 + E_3^2 - E_4^2)$$
$$= -(Dm_1^2 - Dm_2^2 + Dm_3^2 - Dm_4^2)$$

or in vector notation:

(vii.)'

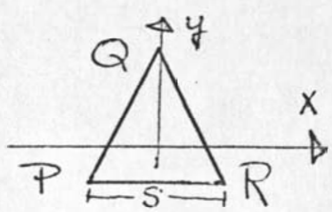
$$(2\,Dm_{1i}\,,\ -2\,Dm_{2i}\,,\ 2\,Dm_{3i}\,,\ -2Dm_{4i}) \times \begin{pmatrix} E_1 \\ E_2 \\ E_3 \\ E_4 \\ \sum \pm E^2 \end{pmatrix} = -Dm_{1i}^2 + Dm_{2i}^2 - Dm_{3i}^2 + Dm_{4i}^2$$

$i = 1$ to 5

Equation (vii) is the geometric relationship used to track an arbitrary transmitter point from a square array of receivers.

It is possible to use the above relationship with only four locii and to solve four quadratics in four unknowns. The extra answers can be eliminated by initializing the ultrasonic tracking program at zero wave counts and then insisting upon answers which are positive and smaller than the dimensions of the work area.

If the ultrasonic transmitters are placed in an equilateral triangle with a side of length S and are lettered p, q, and r and assigned the following coordinates:



p at $(-S/2\ ,\ -S/(2\sqrt{3})\ ,\ 0)$

q at $(0\ ,\ S/(\sqrt{3})\ ,\ 0)$

r at $(+S/2\ ,\ -S/(2\sqrt{3})\ ,\ 0)$

the distance from a receiver at an arbitary point (x, y, z) to each transmitter is then:

$$D_p^2 = (x + S/2)^2 + (y + S/(2\sqrt{3}))^2 + z^2$$
$$D_q^2 = x^2 + (y - S/(\sqrt{3}))^2 + z^2$$
$$D_r^2 = (x - S/2)^2 + (y + S/(2\sqrt{3}))^2 + z^2$$

expanding and letting $l^2 = x^2 + y^2 + z^2$:

(viii.)

$$D_p^2 = l^2 + S^2/3 + S\,x + (S/\sqrt{3})\,y$$
$$D_q^2 = l^2 + S^2/3 - (2S/\sqrt{3})\,y$$
$$D_r^2 = l^2 + S^2/3 - S\,x + (S/\sqrt{3})\,y$$

solving for x, y, and l in terms of the distances squared:

(ix.)

$$x = (D_p^2 - D_r^2)\,/\,2S$$
$$y = (D_p^2 + D_r^2 - 2D_q^2)\sqrt{3}\,/\,6S$$
$$l^2 = D_p^2 + D_q^2 + D_r^2 - S^2$$

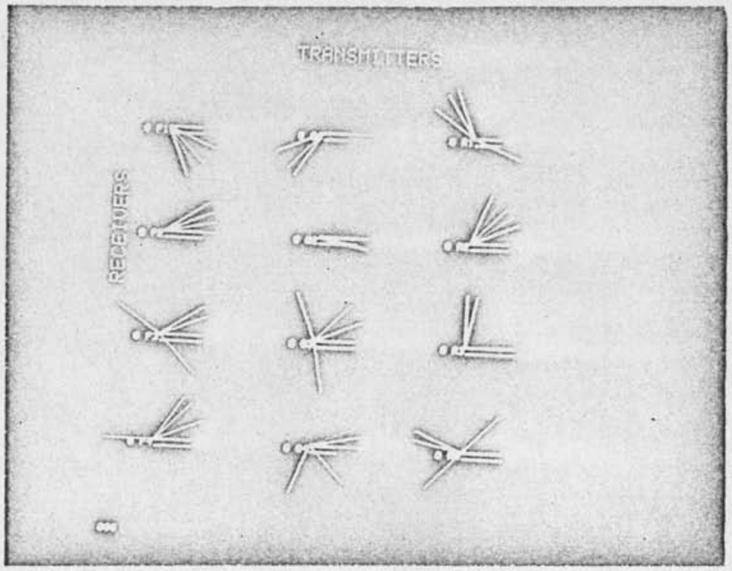
There is no redundancy to exploit directly in tracking just one receiver, however if we consider the x, y and l^2 for all four receivers along with the earlier x, y and l^2 for the three transmitters, in the square's coordinates, we can then write 18 expressions for all the fixed lengths in the system: 6 expressions for the sides and diagonals of the square as seen from the triangle, and 12 expressions for the sides of the triangle as seen from the square. These 12 expressions for the triangle come about because one expression for each side of the triangle can be derived for each of the four isocelles right triangles contained in the square. If the length of a side calculated from a given combination of Dm's is within a tolerance, the Dm's involved are validated. If enough Dm's are valid, then the remaining invalid Dm's can be corrected from valid x, y and l^2 for the points in question.

PROGRAMMING

The geometric soltuions have been coded and checked out on a simulator of the ultrasonic hardware. The simulator includes three orthogonal views of the position and orientation of the triangle with respect to the square, phasors indicating the phase, line segments representing the magnitudes of the D's, Dm's, M errors and length of sides of the triangle. The simulator displays are also helpful in monitoring actual hardware performance. Photographes of the displays are included on a separate page below.

The equation (vii.) geometric solution was programmed using a Gauss Elimination to solve the System of Linear Equations. The program works adequatly on data supplied by the simulator of the head sensor phase hardware. It is ultimately restricted by the independence of the linear equations at the locii where samples are taken. Independence can be insured, by doing the Elimination after each new sample locus and demanding that the product of the pivots found so far be large, otherwise the new locus is not included and the computer waits for a better point. Also, I would note that although an 18-bit fixed point elimination was attempted for the sake of computational speed on the PDP-1, a floating point simulator was finally necessary to maintain accuracy.

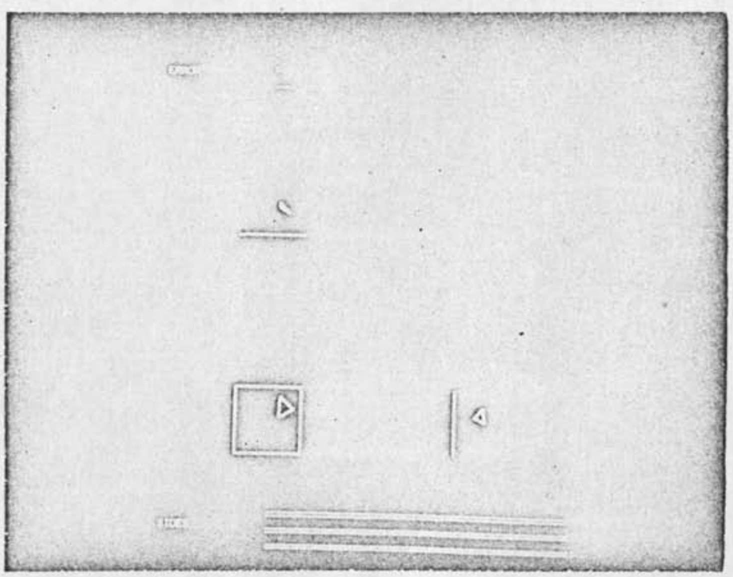
Phase Sensor Hardware Simulator Displays



Phasors

D's

Dm's



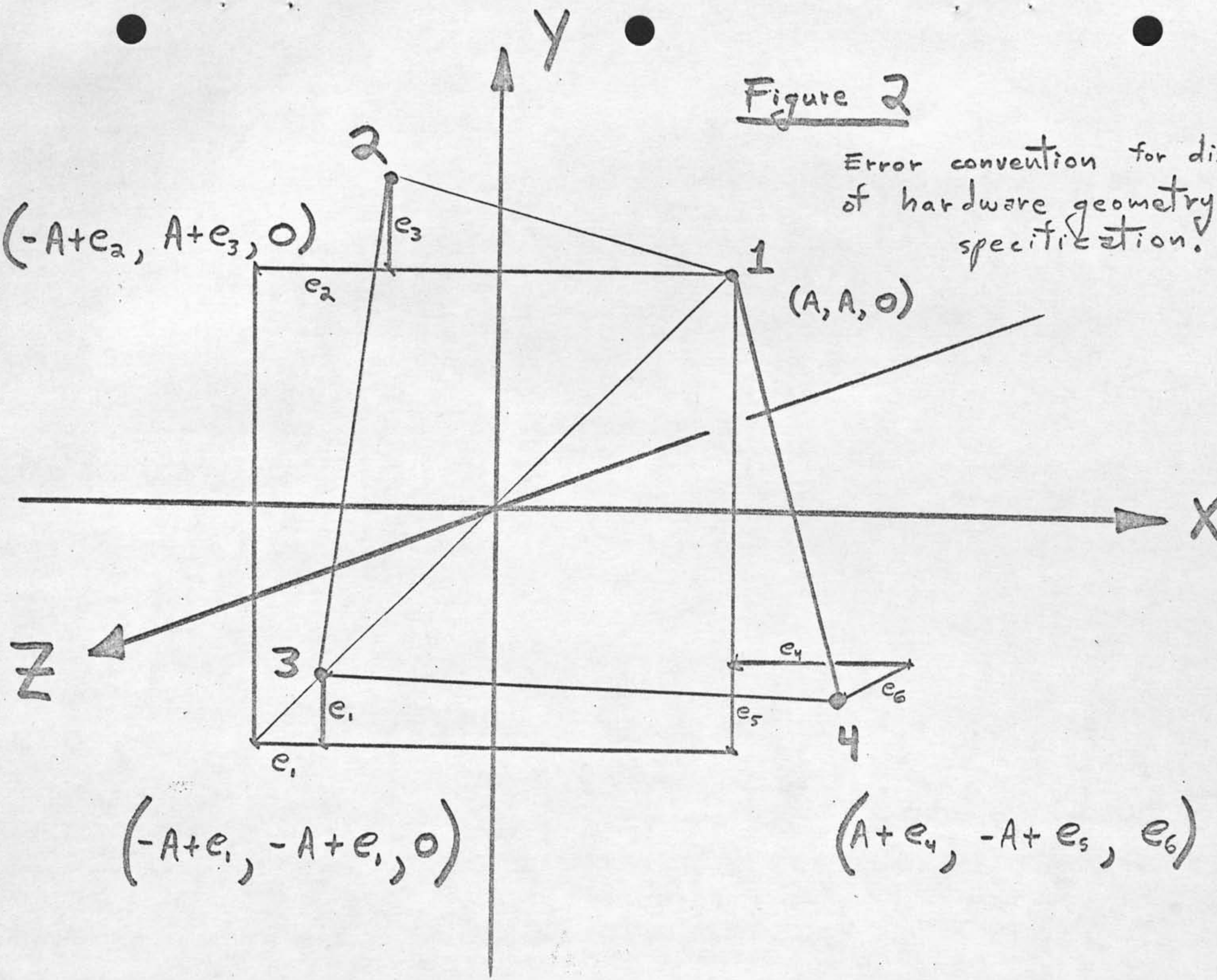
M errors

Position and Orientation of simulated Transmitters + Receivers

Sides of Triangle

Figure 2

Error convention for distortion of hardware geometry from specification.



A linear equation system correction technique, Seidel's recursion, was tried, but it only affected the very low order floating bits and didn't change the fixed point answers. However, the method has failed to work on data from the real ultrasonic hardware because of the following difficulties: i. Failure of the hardware geometry, the receivers were not in a square. ii. Phase noise, there is a constant jitter in the phase measurements. iii. Ordinary hardware failures - cables shorted, flip-flops that stick, etc.

The correction of invalid Dm's from valid ones by geometric redundancy has been coded and checked out with simulated data and simulated perturbations of the Dm's. This procedure takes full advantage of all the redundancies in the system. In any tenth of a second or less, this procedure could suppress any two bad Dm's which might appear, and up to four bad Dm's if they didn't all involve the same transmitters and receivers. It is hoped that this procedure will be able to suppress a considerible load of stray hardware bugs, echoes and noise. However, this procedure is dependent on the special geometry of the hardware, which so far has failed to meet the specifications.

HARDWARE GEOMETRY ERROR DETECTION

Although, the ultrasonic receivers could be merely secured to the ceiling in any convenient non-coplanar quadrilateral arrangment and thier exact locations measured - the geometric procedures mentioned above have continued to motivate attempts to place the receivers in a square, thus necessitating a method for detecting errors in the allignment of the receivers. The transmitter triangle is small, in order to fit easily on the viewer's headset, and the transmitters are rigidly and accurately mounted in plastic. The receiver square, on the other hand, must be large with respect to the work area, because a small square would lead to approximately similar D's which by equation (iii), would lead to error sensitive calculation of the x and y coordinates of an arbitrary transmitter point.

Four arbitrary points in space have 12 coordinates, thus 12 degrees of freedom. A square of side 2A has six coordinates, three for translations and three for rotations, thus 6 degrees of freedom. Thus, there are 6 possible degrees of freedom or errors in which four arbitrary points can diverge from being in a square. In order to discuss the distortion of the receiver square, we must assign names to the position errors. Consider the perfect square of side 2A as a referance frame; receiver 1 can be placed at perfect corner 1; receiver 3 can be placed on the 45-degree line through perfect corners 1 and 3 with an error in its x and y coordinates of e1 ; receiver 2 can be placed in the plane of the perfect square with errors e2 and e3 in x and y ; and the locus of receiver 4 is determined, thus forming a very flat tetrahedron with errors from perfect corner 4 of e4, e5 and e6 in x, y, and z respectively. This choice of errors is diagrammed in figure 2.

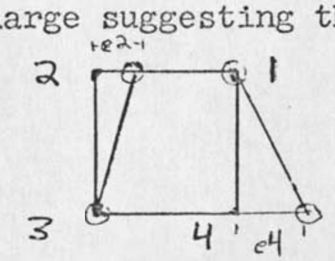
The exact size of the square can be entered thru the software, so we will assume that the 2A is adjusted to make e1 zero. We can again derive the distances from the receiver points to a transmitter at an arbitrary point (x,y,z), and can find the M error, from definition (iv.), which is now a function of x, y, and z:

(x.)

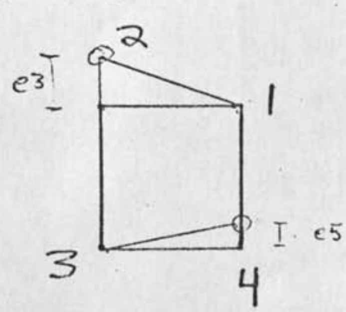
$$M = 2\,(e2 + e4)\,x + 2\,(e3 + e5)\,y + 2\,e6\;z$$
$$+2A\,(e2 - e3 - e4 + e5)$$
$$e2^2 - e3^2 - e4^2 - e5^2 - e6^2$$

Thus, by moving the transmitter along the x, y and z axes, and observing the computer's evaluation of the M error on a display screen, we can tell what sort of distortions are present.

If the M error varies strongly with motions in the z direction, then the receivers are not coplanar. If the M error varies strongly with motions in the x direction, then e2 and e4 are large suggesting that the sides $\overline{14}$ and $\overline{23}$ are not parallel:



If the M error varies strongly with motions in the y direction, then e3 and e5 are large suggesting that the sides $\overline{12}$ and $\overline{34}$ are not parallel.



If the M error is roughly a large constant everywhere, this suggests that:

(xi.)

$$e2 + e4 = 0$$

$$e3 + e5 = 0$$

$$e6 = 0$$

And if we write out expressions for the slopes of each side of the square in terms of the "e" distortions:

(xii.)

$$\text{slope} = (y - y') \;/\; (x - x')$$

$$\text{slope } \overline{12} = -e3 \;/(2A - e2)$$

$$\text{slope } \overline{43} = e5 \;/(2A + e4)$$

$$\text{slope } \overline{23} = (2A + e3) \;/\; e2$$

$$\text{slope } \overline{14} = (2A - e5) \;/\; -e4$$

Substituting equations (xi.) into (xii.) yields:

$$\text{slope } \overline{12} = \text{slope } \overline{43}$$

$$\text{slope } \overline{23} = \text{slope } \overline{14}$$

Thus indicating a parallelogram. If the M error > 0 then the $\overline{13}$ diagonal is too large, while if M error < 0 then the $\overline{24}$ diagonal is too large.

In order to track head position to within half a wavelength of ultrasound, that is about a sixth of an inch, the M error due to hardware distortions must be sig-

significantly smaller than the M error due to small inconsistencies in the actual distance measurements. Let the accuracy we want in Dm be R; and let the tolerance we must have in the distortions of the square be T; and let M be the M error defined in expression (v.); accordingly:

$$\frac{\partial M}{\partial ei} \times T < \frac{\partial M}{\partial Dmj} \times R$$

Take for example e2 and Dm1; that is i = 2 and j = 1:

$$T < \frac{2\,Dm1 \times R}{2(\,x + A + e2\,)}$$

Reasonibly: $0 \leq |e2| \leq 1''$

$A = 23''$

$0 \leq |x| \leq 48''$

$0 \leq |Dm| \leq 48''$

$R = (1/6)''$

Thus approximately:

$$T < 2/3\ R = (1/9)''$$

The square of the receivers must be good to better than (1/9)".

SUPPRESSION OF THE REMAINING ERRORS

Most errors due to stray noise and echoes can be adequetly handled by the geometric redundancy checks outlined above. Further efforts, however, might pursue suppressing the air motion jitters by an appropriate averaging over several obervations and countering the resulting tendency for sluggish tracking by keeping track of velocity and acceleration to anticipate displacements, as has been done in some light pen tracking routines.

CONCLUSION

Other techniques of tracking body motion include Roberts Lincoln Wand of Lincoln Laboratory and P. deBruin's Spark-Wand of the Harvard Physics Dept. ; both of which are pulsed ultrasonic techniques. Other techniques might involve pulsed or phased microwaves, inertial tracking or mechanical linkages. In any event, ultrasonic phase tracking still appears to be a feasible, although non-trivial solution to the demand for such a graphics input device.

Part IV: A Clipping Divider

The device described here was designed and constructed at Harvard University by Associate Professor Ivan E. Sutherland, Principal Investigator, and Robert F. Sproull. Agency support was used to construct the hardware described in Part IV.

In addition work to integrate the Clipping Divider into a total three-dimensional display was supported in part by the Advanced Research Projects Agency (ARPA) of the Department of Defense under contract SD 265, and in part by the Office of Naval Research under contract ONR 1866(16).

The material in Part IV has been accepted for Mr. Sproull's presentation at the Fall Joint Computer Conference.

Final Report Aug. 1968

ABSTRACT

This paper describes a new algorithm for solving the "windowing" problem and special-purpose hardware which uses this algorithm. The clipping divider computes either perspective views of three-dimensional drawings or arbitrarily magnified views of two-dimensional drawings. It eliminates those portions of the drawing outside of a rectangular "pyramid of vision" or a rectangular "window" respectively, by computing scope coordinates for the ends of the visible portion of each line. Combined with a matrix multiplier, which is described briefly, the clipping divider can compute true perspective views of moving three-dimensional objects fast enough to present pictures of up to 3000 lines flicker-free. The clipping divider and matrix multiplier can also display a wide variety of curves and curved surfaces automatically.

The paper also shows how the clipping divider is relevant to a new method for solving the "hidden line" problem in order to display opaque three-dimensional objects.

INTRODUCTION

When compared with a drawing on paper, the pictures presented by today's computer display equipment are sadly lacking in resolution. Most modern display equipment uses 10 bit digital to analog converters, providing for display in a 1024 by 1024 square raster. The actual resolution available is usually somewhat less since adjacent spots or lines will overlap. Even large-screen displays have limited resolution, for although they give a bigger picture, they also draw wider lines so that the amount of material which can appear at one time is still limited. Users of large paper drawings have become accustomed to having a great deal of material presented at once. The computer display scope alone cannot serve the many tasks which require relatively large drawings with fine details.

On the other hand, a drawing can be represented in computer memory with very high resolution and precision. For example, if each coordinate is represented with 16 bits, a picture 65 inches square can be represented with resolution of about a thousandth of an inch. If such a picture could be displayed with its full resolution, it would be far better than can be provided on paper. Moreover, it is often convenient to represent coordinates in memory, for example in an 18 bit computer, with more than the 10 bits used by the display scope, even though the additional resolution may not be seen in every view. With such great resolution availability, large and complex pictures can be represented which contain exceedingly fine details.

Unfortunately, the limitations of display equipment prevent a user from seeing the entire drawing and the fine detail simultaneously. But a sophisticated computer graphics system should provide for expansion of the picture so that any part of it can be examined in detail. The ability to expand the picture so that fine details are made visible partly compensates for the lack of resolution available in the display itself.

If the picture on the display is enlarged, parts of it may move off the screen. Programs to enlarge the drawing must compute not only the location of each part of the drawing after enlargement, but also which parts of the drawing are to appear at all. If all of a particular line or figure remains in view, it may simply be enlarged. If a figure or line moves entirely out of view, it must be eliminated from the picture. If a figure or line intersects the edge of the visible area, the part of it which is visible must be shown and the part of it outside the visible area must be eliminated.

The process of eliminating parts of a drawing which lie outside the observer's field of view has come to be known as "windowing" (1). One can think of the task as if one were looking at a large drawing through a small window, as shown in Figure 1. Everything that lies within the window should be shown, everything outside the window should be eliminated. If the window is made bigger, more material will be shown but will be correspondingly smaller on the display scope. If the window is made smaller, the material still inside it will appear on the screen correspondingly enlarged. Windowing is most difficult for parts of the drawing which are only partly visible.

There are two main methods used to accomplish windowing: blanking and clipping. If blanking is to be used, the display scope itself must be blanked electronically whenever it is asked to display information outside the visible region. Systems which provide for blanking must provide not only for accurate display within the visible region, but also for deflection far outside the screen area. If the picture is to be very much larger than the actual scope area, the accuracy required of the electronic components involved may make them inordinately expensive. In any case, because the display must trace out both the visible and the blanked parts of the picture, the flicker rate will depend on the complexity of the total drawing regardless of how little may actually be seen.

Windowing may also be performed by clipping, the process of discovering which portions of a drawing are within the window and computing appropriate scope coordinates for them. If clipping is used, the display is given only valid visible information with the portions of the drawing outside the window already eliminated. For drawings composed of straight lines, clipping requires only enough arithmetic to compute the intersection of a line with the edge of the window. Because the clipping process requires many tests to decide whether a line intersects an edge of the window and if so which one, clipping programs are relatively slow. A typical clipping program takes between one and ten milliseconds per line clipped.

Because it is essential to perform windowing if drawings are to be enlarged, nearly all sophisticated computer graphics systems do windowing. They do a lot of windowing because the entire drawing must be processed each time the picture shown on the 'scope is moved or changed in scale. Display equipment manufacturers are beginning to provide some hardware assistance to the windowing task, usually in the form of blanking capability. Yet windowing is still a problem because the methods previously available have been too slow. If blanking is used, the flicker rate of the display suffers; if clipping software is used each motion or enlargement of the picture may cost several seconds of delay. This paper describes a device which solves the windowing problem at a speed commensurate with high-performance line-drawing display equipment.

CLIPPING IN TWO AND THREE DIMENSIONS

Our clipping divider came about through a need to generate dynamic perspective images of three-dimensional objects. The head-mounted three-dimensional display project described elsewhere in this issue (2) calls for three-dimensional information to surround the observer. The clipping divider is necessary to perform the division required for a true perspective projection and to eliminate those parts of the three-dimensional drawing behind the observer or beside him but outside of the limited field of view provided by the display. The clipping divider has to operate fast enough to process information as it is displayed so that the picture can be updated as the user moves his head.

The material to be processed by the clipping divider is always a line or vector drawing. Each part of the drawing is made up of straight line segments specified either in terms of their absolute end point coordinates or in terms of the position of one end relative to the other. In the original three-dimensional perspective task, each absolute or relative specification is given in a three-dimensional coordinate system whose origin is at the user's eye. The three-dimensional drawing is specified with up to 20 bits for X, 20 bits for Y, and 20 bits for Z so that the resolution available at the clipping divider input is far higher than required by the scope. The high input resolution is needed whenever the observation point is placed very close to an object.

In a two-dimensional application the clipping divider accepts information about the lines or vectors of a large flat drawing. We think of the drawing as being written in memory on a large "page" of paper and call its coordinate system "page coordinates" to contrast them with "scope coordinates". The clipping divider will present on the scope only the part of the drawing within the "window". The window is a rectangle on the drawing aligned with the coordinate axes. The window is specified by giving the page coordinates of its left, right, bottom, and top edges, up to 20 bits each. These four numbers are stored internally in the "window" registers. Each XY location in the drawing may be specified in page coordinates with up to 40 bits, 20 for each axis. The coordinates stored in memory can be in a form suitable for computation rather than packed in a way peculiar to the display. In an 18 bit machine, for example, the two least significant bits of the 20 bit clipper input are made to be a copy of the sign bit so that each coordinate occupies a single word of storage. Coordinates can be treated with the ordinary arithmetic instructions of the computer. There is no need to pack X and Y information into a single word for use by the display. The 20 bit input resolution is useful because the clipping divider can magnify a portion of the drawing to show on the display scope.

Whereas clipping in two dimensions is by now a fairly familiar process, how to do clipping for perspective projections is less widely known. In order to present a perspective picture of material which surrounds the observer, clipping must be done in three dimensions before doing the perspective division. Clipping must preceed division because the unclipped ends of three-dimensional lines may have negative or zero values of Z. Division by a negative Z value will give an erroneous position on the wrong side of the picture; division by zero or too small a value of Z will cause overflow.

In three-dimensional applications the region within which lines are visible is a pyramid whose vertex is at the eye. The left, right, bottom, and top edges of this "pyramid of vision" are the planes X=-Z, X=+Z, Y=-Z, and Y=+Z respectively, as shown in Figure 2. The clipping process in three dimensions involves computing the intersection of each line with these four planes. The pyramid of vision encompasses a 90 degree field of view. Scaling before the clipping process can provide for other viewing angles.

The clipping divider maps whatever drawing information falls within the pyramid of vision or the window onto a portion of the scope face. The portion of the scope within which information is presented is a rectangle aligned with the axes of the scope. The size and position of this rectangle, or "viewport", is specified by giving the scope coordinates of its left, right, bottom, and top edges, as shown in Figure 3. These four numbers are stored internally in the "viewport" registers. If three-dimensional information is being presented, the viewport will contain a perspective picture of the part of the three-dimensional drawing which falls within the field of view. If two-dimensional information is being presented, the viewport will contain an enlarged version of the information which falls within the window. The ability to map information onto a part of the scope face rather than all of it is important if several views of a single drawing are to be presented simultaneously (3,4).

THE MIDPOINT ALGORITHM

The clipping divider utilizes a new algorithm for solving the windowing problem. We call this algorithm the "midpoint" algorithm because it involves computing the midpoint of the line. The midpoint is easily found by adding together the endpoint coordinates and shifting the sum right one bit. If implemented in software, the midpoint algorithm would be slower than a direct geometric computation of the intersection of the line and the edge of the window. Hardware which implements the algorithm, however, is able to capitalize on the fact that additions are much easier to perform than either multiplication or division.

The clipping divider distinguishes three kinds of lines:

1) lines with neither end in view,

2) lines with only one end in view, and

3) lines with both ends in view,

as shown in Figure 4. In each case, the operations performed reduce the line to a shorter line of a simpler case.

For lines of the first case with neither end in view, we check to see if some portion of it could possibly be in view. Obviously if both ends of the line are:

a) to the right of the window,

b) above it,

c) to the left of it, or

d) below it

then no portion of the line could possibly be seen and the line can be rejected. In the three-dimensional case, some time can be saved by also rejecting lines if both start and end have negative Z values. If the line passes this "trivial test", we compute its midpoint.

The midpoint of the line is either inside the window or outside. If the midpoint is inside the window, we can treat the line as two segments of case two, each of which has one end, the common midpoint, in view. If the midpoint is not within the window, it divides the line into two pieces, only one of which can possibly pass through the window. As shown in Figure 5, the trivial test on each of the pieces tells which to reject, leaving a shorter line neither of whose ends is in view. If the trivial test indicates that both halves should be rejected, no part of the line passes through the window. Thus lines with positive slope will be rejected if any point is detected within the regions shown shaded in Figure 6. Lines with negative slope will be rejected if any point is found within similar regions at the other corners.

For lines of the second case with only one end of the line in view, we again compute the midpoint of the line. If the midpoint is outside the window, half of the line can be eliminated. If the midpoint is inside the window, it is closer to the edge of the window than the original point and still, of course, on the line. We continue to compute midpoints within the segment which intersects the window edge to make a logarithmic search for the place where the line penetrates the edge of the window.

Finally, having reduced the line to the third case where both ends are within the window, albeit on the edge, we convert these endpoint coordinates to coordinates suitable for display on the scope. This conversion involves division by the window size in two dimensions or by the Z depth in three dimensions, and multiplication by an appropriate factor to account for the size and position of the viewport. These conversions are shown in Figures 7 and 8. Because the points used in the division are guaranteed to be within the window or pyramid of vision, overflow will never occur.

WINDOW-EDGE COORDINATES

During the clipping process, information about a line is represented in a special coordinate system based on the edges of the window. Each point is represented as four numbers, each of which tells how far the point is from one edge of the window. These four numbers can be thought of as a four-component vector whose components are given by:

$$[\ X - W_L\ ,\ X - W_R\ ,\ Y - W_B\ ,\ Y - W_T\ ]$$

in two dimensions, or

$$[\ X - (-Z)\ ,\ X - (+Z)\ ,\ Y - (-Z)\ ,\ Y - (+Z)\ ]$$

in three dimensions. This "window-edge" coordinate system makes it very easy to tell if a point is inside or outside of the window. The ordinary coordinates of the point are easily retrieved from its window-edge representation by adding or subtracting components.

The sign bits of the four components of the window-edge representation contain all the information required to test the position of a point relative to the window or pyramid of vision. If the signs of the four components of the window edge representation are + - + - respectively, the point is visible. For any other combination of signs, the point is outside the viewing region. We have found it convenient to think of the position of a point relative to the window in terms of a simple four bit code derived from the signs of the window-edge representation. This four bit "out code",

$$OC = [\ S_L\ ,\ \overline{S_R}\ ,\ S_B\ ,\ \overline{S_T}\ ].$$

The sign bits of the right and top components have been complemented so that "one" indicates a position outside the window. Figure 9 shows the out codes for different positions around the window or pyramid of vision.

Whether or not a line should be rejected can be determined by the logical intersection of the four-bit "out codes" for its start (subscript s) and end (subscript e). If both start and end are to the left of the window, for example, both out codes will have a "one" in their first component, their intersection will be non-zero, and the line can be rejected. The trivial rejection critereon is thus:

If $(OC_s \text{ and } OC_e) \neq 0$ then reject.

Similarly, if the mid-point of a line is not on the screen, the intersection of the out codes for the start, end, and midpoint tells which part of the line to reject.

If $(OC_s \text{ and } OC_m) \neq 0$ then reject first half.

If $(OC_e \text{ and } OC_m) \neq 0$ then reject second half.

If both halves of the line are rejected, of course, the line is completely eliminated, as was shown in Figure 6.

The same "out code" tests serve for both the two- and three-dimensional clipping. In the three-dimensional case, lines which pass behind the observer are automatically rejected by the same tests.

HARDWARE

The clipping divider contains eight individual adders arranged in two groups of four. Each adder has associated with it a working register and a shifting register, as shown in Figure 10. One group of four adders is used to determine where the line enters the field of view, the other group of four adders is used to determine where the line leaves the field of view. After the clipping process is complete, the adders are rearranged into four groups of two. Each pair of adders does a division and multiplication to provide for scaling in two dimensions or perspective division in three dimensions. The clipping divider also contains registers which hold the position of the window edges and the viewport edges, and the coordinates of a staring position to use for the line.

The clipping operation is begun by loading the working registers with the window edge coordinate representation of the two ends of the line. If the line is specified as a relative vector, the specified displacements are added to the starting coordinates to find the end of the line. Enough adders are available to do the addition of all coordinates simultaneously. The shifting or "delta" registers are loaded with the displacement required to go from one end of the line to the other. If absolute specifications for the ends of the line were given, this displacement is computed as their difference. When the setup process is finished, each group of adders has been provided information about the absolute coordinates of its end of the line and the displacement required to get to the other end.

The possibility of trivial rejection is checked next. If both ends are outside the window, their "out codes" may show that the line can be trivially rejected. If both ends are in the window, the clipping process can be omitted. If the line is a non-trivial case, computation proceeds.

Each step in the clipping process involves shifting the delta registers one place to the right. After the first shift, the delta registers contain the displacement required to reach the midpoint of the line. The adder output will then be the midpoint coordinates. If the "out code" tests described in the previous section are satisfied, the midpoint value replaces the endpoint value in the working registers. If the out code tests fail, the midpoint is discarded. Accepting the midpoint as a replacement for the endpoint is equivalent to eliminating the half of the line next the endpoint. Because each step starts by shifting the delta registers one place to the right, each step considers a line that is half as long as the previous one. The clipping process is complete when the delta registers contain zero.

The clipping process as implemented here is essentially a vector version of ordinary division. A "quotient" could be generated by recording successive "zeros" or "ones" according to the acceptance or rejection of each successive midpoint. The quotient would be a binary fraction representing the ratio of the length of the clipped-off part of the line to the total length of the line. Because we do not want the quotient, we do not bother to record it. We are only interested in the coordinates of the window edge intersection. Notice that when the clipping process is complete, the component of the window edge specification which corresponds to the edge intersected will be zero. Scalar division hardware also reduces the numerator to zero. Unlike a scalar division, however, the vector division described here has other components which are non-zero. These other components carry the answer information we want.

After the clipping process is complete, the working registers contain the ends of the visible segment of the line in window edge coordinates. These coordinates must be converted to the appropriate scope coordinates to position the displayed line properly on the scope picture. The sum of pairs of window edge coordinates can be used to find the position of the point relative to the center of the window, WC. For example:

$$(X - W_L) + (X - W_R) = 2X - (W_L + W_R) = 2(X - WC_x) .$$

In two dimensions, the difference of pairs of window edge coordinates can be used to find the size of the window, WS, for scaling.

$$(X - W_L) - (X - W_R) = (W_R - W_L) = 2(WS_x) .$$

In three dimensions the difference can be used to find the depth information, Z, for perspective division.

$$(X - (-Z)) - (X - (+Z)) = 2(Z) .$$

Thus in both two and three dimensions the clipping divider divides the sums of pairs of window edge coordinates by their differences.

The transformation used in going from the clipped endpoints to the scope also involves the size and position of the viewport, as was shown in Figures 7 and 8. The transformations involve both division by the window size or Z coordinate and multiplication by the viewport size. The division and multiplication are performed simultaneously by pairs of coupled adders. One adder with its associated shifting and working registers is used as an ordinary scalar divider. The other adder with its working and shifting registers provides for the multiplication. Instead of recording the bits of the quotient as they are generated by the divider, the bits are used immediately to control addition of the multiplicand to the accumulating product in the multiplier, as shown in Figure 11. If the sign test in the scalar division is successful, the output of both adders replaces their respective working registers. This simultaneously provides a new dividend for the next trial, and a new partial product closer to the answer.

INTERFACING FOR THE CLIPPER

The clipping divider was designed to be part of a complete display system. The clipping divider is provided input data by a memory interface channel which fetches the data from memory. The channel is capable of interpreting codes in the information it gets from memory as special instructions, some of which it uses to direct the actions of the clipper. The clipping divider delivers its output to an ordinary two-dimensional line-drawing display. The clipping divider is intended only to provide the very considerable arithmetic capability required between memory and the display scope.

The clipping divider is an independent separately-timed, digital computing device. It watches an input flag which is raised whenever input data is available. As soon as its previous task is complete, it will accept the input data and clear the input flag. Along with the input data it accepts a 16 bit "directive" which indicates what the clipping divider is to do with the data. If no output is generated as a result of the task assigned, the clipping divider returns to a waiting state until the input flag is again raised. If some output is generated, the clipping divider raises an output flag and waits for the output to be accepted before it will again accept input data.

The clipping divider was designed to run with a 250 nanosecond clock. It takes 42 clock cycles to complete a worst case clipping task which at design speed would be ten and one half microseconds. Many tasks, however, take far less time. Trivial rejection of a line, for example, takes only 3 add times. Non-trivial rejection of a line takes between 5 and 21 add times, depending on how many steps are required to determine that the line can be rejected. If both ends of the line are inside the window, clipping can be omitted, and only the scaling or perspective division need be done. If both ends of the line are visible, the clipping divider will scale it correctly in 25 add times. As this is written, the clipper is operating at about half of the design speed.

WINDOWS WITHIN WINDOWS

It is often useful to structure information for display on a CRT. Subpictures or symbols, such as a resistor in an electrical drawing, an integral sign in a mathematical expression, or a standard part on a mechanical drawing need only be stored in computer memory once. A symbol may be displayed in many different positions on the CRT by multiple calls on its single definition just as a program subroutine may be called from many places. The memory interface channel which delivers data to the clipping divider keeps track of subroutine returns, nesting of subpictures, and previous window or viewport sizes.

The process of displaying a subpicture on the scope can be thought of as the combination of two transformations. The first transformation will place a replica of the symbol, possibly reduced in size, on the large drawing in memory. The second transformation will paint a picture on the scope, possibly with magnification, of the symbol on the drawing. The first transformation maps all the information inside a "master" rectangular area of the definition space onto a rectangular region or "instance" rectangle within which the symbol is to appear on the page. The second transformation maps whatever portion of the instance is visible within the current window area onto the appropriate viewport on the scope face. These two transformations are shown at the top of Figure 12.

The clipping divider can perform both transformations required for displaying subpictures in a single step. If the structure of the drawing calls for nested subpictures, the entire set of transformations required by any nesting can be handled at once. The clipping transformation used in the clipping divider,

$$X' = \frac{X-WC_x}{WS_x} VS_x + VC_x ,$$

is rich enough so that any combination of two or more such transformations is a single transformation of the same form. All that is required to combine transformations when entering a display subroutine is to compute and load the new composite clipping limits.

The job of computing new clipping limits involves many tests to distinguish between cases. For example, suppose a portion of the symbol is inside the window and a portion is outside. During the display of such a symbol the clipper box should use a new window, W', and a new viewport V', each smaller than before. On the other hand, if the symbol is entirely inside the window, only the viewport will be reduced in size. If none of the symbol is visible, then there will be no new clipping limits and, in fact, there is no reason to do the subroutine at all. Some examples of these cases are shown in Figure 13.

Because the computation of clipping limits is similar in kind to the clipping computations for lines, it is easily implemented with the same hardware. A special control mode has been provided for performing this computation. The memory interface channel stores the previous window and viewport values in a pushdown stack prior to display of a symbol. The channel then gives the clipper data from memory describing the size and position of the symbol. The clipping divider can establish the new composite clipping limits automatically.

THE MATRIX MULTIPLIER

We have also designed and built a digital four-by-four matrix multiplier. Because we represent data in homogeneous coordinates, a single four-by-four matrix multiplication can account for both translation and rotation. Together, the matrix multiplier and clipping divider can present perspective views of three dimensional objects tumbling in real time. The combination can also be used to display curves as will be described in the next section.

The matrix multiplier uses a separate multiplier module for each column of the matrix. Each module contains an accumulator, a partial product register, storage for the four matrix elements in that column, and the multiplication logic. The entries of a row of the matrix serve simultaneously as four separate multiplicands. An individual component of the incoming vector serves as the common multiplier. The four multiplications for a single row are thus performed simultaneously. For additional speed, the bits of the multiplier are examined four at a time rather than individually to control multiple-input adding arrays.

DISPLAY OF CURVES

The matrix multiplier and clipping divider can be used for generating a wide variety of curves quite rapidly. Suppose, for example, that we have stored in computer memory a collection of vectors of the form [t^3 , t^2 , t , 1] for which t varies uniformly from 0 to 1. The first of these vectors will, of course, be [0 , 0 , 0 , 1] and the final one will be [1 , 1 , 1 , 1]. If these vectors are multiplied by a particular four by four matrix, as shown below, the resulting vectors will be cubic polynomials in t where the coefficients of the polynomials are the entries of the matrix.

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} t^3 \\ t^2 \\ t \\ 1 \end{bmatrix} = \begin{bmatrix} (a_{11}\, t^3 + a_{12}\, t^2 + a_{13}\, t + a_{14}) \\ (a_{21}\, t^3 + a_{22}\, t^2 + a_{23}\, t + a_{24}) \\ (a_{31}\, t^3 + a_{32}\, t^2 + a_{33}\, t + a_{34}) \\ (a_{41}\, t^3 + a_{42}\, t^2 + a_{43}\, t + a_{44}) \end{bmatrix}$$

Although the process of generating the four cubics can easily be thought of as a matrix multiplication of data stored in memory, the four cubics are actually generated by difference equation methods. Each new point on the curve requires only three additions per cubic equation, or twelve additions in all. The equipment does not make any references to memory during generation of a curve.

Because the clipping divider divides the X and Y values it is given by two separate Z values (which are usually made to be the same), the resulting positions of the points on the display will follow equations of the form:

$$X_s = \frac{a_{11} t^3 + a_{12} t^2 + a_{13} t + a_{14}}{a_{21} t^3 + a_{22} t^2 + a_{23} t + a_{24}}$$

$$Y_s = \frac{a_{31} t^3 + a_{32} t^2 + a_{33} t + a_{34}}{a_{41} t^3 + a_{42} t^2 + a_{43} t + a_{44}}$$

These expressions differ from those used by Robert's curve drawing display (5,6) in that cubics rather than quadratics are used and there are separate denominators for X and Y. If connected by short straight line segments, the points generated in this way can adequately represent a curve. The family of curves that can be generated includes all of the conic sections. It also includes a wide variety of curves with inflection points, such as are shown in Figure 14. The matrix multiplier and clipping divider described in this paper can be used to generate such curves in a few hundred microseconds.

Although it is easy to see how the curve drawing system operates, it is not so easy to find the matrix which corresponds to a given desired curve. The mathematics for finding this matrix is more complicated than would be appropriate to discuss here. Suffice it to say that the matrix required to draw a particular desired curve can be found from many alternative geometric specifications. A curve may be specified by the position and tangent direction at its beginning and end, and by the requirement that it pass through two additional points. Alternatively, the curve may be made tangent to specified lines at its ends and be forced to pass with a given slope through a single additional point.

Methods are available for manipulating the matrices which specify the curves (7,8). For example, suppose a particular matrix draws a particular curve from point P through point Q to point R. We might wish to partition the curve at point Q so as to draw it in two sections, each identical in shape to part of the original curve. Multiplying the full-curve matrix by a selected partitioning matrix as shown in Figure 15 will produce the matrix required for the corresponding part.

THE WARNOCK HIDDEN LINE ALGORITHM

John Warnock at the University of Utah has recently invented a new algorithm for solving the hidden line problem (9). The computation time required by the Warnock Algorithm grows more slowly with picture complexity than has ever been the case before. The Warnock Algorithm breaks the picture down into successively smaller "windows" within which the solid objects are examined. If there is nothing of interest within a particular window it need not be further subdivided. If, however, the picture within a certain window happens to be very complex, that window will be subdivided for more detailed examination. Ten levels of binary subdivision will, of course, suffice to produce pictures with a resolution of 1024 lines.

The basic operation of the Warnock Algorithm is to detect whether any edge of a polygon passes through a window. If no edge of the polygon passes through the window, Warnock's program must detect whether the polygon surrounds the window or lies entirely outside the window. The clipping divider described in this paper does this basic operation of detecting whether an edge passes through a window very quickly. In many cases, the line can be trivially rejected as outside the window after only five add times. If the window is fairly large, a midpoint of the line may fall within the window after only three or four more add times, or less than two microseconds. The full clipping process is necessary only if the line just nicks the corner of the window.

A special mode has been provided in the clipping divider for use with the Warnock Algorithm. In this mode, the clipping divider merely announces whether or not a particular line passes through the window; it does not bother to compute the intersections of the line with the edges of the window, nor to transform the resulting information into scope coordinates. Additional equipment is provided in the clipping divider to detect whether or not a sequence of lines surrounds the window. We believe that with the clipping divider hardware, Warnock Algorithm programs can be written which will do the hidden line computation in real time.

CONCLUSIONS

Use of a clipping divider makes a fundamental improvement in the logical characteristics of a display. With a clipping divider, a display system can be thought of by its programmer as capable of presenting a magnified image of any portion of a very large picture. The programmer can be entirely free of the bit-packing or resolution difficulties all too common in conventional displays. The picture itself can be represented in the coordinate system most convenient to the computer. For instance, in a computer with a 24 bit word, the coordinates of the endpoints of lines can be represented with the full resolution available in 24 bits. There is no need to pack or unpack information into a "display file". He merely specifies the left edge, the right edge, the bottom edge, and the top edge of his window in the same coordinate system used for representing the picture; the display will present on the screen whatever part of the picture is contained within that window. The programmer needs to know about the peculiar coordinate system of the scope only to set the viewport. Because the clipping divider scales its output to be within the range specified as the viewport, display equipment with any origin convention can easily be accomodated. We believe that the freedom from size and resolution limitations, bit-packing, coordinate conversion, and separate display files provided by the clipping divider is well worth the investment required.

ACKNOWLEDGEMENTS

We would like to thank the many people who have been involved in the clipper project. Charles Seits designed the high-speed matrix multiplier and provided essential criticism of the clipper design. Ted Lee and Dan Cohen, both graduate students, have been an invaluable part of the project throughout. Lee's work on curves and surfaces, soon to be a PhD thesis, inspired in large part by Coons of MIT, is but hinted at in this paper. Cohen's imagination and thoughts about perspective presentation, "clipping," hidden-line algorithms, and other subjects have had a strong effect on this project. He stimulated invention of the midpoint algorithm and first suggested that our equipment could be used to draw curves.

REFERENCES

1. Dwyer, William, "Windows, Shields and Shading", Proceedings of the 6th Meeting of UAID, Washington, D.C., p. 258 (October 16-19, 1967).
2. Sutherland, Ivan E., "A Head-Mounted Three-Dimensional Display", Proceedings of the Fall Joint Computer Conference, 1968 (this issue).
3. Sutherland, William R., The On-Line Graphical Specification of Computer Proceedures, Massachusetts Institute of Technology Lincoln Laboratory Technical Report No. 405 (May 1966).
4. Johnson, T. E., SKETCHPAD III: Three-Dimensional Graphical Communication with a Digital Computer, MIT Electronic Systems Laboratory, Technical Memorandum ESL-TM-173 (May 1963).
5. Roberts, Lawrence G., "Conic Display Generator Using Multiplying Digital-Analog Converters", IEEE Transactions on Electronic Computers, Volume EC-16, Number 3 (June 1967).
6. Blatt, Howard, "Conic Display Generator Using Multiplying Digital/Analog Decoders", Presented at the Fall Joint Computer Conference, Anaheim, California (Fall 1967).

7. Coons, Steven A., Surfaces for Computer-Aided Design of Space Forms, Project MAC Report, MAC-TR-41, MIT (June 1967).

8. Lee, Theodore M.P., Three-Dimensional Curves and Surfaces for Computer Display, Thesis in preparation at Harvard University.

9. Warnock, John E., A Hidden Line Algorithm for Halftone Picture Representation, University of Utah Technical Report 4-5 (May 1968).

10. Adams Associates, "Computer Display Fundamentals", The Computer Display Review (Revised March 31, 1968).

FIGURE CAPTIONS

Figure 1: Magnification by looking at part of a large drawing through a small window.

Figure 2: Only material inside the pyramid is visible.

Figure 3: Multiple viewports in two and three dimensions.

Figure 4: Three end point cases.

Figure 5: If midpoint is not within the window, one half can always be rejected.

Figure 6: Simple rejection criterea for positive-slope lines.

Figure 7: Clipping in 2 Dimensions.

Figure 8: Clipping in 3 Dimensions.

Figure 9: Values of the "out code" in and around the window for positive Z (left) and negitive Z (right).

Figure 10: Hardware Configuration for Clipping.

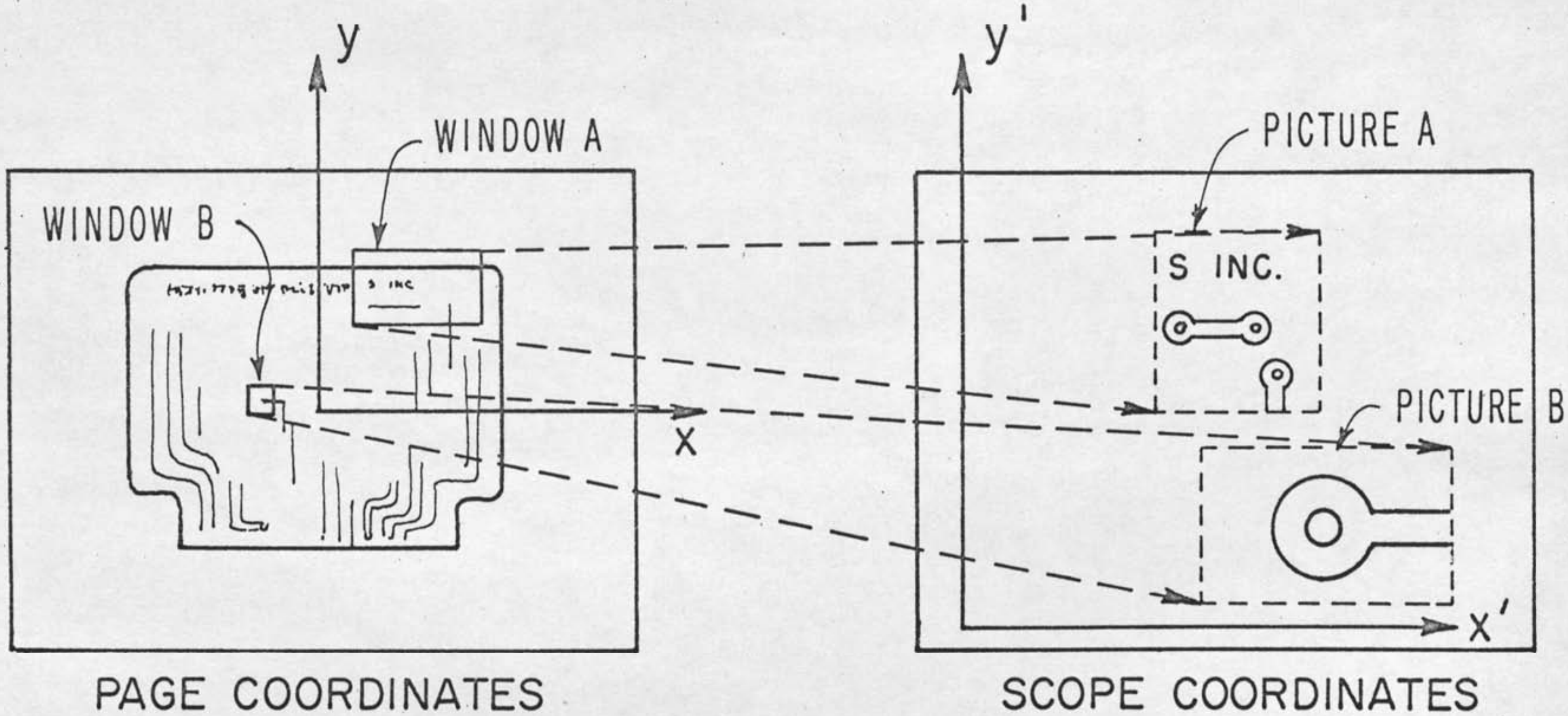


Figure 1: Magnification by looking at part of a large drawing through a small window.

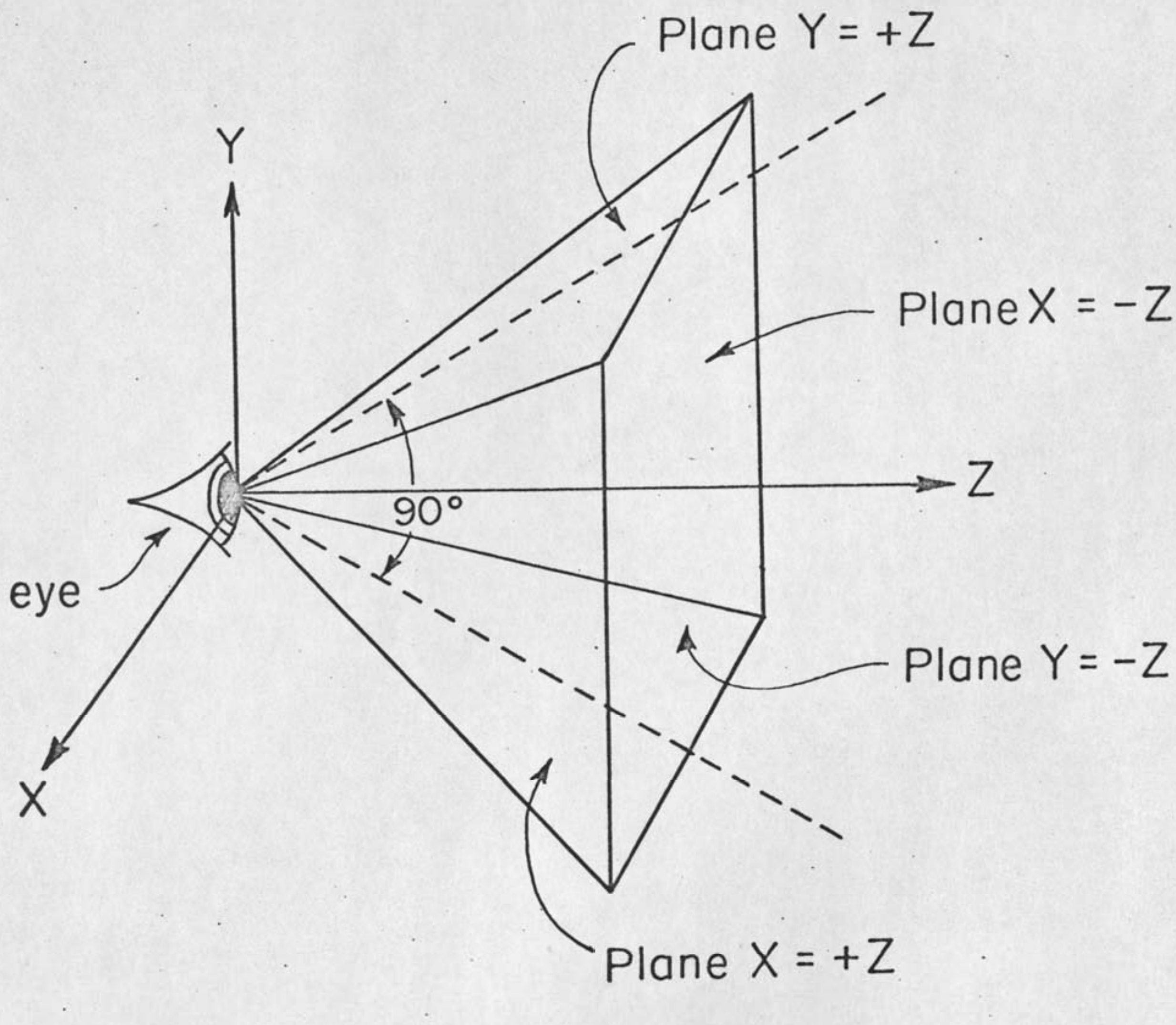


Figure 2: Only material inside the pyramid is visible.

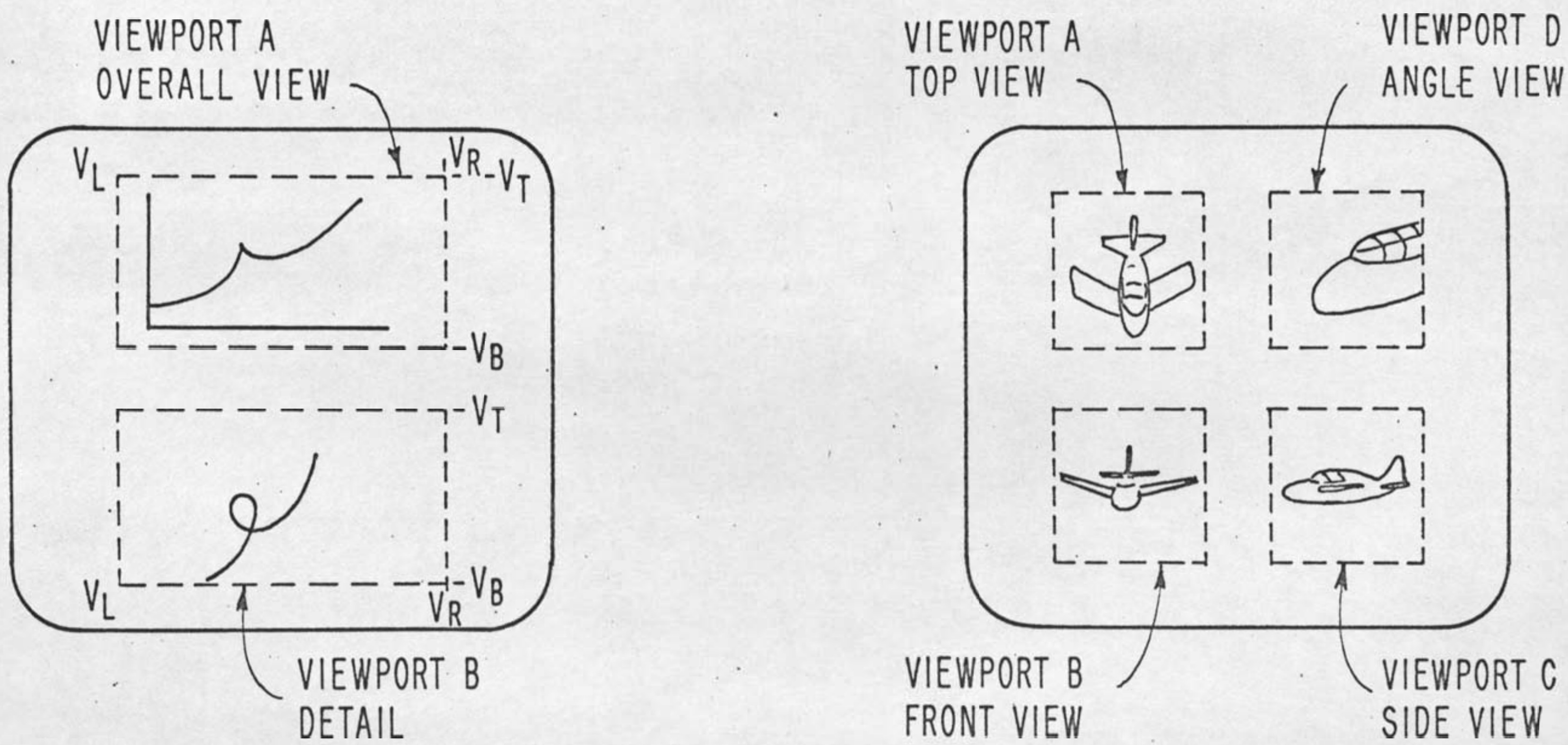


Figure 3: Multiple viewports in two and three dimensions.

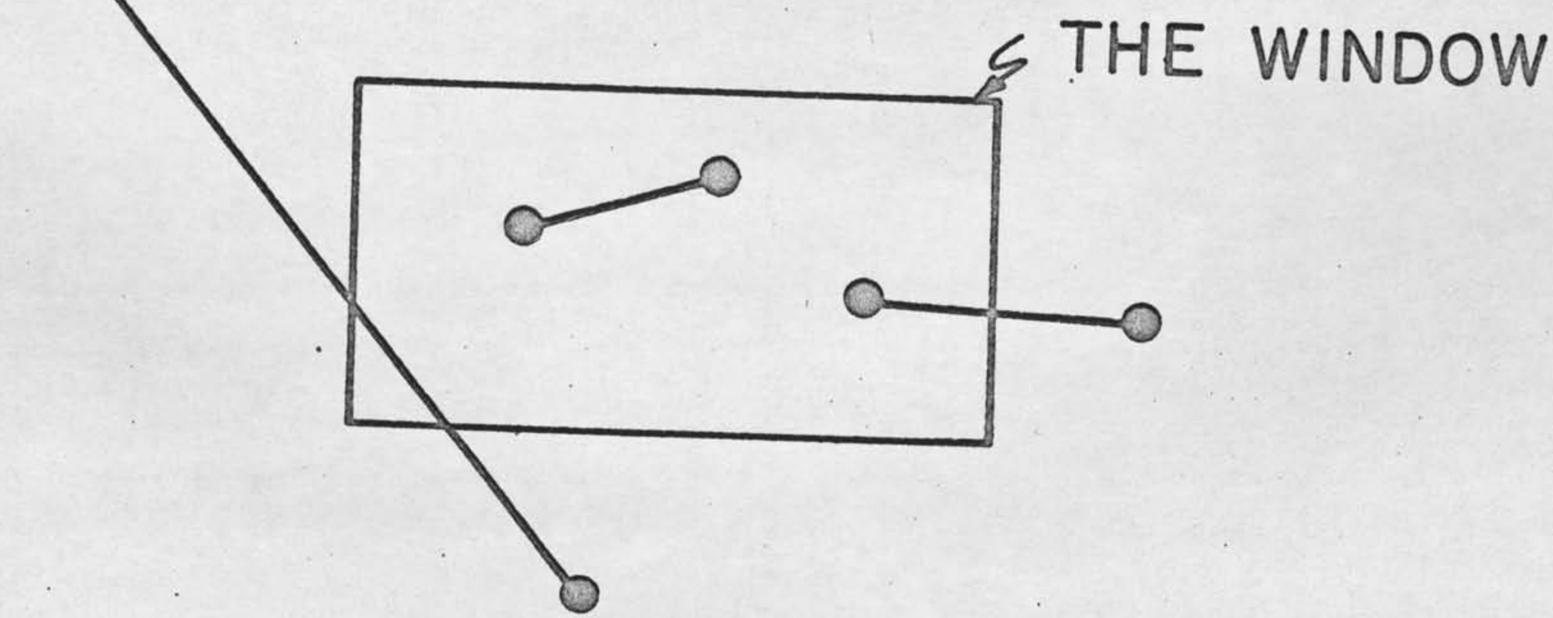


Figure 4: Three end point cases.

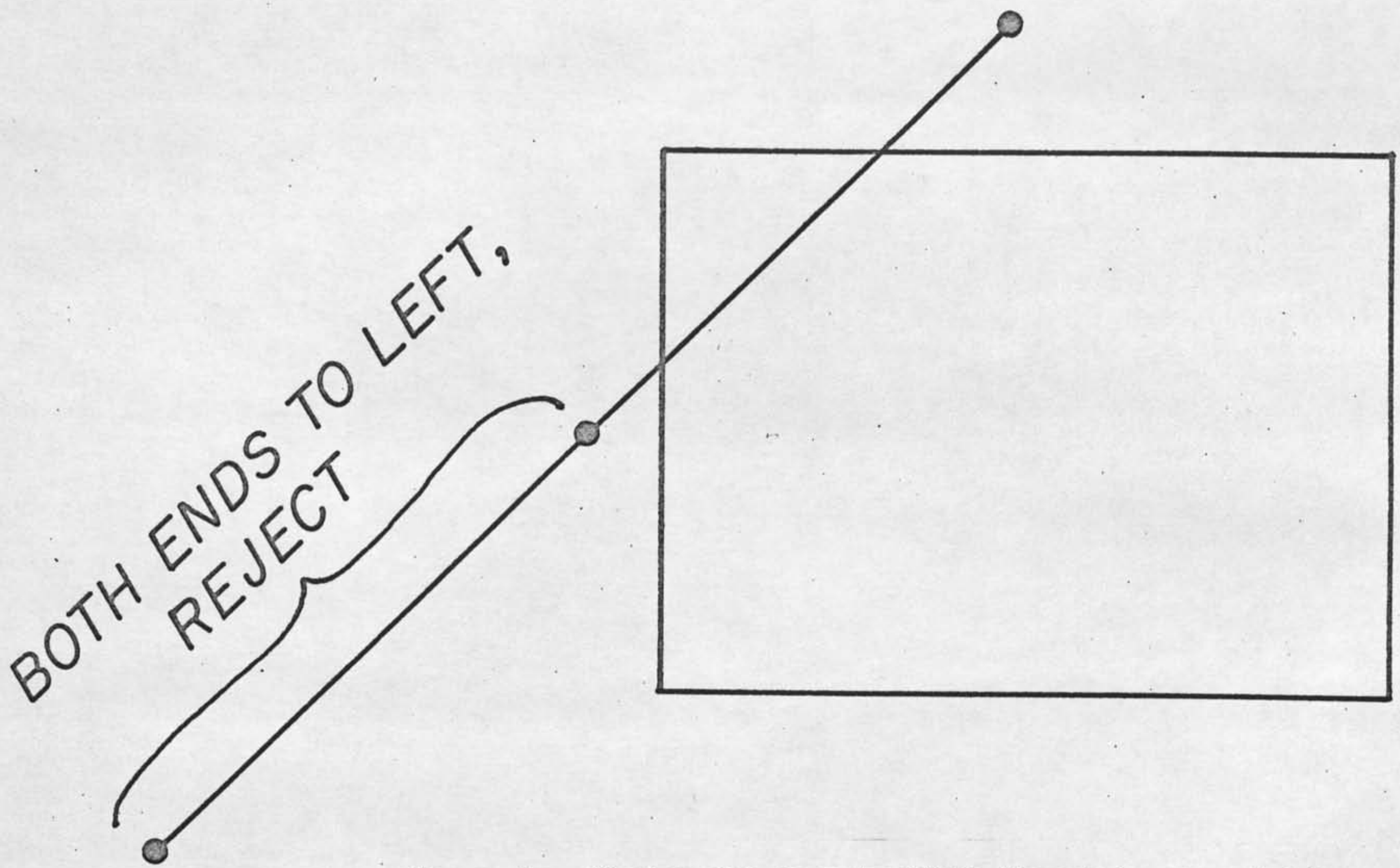


Figure 5: If midpoint is not within the window, one half can always be rejected.

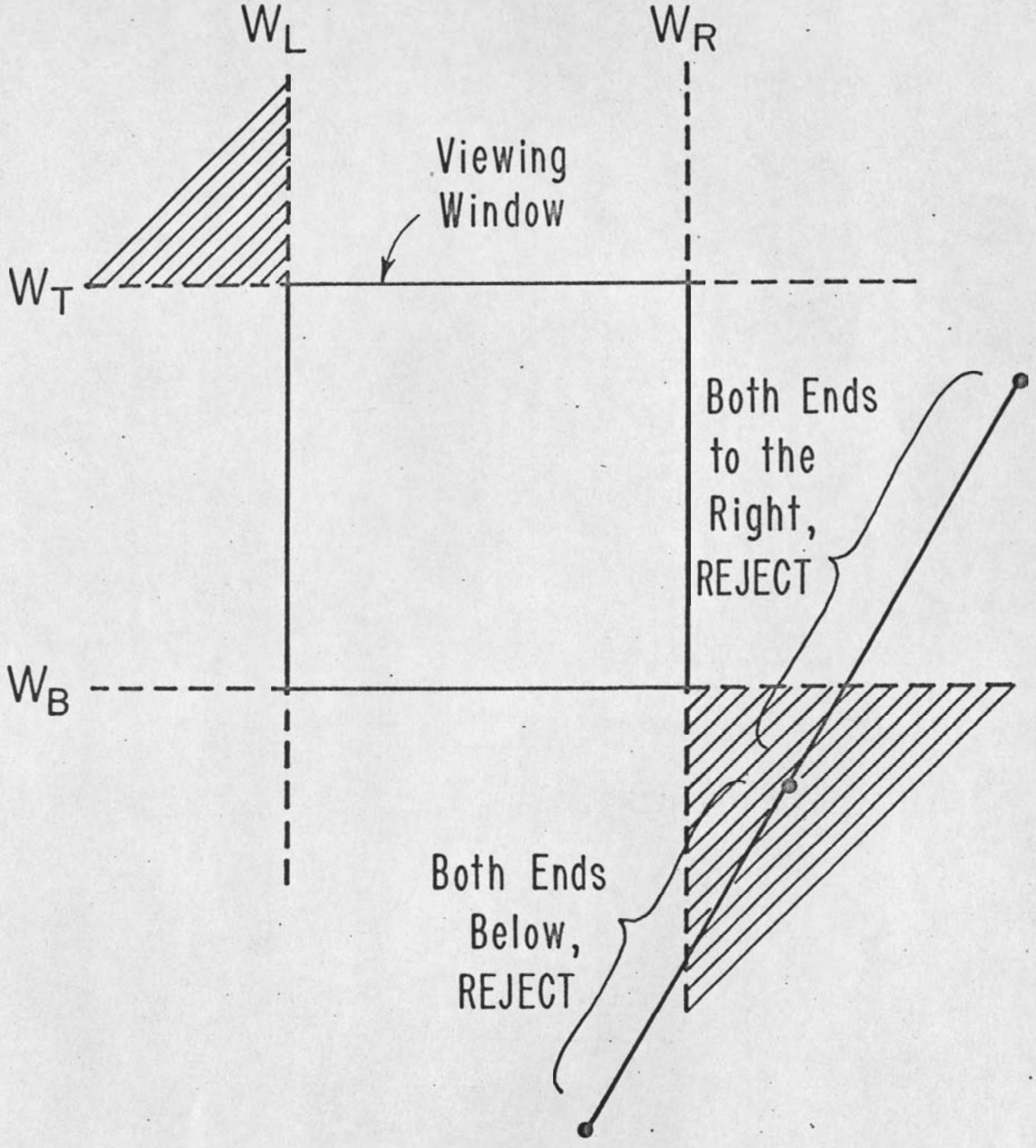
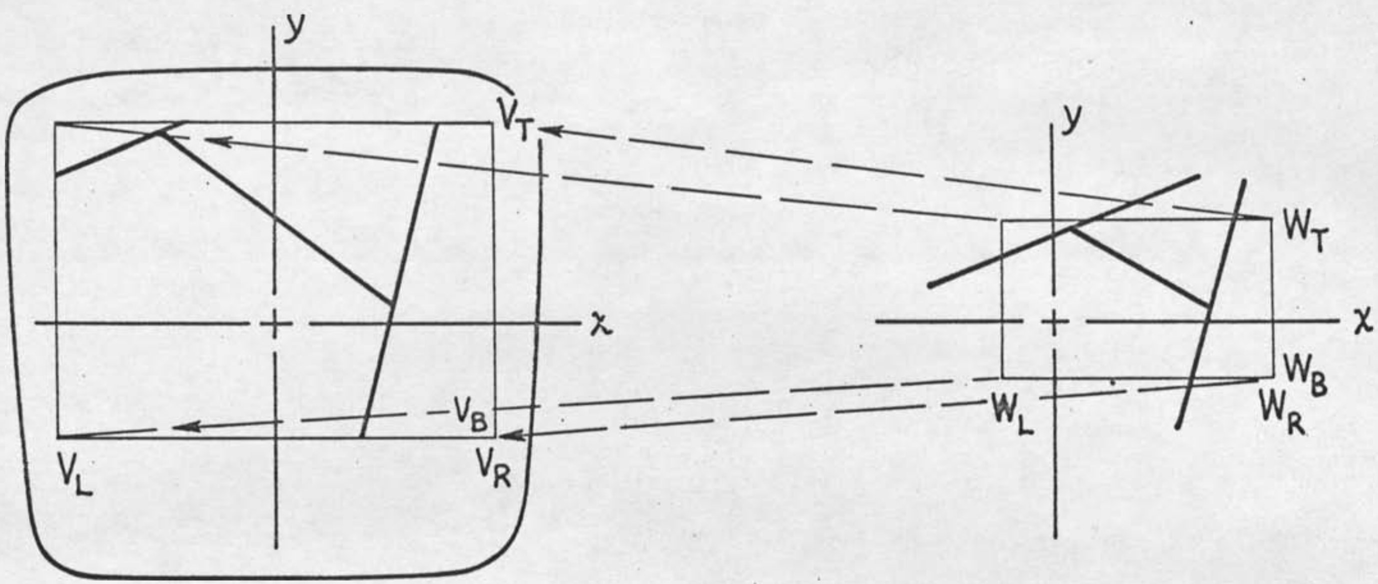


Figure 6: Simple rejection criterea for positive-slope lines.

Figure 7: CLIPPING IN 2 DIMENSIONS



SCOPE COORDINATES

$$VC_x = \frac{V_R + V_L}{2} \qquad X_S = \frac{X_P - WC_x}{WS_x} VS_x + VC_x$$

$$VC_y = \frac{V_T + V_B}{2} \qquad Y_S = \frac{Y_P - WC_y}{WS_y} VS_y + VC_y$$

$$VS_x = \frac{V_R - V_L}{2}$$

$$VS_y = \frac{V_T - V_B}{2}$$

PAGE COORDINATES

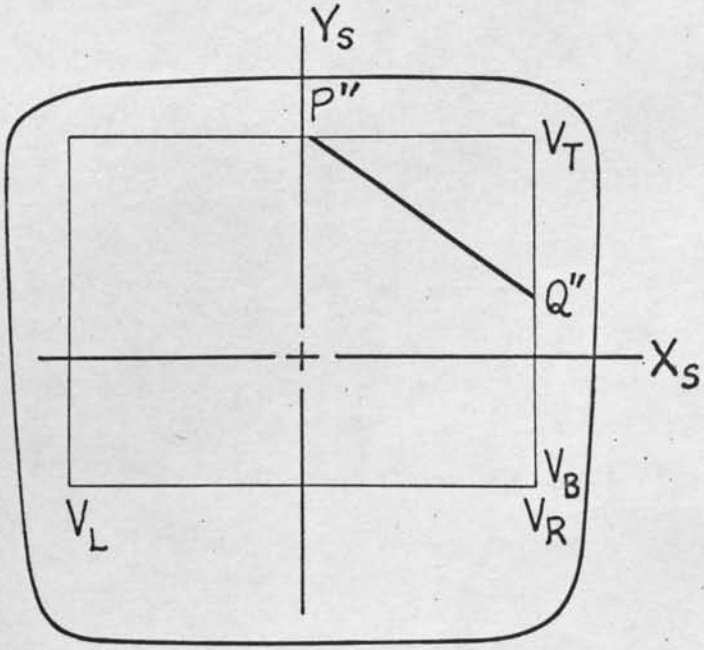
$$WC_x = \frac{W_R + W_L}{2}$$

$$WC_y = \frac{W_T + W_B}{2}$$

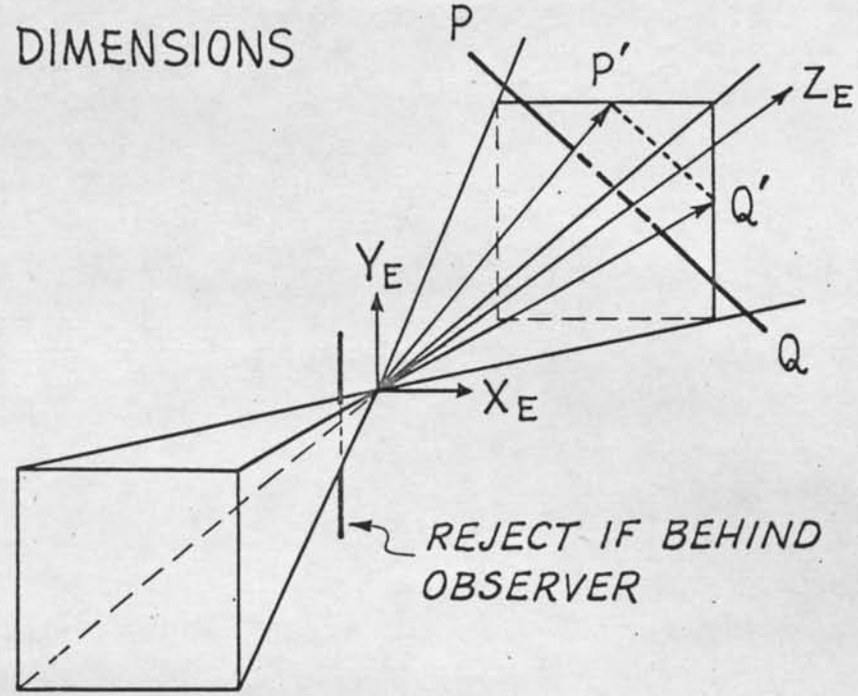
$$WS_x = \frac{W_R - W_L}{2}$$

$$WS_y = \frac{W_T + W_B}{2}$$

Figure 8: CLIPPING IN 3 DIMENSIONS



SCOPE COORDINATES



EYE COORDINATES

$$X_S = \frac{X_E}{Z_E} VS_x + VC_x$$

$$Y_S = \frac{Y_E}{Z_E} VS_y + VC_y$$

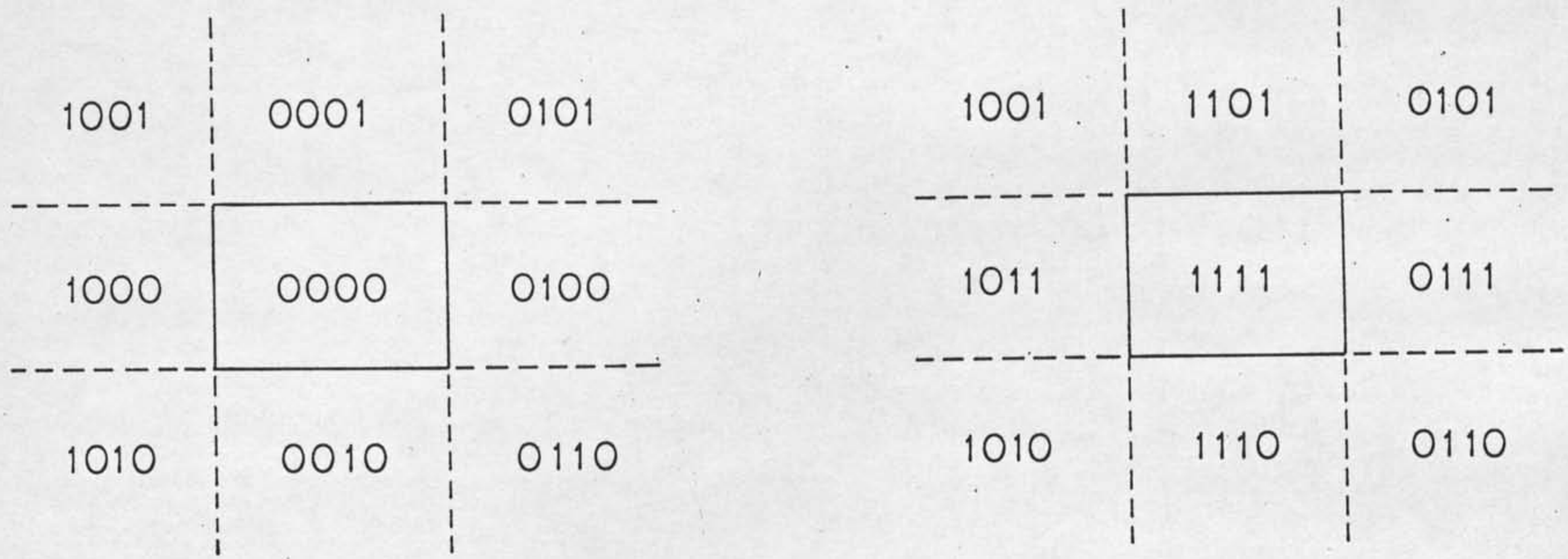


Figure 9: Values of the "out code" in and around the window for positive Z (left) and negitive Z (right).

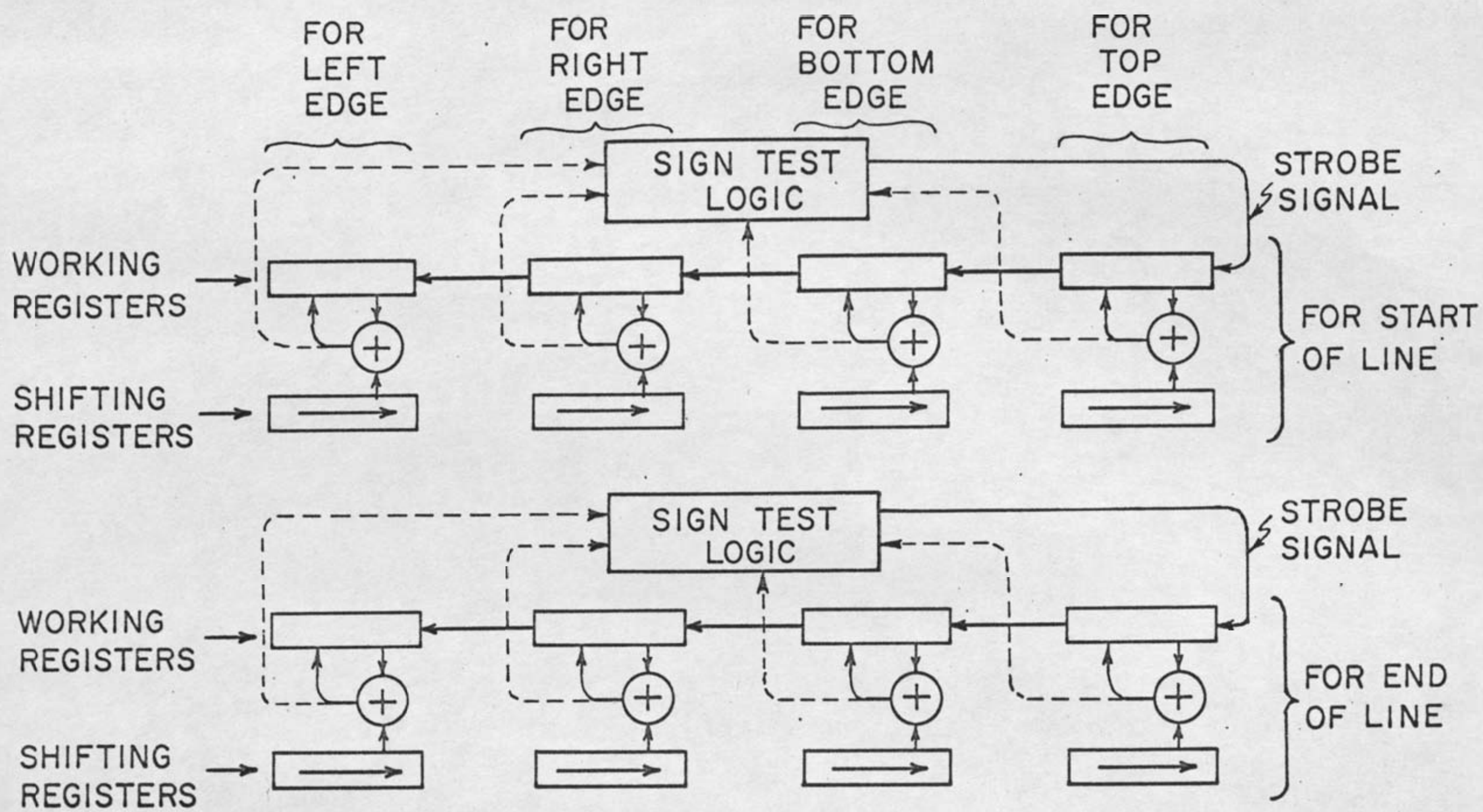


Figure 10: Hardware Configuration for Clipping.

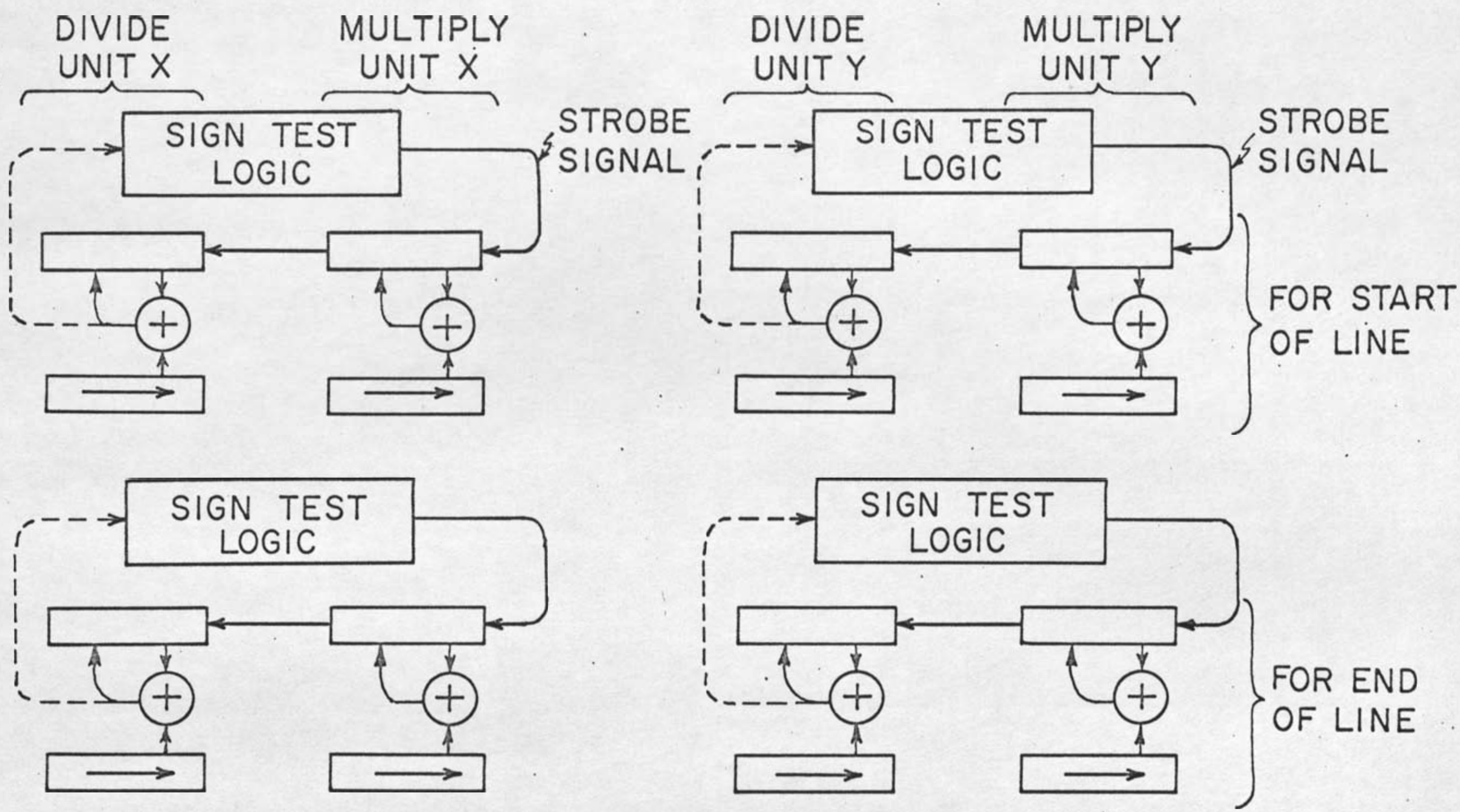


Figure 11: Hardware Configuration for Scaling.

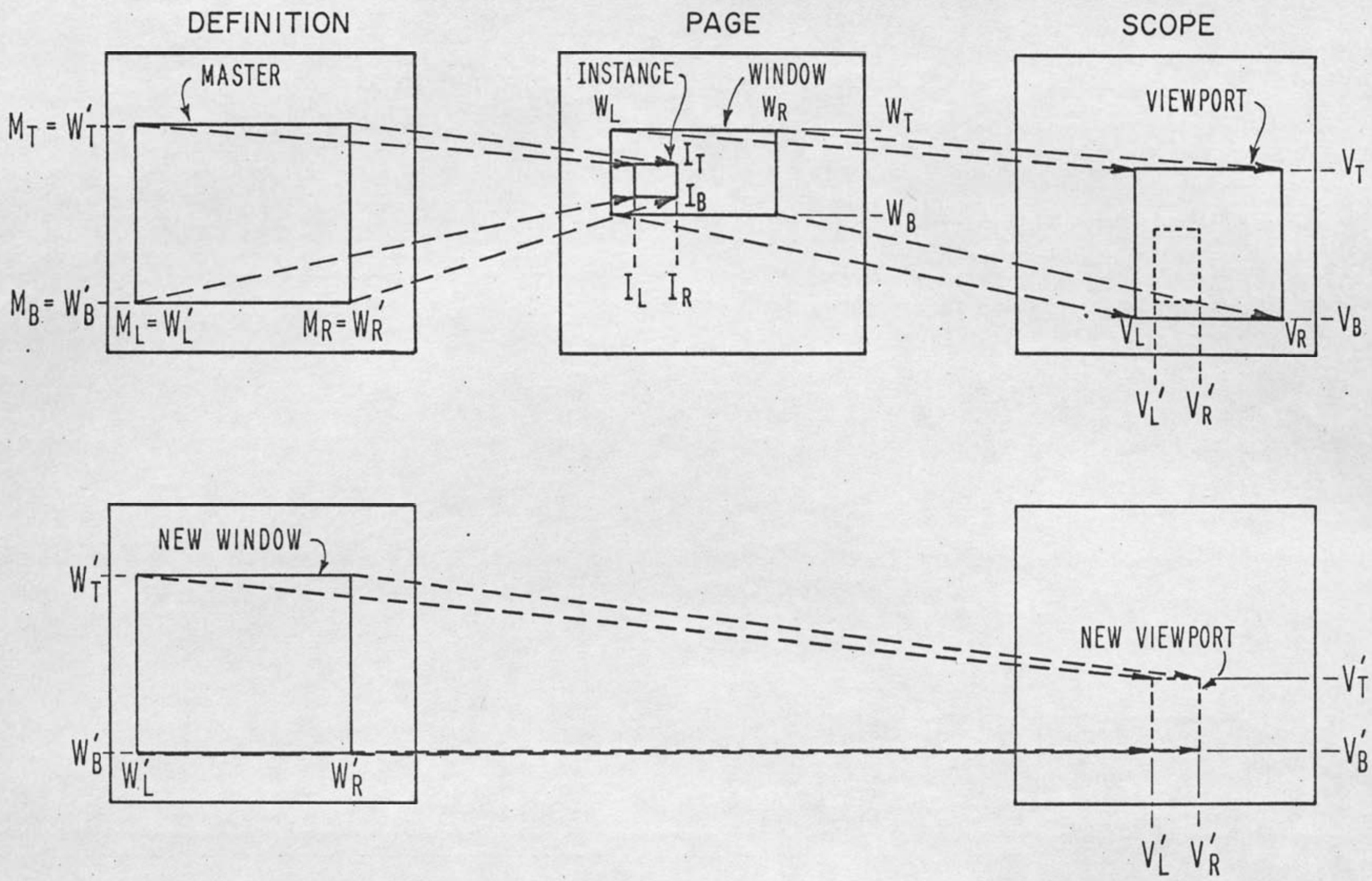


Figure 12: The two transformations for a subpicture (top) can be replaced by a single transformation (bottom).

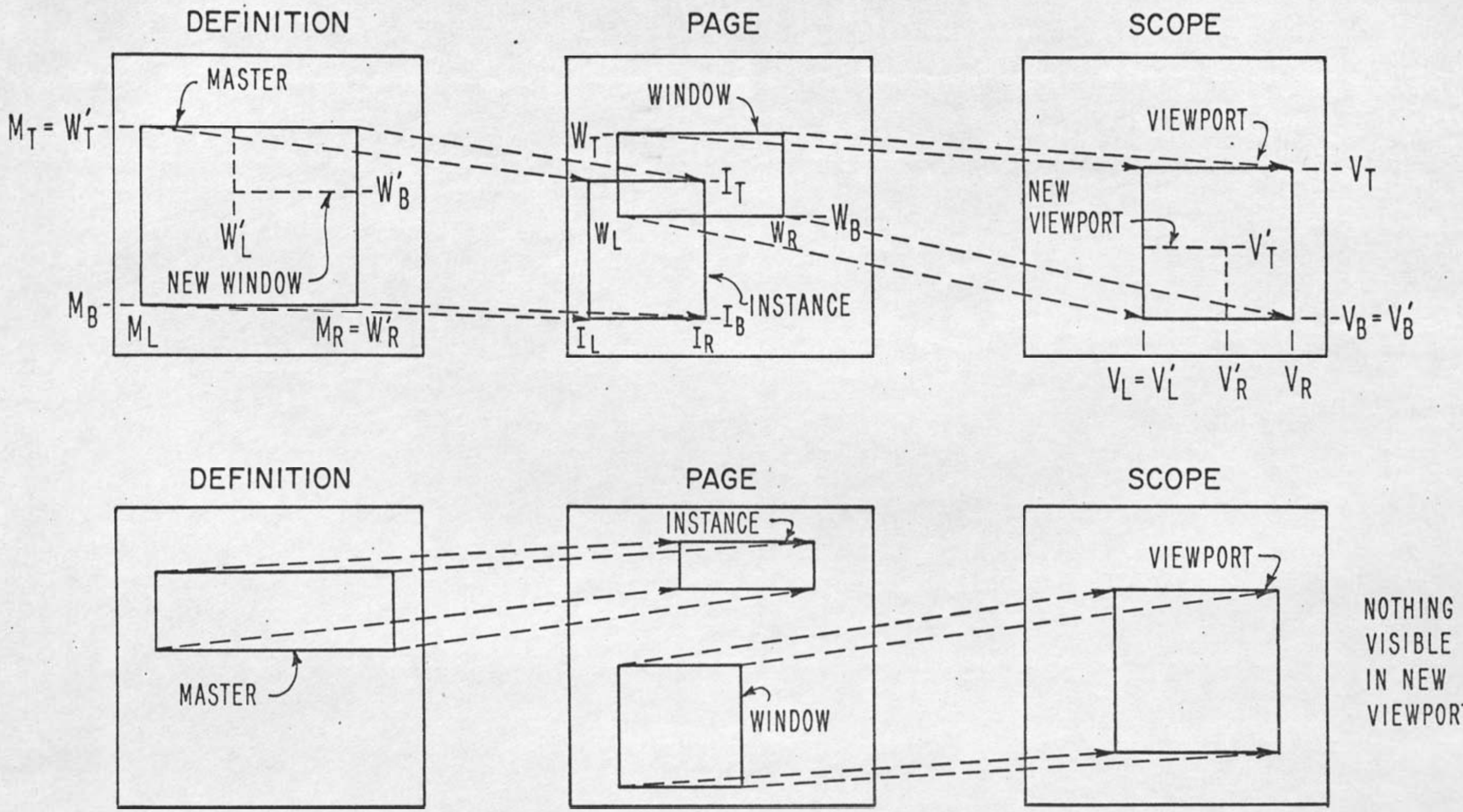


Figure 13: Finding the new window (W') and viewport (V')

from instance (I) and Master (M).

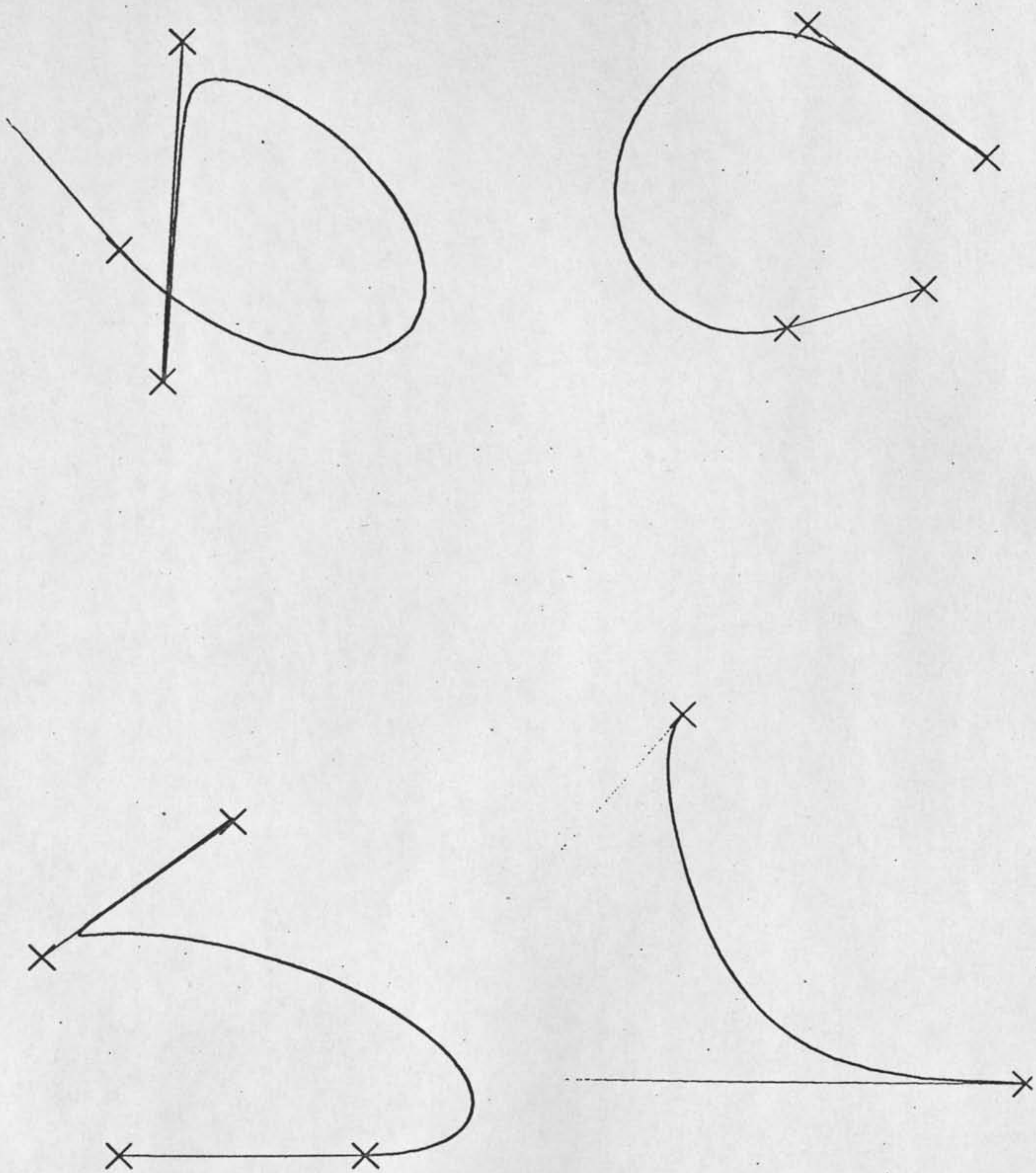
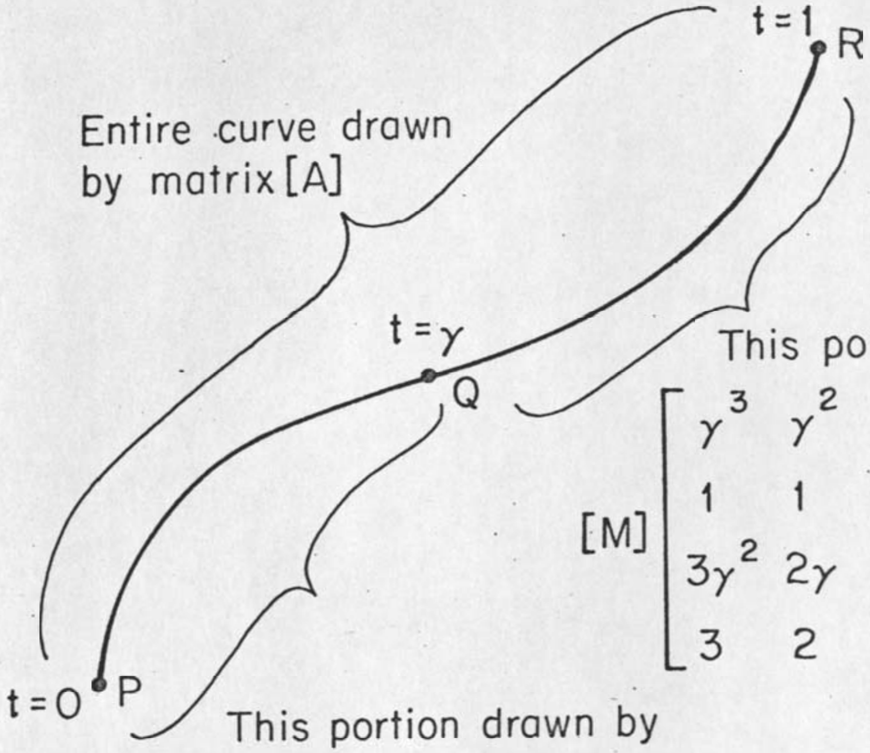


Figure 14: Some examples of curves obtainable with the equipment.



This portion drawn by

$$[M]\begin{bmatrix} \gamma^3 & \gamma^2 & \gamma & 1 \\ 1 & 1 & 1 & 1 \\ 3\gamma^2 & 2\gamma & 1 & 0 \\ 3 & 2 & 1 & 0 \end{bmatrix}[A]$$

This portion drawn by

$$[M]\begin{bmatrix} 0 & 0 & 0 & 1 \\ \gamma^3 & \gamma^2 & \gamma & 1 \\ 0 & 0 & 1 & 0 \\ 3\gamma^2 & 2\gamma & 1 & 0 \end{bmatrix}[A]$$

$$\text{where } [M] = \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Figure 15: Partitioning a curve into sections.