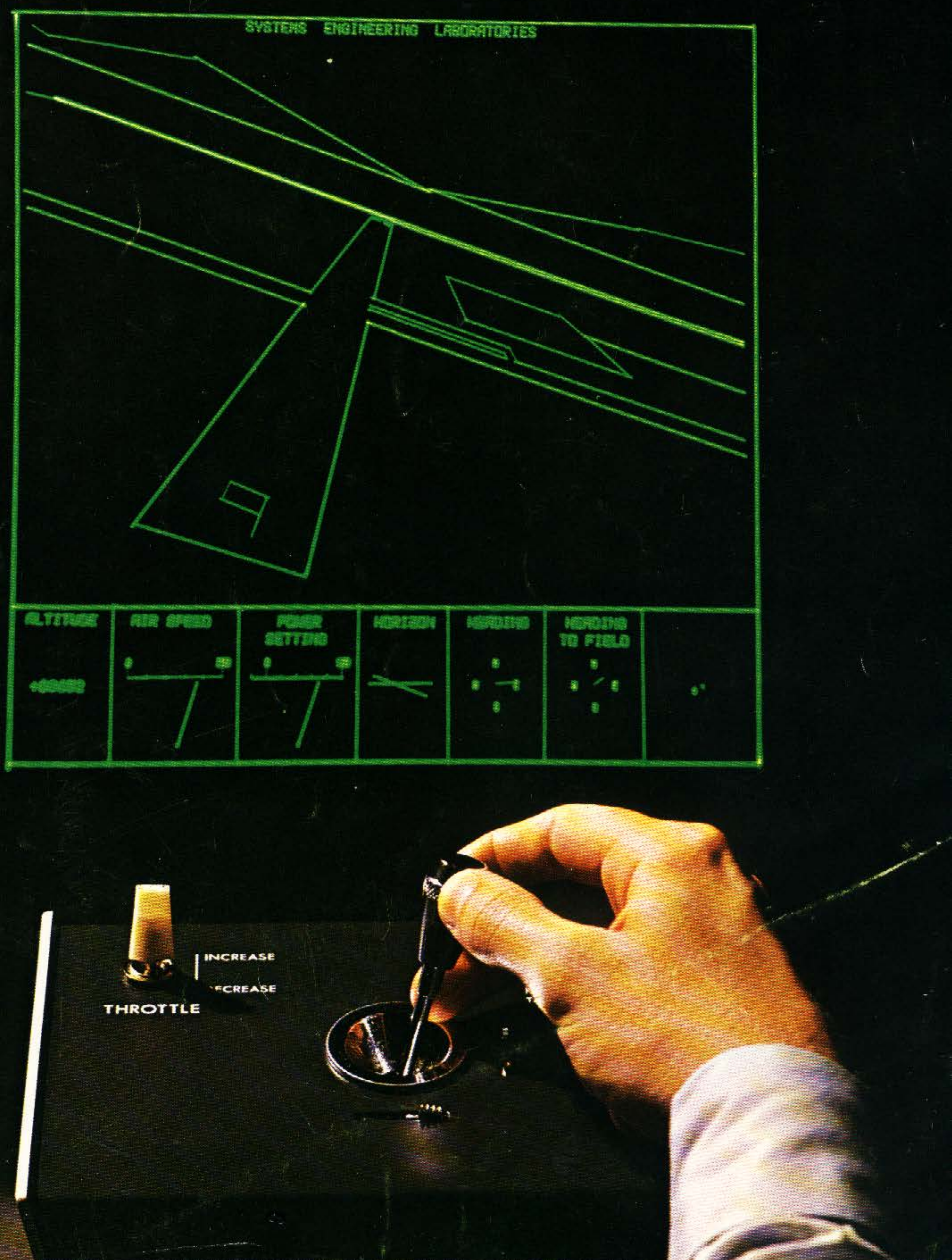
DATA PROCESSING MAGAZINE

APRIL 1968



The production of dynamic perspective views, whether for use by simulators of undersea, atmospheric or deep space vehicles, requires the efforts of both computational and systems specialists. The authors of this article typify this combination. Frank S. Greatorex, Jr., a senior systems analyst at Adams Associates, has long been involved with graphic applications. Dan Cohen, a mathematician, is a specialist in projective geometry and computer simulation.

Producing Dynamic Perspective Views for Vehicle Simulation

By FRANK S. GREATOREX, JR. and DAN COHEN

GREATOREX

COHEN

In recent years, substantial progress has been made in the use of on-line cathode ray tube (CRT) display consoles capable of presenting line drawings as well as alphanumeric characters for engineering and scientific purposes. However, although many application techniques have been developed and hardware improvements made, display units have not yet become an everyday tool of designers. Instead, the majority of units delivered to date are being used in conjunction with graphic research projects.

A number of these research projects involve the rapid dynamic calculation of perspective views, which can then be displayed to give the impression of movement; for example, attempts to evaluate prospective highway designs by displaying a view of the highway and surrounding terrain as seen by the driver of a moving vehicle.

Figure 1. The Systems Engineering Laboratories Airplane Simulator in Operation at the 1967 Spring Joint Computer Conference.

This article examines a similar application of such research and one which is also representative of the field in general.

Early in 1967, Systems Engineering Laboratories, of Fort Lauderdale, Florida, asked Adams Associates to assist them in preparing a computer-driven display demonstration for the Spring Joint Computer Conference in Atlantic City in April. The demonstration was to be an airplane simulator in which a "pilot" would fly the aircraft and attempt to land it on a runway displayed on a CRT unit in front of him.[1]

The equipment used consisted of a SEL 810A computer with a 16,384-word memory (although only slightly more than 4,000 words were required for the demonstration package), a prototype version of the SEL 816A CRT unit,[2] and a pilot control box, all of which are shown in figure 1 as they appeared at the 1967 SJCC.

The 810A is a 16-bit machine with a 1.75-microsecond memory cycle, a 3.5-microsecond add time, one index register, and a cycle-stealing data channel linking it to the display. The prototype CRT console had a 1024 x 1024 display raster over a 12″ x 12″ area, required 28 microseconds to draw a line vector and 28 microseconds to display a character, and had a refresher rate of 40 cycles per second. The pilot control box contained a control stick (joystick) that caused the aircraft to dive, climb, or turn (and roll) left or right. In addition, there was a throttle lever for increasing and decreasing power.

Information Provided

The computer program provided continuously varied graphic output as a function of operator (pilot) control, entered via the joystick and throttle. A number of clearly recognizable land and man-made features were displayed in dynamic perspective for realism and visual reference, as shown in figure 2, together with a full complement of functional instruments, including an air-speed indicator, engine-power gauge, digital altimeter, heading indicator, radio direction finder, and artificial horizon.

As shown in figure 3, input from the simulated radio direction finder was displayed in two parts: first, a pointer showing the direction of the airport from the current plane position; and second, a blip on a small map showing the plane's position relative to the airport. The heading indicator appeared on the CRT as a line pointing in the direction of flight. A sound generator was also used to simulate variations in engine pitch and intensity.

Seated at the control panel, the pilot-operator was presented with a changing CRT-displayed view of the world beyond his windshield and all the instruments necessary to relate him to his airplane. A typical flight would involve taxiing to the runway, orienting the airplane into the takeoff path, and gradually applying power to bring the plane up to flight speed. All information displayed to the pilot-operator during these and subsequent events was in real-time and included both a rapidly-changing perspective of the moving landscape and dynamic displays of instruments.

Flight Simulated

In-flight maneuvering was similarly comprehensive. Stalling speeds were routinely computed on the basis of constantly varying relevant data, such as relative air speed and angle of attack. Excessive speeds, such as those resulting from full-power dives, were likewise monitored by the program. Excessive or subnormal speeds prematurely terminated the flight and displayed alarm messages such as YOU CRASHED. Contact with the ground was permitted only within assigned landing strips, and any landing or takeoff attempt which would tend to brake the airframe, such as

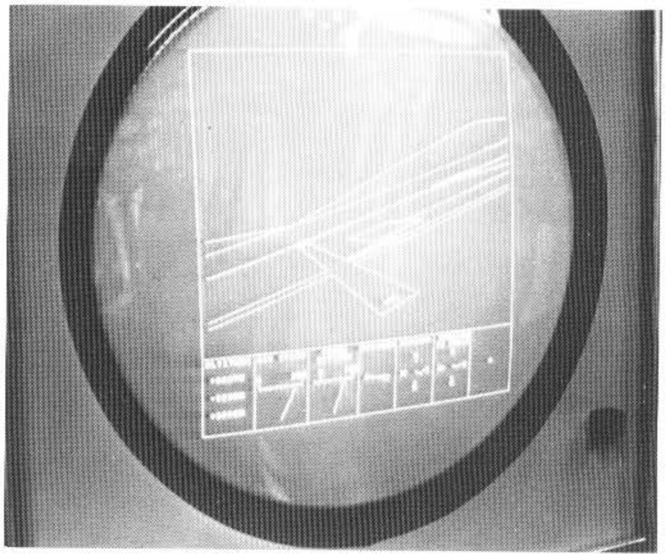
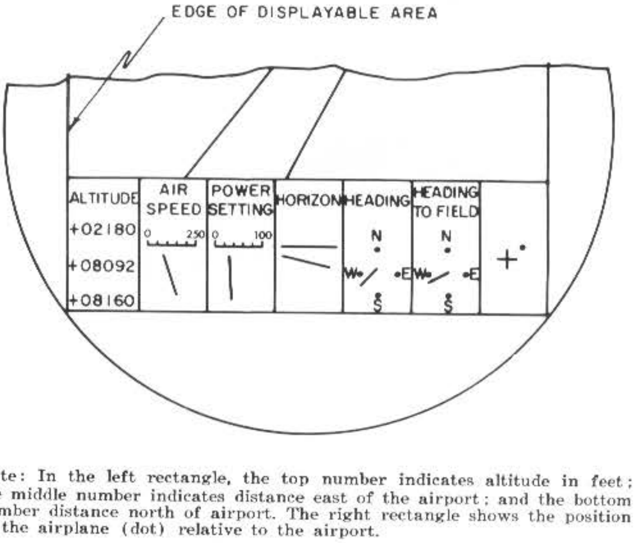


Figure 2. View of CRT Unit of Airplane Simulator



Note: In the left rectangle, the top number indicates altitude in feet; the middle number indicates distance east of the airport; and the bottom number distance north of airport. The right rectangle shows the position of the airplane (dot) relative to the airport.

Figure 3. Closeup of Displayed Controls

1. This demonstration was subsequently repeated at the 1967 Fall Joint Computer Conference in Anaheim, and will be shown again in Atlantic City this spring.—Ed.

2. The prototype equipment has since been replaced by the commercially-available SEL 816A computer graphic system.—Ed.

a wing or the nose being too low, would also terminate the flight.

In summary, the simulator allowed the pilot to take off, circle the area, orient himself to a runway, and bring the plane in for a landing. He had full control over the plane's speed and orientation, except that violent maneuvers, such as inside loops and barrel rolls, were not permitted. As well as indicating to the pilot that he had crashed, the system would tell him that he was lost if he flew outside an imaginary diamond surrounding the airport. Since the mountains were beyond the diamond, there was no problem in eliminating hidden lines from the display. This is discussed more fully later.

Indicative of the extent to which this program simulated actual flying conditions was the interesting fact that experienced pilots invariably performed better than non-pilots, even when the latter had some knowledge of the program's operation. For example, the model used at the Conference displayed an airport with two long runways for landing and takeoff, and two short ones for taxiing. While real pilots were often able to land on the short strips, non-pilots initially found it difficult to land on any of the runways.

Technical Approach

The program structure consisted of two main parts: an airplane simulator and a picture generator. The former was a simplified model for responding to external control, namely, input from the operator as provided by the throttle (engine power) and joystick (direction, pitch and roll). The simulator interrogated the control console every sixtieth of a second and read updated positional information for processing. The simulator further provided a number of aircraft parameters, including engine power, lift coefficients, response rates and maximum speed and altitude, to be easily varied to permit rapid simulation of entirely different models. Ground topography, airport design, and wind direction could also be modified to simulate almost any desired set of flight aspects and characteristics.

The picture generator interpreted data relayed from the simulator and incorporated this information with dynamic perspective procedures and pre-established parameters to create the display shown on the CRT.

Athough no attempt was made to develop a total simulation model involving the many complex differential equations necessary for thorough airframe and aerodynamic analysis, the resultant display was an accurate visual representation of real-time events entirely congruent with those of an actual pilot at the specified location and orientation. Its degree of generalization was bounded only by hardware limitations. Since the intention here is to describe computer generation of dynamic perspective views, the remainder of this article will be concerned only with the display procedure once the aircraft position and orientation have been determined.

The ground features of the basic display model are shown on the map in figure 4. A single polystring of both intensified and non-intensified line segments was used for the model. Two tables stored this polystring. The first was a table of points which included X, Y and Z coordinates. The second contained the line segment identifiers, two items being stored for each segment: the number in the point table of the segment end point, and a bit to indicate whether the segment was intensified. The segment start point was understood to be the end point of the previous segment.

The core storage of the computer was divided into four major sections: the program, the above-described tables of data-base points, and two display buffers from which display commands were fed to the CRT unit. Two buffers were used so that one could be displayed while the other was being computed. As a programing convenience, the airplane status data display at the bottom of the CRT was duplicated in both buffers.

A complete program cycle would begin with the reading of current throttle and joystick positions from the control box. These readings would then be compared with those of the various cycle to compute the airplane's position and orientation, and to update the flight data and display command tables. At the same time, the CRT and its control logic extract display commands from the other buffer. When the second buffer is ready and the display of the first is completed, the cycle is repeated after shifting the role of the two buffers.

To generate each display frame as required by the perspective model, the computer program:

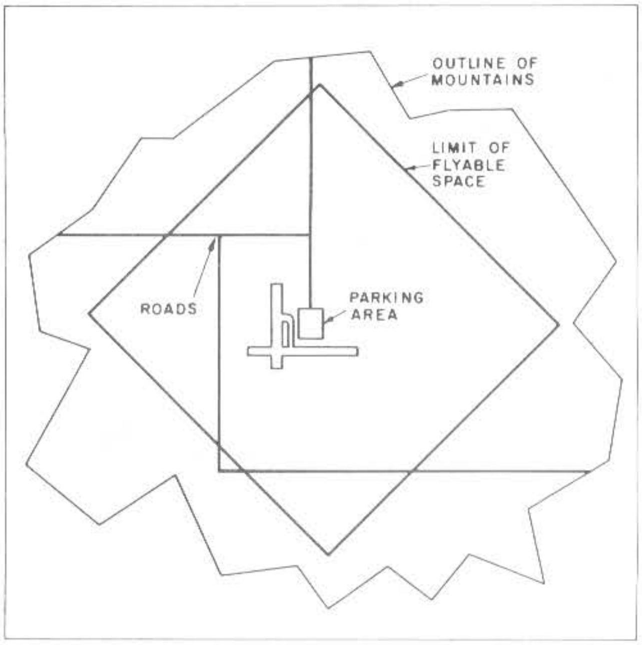


Figure 4. Map of Airport Model

1. Finds the six parameters which describe the position and attitude of the airplane. These are the three eye-position coordinates, and the flight angles for pitch, roll and azimuth.

2. Transforms the three-dimensional model to simplify projection on the viewing plane.

3. Performs three-dimensional clipping where necessary.

4. Performs two-dimensional clipping where necessary.

5. Generates the display instructions required by the hardware.

Steps 3 and 4; or 2, 3 and 4 may be combined.

The Perspective Model

Let the eye be the origin of coordinates, and let ϕ, θ and ψ be the azimuth, pitch and roll angles, respectively. The rotation needed to make the viewing axis along the X-axis, and the wings-axis along the Y-axis, can be described as a composition (product) of the following elementary rotations:

1. Rotation about the Z-axis to account for the azimuth angle, ϕ. (Figures 5A and 5B.)

2. Rotation about the Y-axis to account for the pitch angle, θ. (Figures 5B and 5C.)

3. Rotation about the X-axis to account for the roll angle, ψ. (Figures 5C and 5D.)

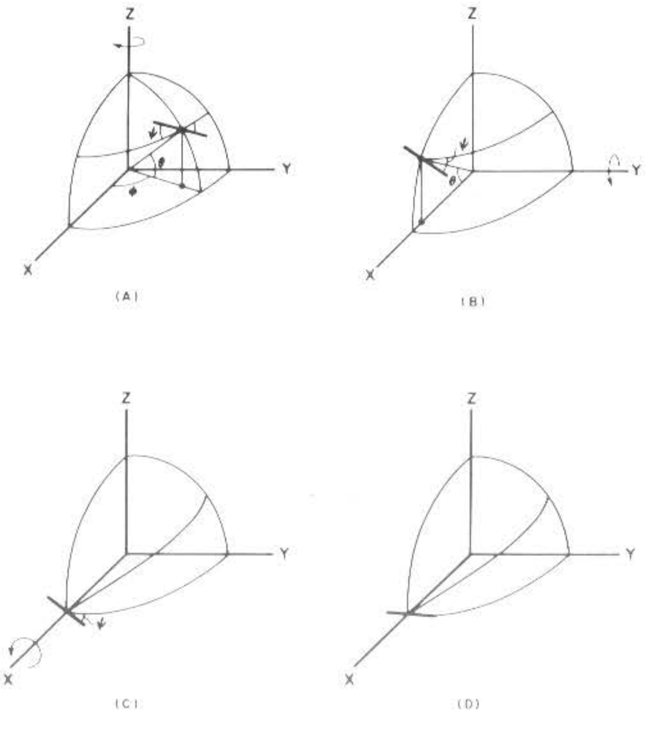


Figure 5. Elementary Rotations

These three rotations are represented respectively by the matrices T_ϕ, T_θ, and T_ψ, where

$$T_\phi = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$T_\theta = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix}$$

$$T_\psi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{pmatrix}$$

The point $\vec{P} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$ is transformed to the point $\vec{P'} = T\vec{P}$ where $T = T_\psi T_\theta T_\phi$. The values of the T-elements are found by a computational procedure described later.

As the center of the rotation is $\vec{P}_E$ (the eye position), $\vec{P}$ should be transformed to $\vec{P'} = T(\vec{P} - \vec{P}_E) = T\vec{P} - T\vec{P}_E = T\vec{P} - \vec{V}$. Note that $\vec{V}$ is common to all points, and therefore needs to be calculated only once per picture.

After this transformation, the projection of points which are in front of the viewing plane (i.e., $a_1 > 0$) becomes very simple. As shown in figure 6, $\xi_1 = -b_1 \left(\frac{D}{a_1}\right)$ and $\eta_1 = c_1 \left(\frac{D}{a_1}\right)$. If $a_1 > 0$, a three-dimensional clipping process should take place before the projection to the viewing plane.

The detailed computational procedure used to find the scope coordinates when the airplane's position and attitude are known is described below:

1. Compute a set of variables based upon the pitch, roll, and azimuth angles θ, ψ, ϕ, respectively, which can be used to compute the scope coordinates for each point on the ground:

$$t_{11} = \cos\phi\cos\theta$$
$$t_{12} = \sin\phi\cos\theta$$
$$t_{13} = \sin\theta$$

$$t_{21} = -(\cos\phi\sin\theta\sin\psi) - (\sin\phi\cos\psi)$$
$$t_{22} = -(\sin\phi\sin\theta\sin\psi) + (\cos\phi\cos\psi)$$
$$t_{23} = \cos\theta\sin\psi$$

$$t_{31} = -(\cos\phi\sin\theta\cos\psi) + (\sin\phi\sin\psi)$$
$$t_{32} = -(\sin\phi\sin\theta\cos\psi) - (\cos\phi\sin\psi)$$
$$t_{33} = \cos\theta\cos\psi$$

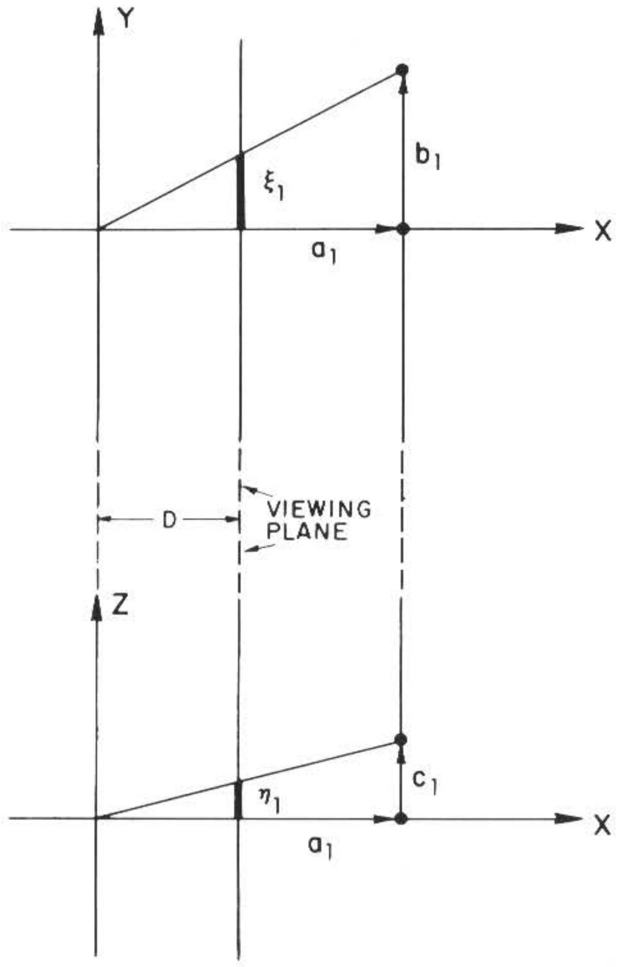


Figure 6. Projection of Lines in Front of Viewing Plane

2. Using the coordinates of the pilot's eye as XE, YE, ZE, compute the components of the vector $\vec{V}$:

$$V_1 = t_{11}\,XE + t_{12}\,YE + t_{13}\,ZE$$
$$V_2 = t_{21}\,XE + t_{22}\,YE + t_{23}\,ZE$$
$$V_3 = t_{31}\,XE + t_{32}\,YE + t_{33}\,ZE$$

3. Initialize the computation cycle by setting the previous set of calculated coordinates equal to the current set and calculating the new a_1 coordinate of the next point:

$$a_1 = t_{11}\,X_N + t_{12}\,Y_N + t_{13}\,Z_N - V_1$$

where X_N, Y_N and Z_N are the coordinates of the Nth point.

4. Test whether a_1 is in front of the windshield (i.e., $a_1 > 0$). If it is, go to step 5; otherwise, to step 11.

5. Compute the transformed b_1 and c_1 coordinates of the point:

$$b_1 = t_{21}\,X_N + t_{22}\,Y_N + t_{23}\,Z_N - V_2$$
$$c_1 = t_{31}\,X_N + t_{32}\,Y_N + t_{33}\,Z_N - V_3$$

and compute the scope coordinates:

$$\xi_1 = -b_1 \left(\frac{D}{a_1}\right)$$
$$\eta_1 = c_1 \left(\frac{D}{a_1}\right)$$

6. If the line from the previous point, M, is not intensified, go to step 10; otherwise, to step 7.

7. Test the previous point to see if it is in front of the pilot, that is, if $a_0 > 0$. If it is, go to step 9; otherwise, to step 8.

8. Compute the coordinates of where the line between the two points, M and N, comes into the sight of the pilot:

$$b_0 = t_{21}\,X_M + t_{22}\,Y_M + t_{23}\,Z_M - V_2$$
$$c_0 = t_{31}\,X_M + t_{32}\,Y_M + t_{33}\,Z_M - V_3$$
$$t = (D - a_0)/(a_1 - a_0)$$
$$\xi_0 = -b_0 - t\,(b_1 - b_0)$$
$$\eta_0 = c_0 + t\,(c_1 - c_0)$$

The reason for the correction is shown in figure 7.

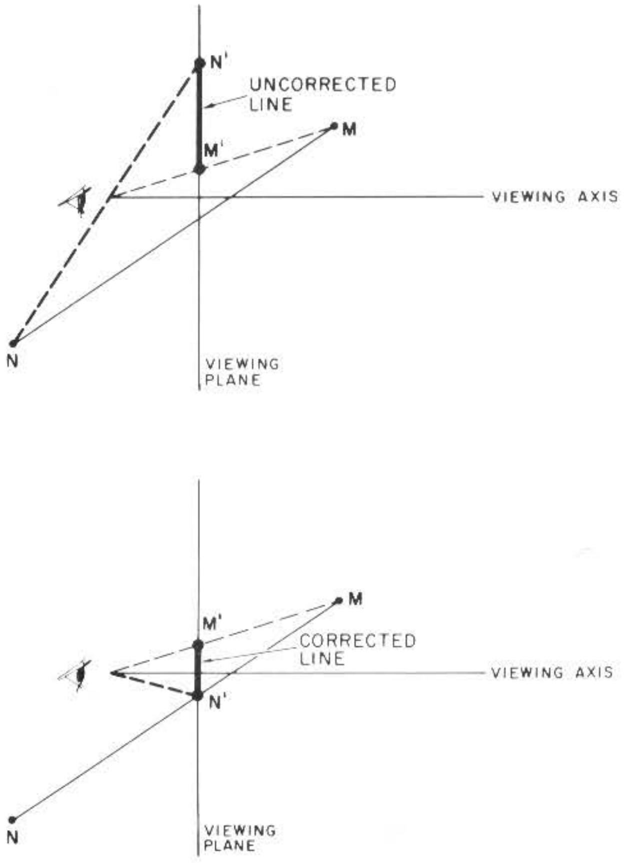
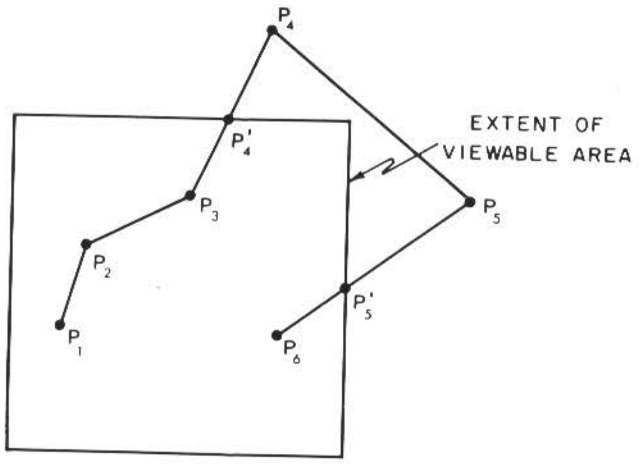


Figure 7. Correction of Lines Behind Viewing Plane

9. Knowing the start point (ξ_0, η_0) and the end point (ξ_1, η_1) on the viewing plane of a line that could be seen by the pilot, it is necessary to test whether the line fits within his area of view (effectively, his windshield). If part of the line falls outside this area, intersections with the edge of the display area are calculated as shown in figure 8. The line segments are then converted to display commands and stored in the appropriate display buffer.

10. Test to see if all line segments have been converted to display vectors. If they have, then a new cycle is initiated; otherwise, return to step 3.



Note: Clipping process calculates points P'_4 and P'_5 and determines that line P_4P_5 is not displayable.

Figure 8. Clipping Lines to Viewable Area

11. If the line from the previous point, M, is not intensified, go to step 10; otherwise, to step 12.

12. If the a_0 coordinate of the previous point is behind the windshield, go to step 10 since the entire line is out of the pilot's view; otherwise, to step 13.

13. Calculate the scope coordinates of point N:

$$b_1 = t_{21} X_N + t_{22} \quad Y_N + t_{23} Z_N - V_2$$
$$c_1 = t_{31} X_N + t_{32} \quad Y_N + t_{33} Z_N - V_3$$
$$t = (D - a_1)/(a_0 - a_1)$$
$$\xi_1 = -b_1 - t\,(b_0 - b_1)$$
$$\eta_1 = c_1 + t\,(c_0 - c_1)$$

Go to step 10.

A flow diagram describing this procedure is shown in figure 9.

The Next Steps

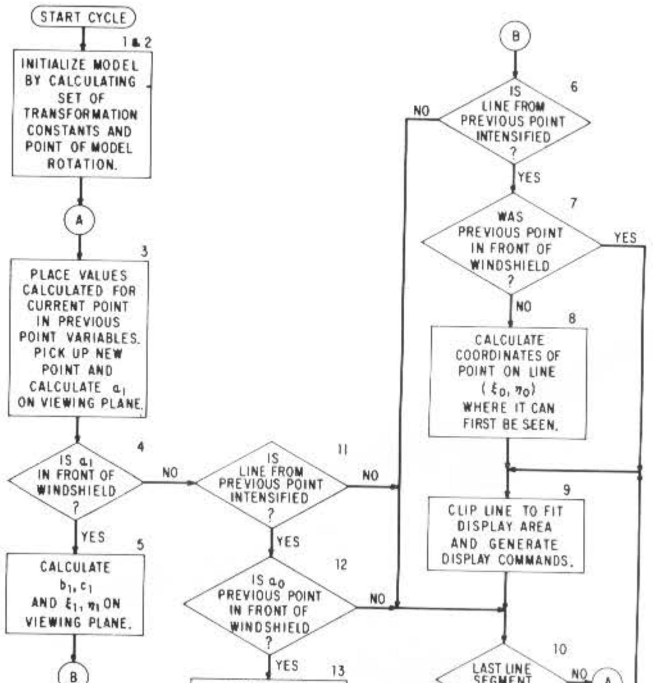
A number of steps must be taken before the airplane simulator described above could be converted into a practical tool for displaying more-detailed perspective views, such as from an automobile traveling along a highway.

The primary need will be to solve the problem of hidden-line determination and elimination. No problem existed with the airplane simulator program since all points of the airport were at zero elevation and the airplane was constrained to fly within the mountain range. In addition, the mountains had no depth to them; only their outline was used.

There are three approaches to the development of vehicle simulators having a dynamic hidden-line elimination capability: 1) the availability of faster computers and special-purpose hardware oriented to the particular mathematical procedures used, 2) a breakthrough in the mathematical procedures, and 3) the utilization of alternate display methods.

Other steps might include: 1) obtaining a digital model of the terrain, 2) developing a mathematical model of the route (possibly as output from computer design programs), 3) inputting vehicle and driver characteristics, and 4) devising a means of controlling the vehicle. While each of these functions has been performed in the past, what remains is to tie them all together with a solution to the hidden-line problem. Once this has been accomplished, design engineers will have a powerful new tool at their disposal.

The authors acknowledge with thanks the assistance of David S. Weisberg and David A. Westlake, of Adams Associates, and express their appreciation of the cooperation given by the staff of Systems Engineering Laboratories.



START CYCLE
1 & 2
INITIALIZE MODEL BY CALCULATING SET OF TRANSFORMATION CONSTANTS AND POINT OF MODEL ROTATION.
A
3
PLACE VALUES CALCULATED FOR CURRENT POINT IN PREVIOUS POINT VARIABLES. PICK UP NEW POINT AND CALCULATE a1 ON VIEWING PLANE.
4
IS a1 IN FRONT OF WINDSHIELD ?
NO
YES
5
CALCULATE b1, c1 AND ξ1, η1 ON VIEWING PLANE.
B
11
IS LINE FROM PREVIOUS POINT INTENSIFIED ?
NO
YES
12
IS a0 PREVIOUS POINT IN FRONT OF WINDSHIELD ?
NO
YES
13
CALCULATE COORDINATES ON VIEWING PLANE (ξ0, η0) WHERE LINE COMES INTO VIEW.
B
6
IS LINE FROM PREVIOUS POINT INTENSIFIED ?
NO
YES
7
WAS PREVIOUS POINT IN FRONT OF WINDSHIELD ?
YES
NO
8
CALCULATE COORDINATES OF POINT ON LINE (ξ0, η0) WHERE IT CAN FIRST BE SEEN.
9
CLIP LINE TO FIT DISPLAY AREA AND GENERATE DISPLAY COMMANDS.
10
LAST LINE SEGMENT ?
NO
A
YES
EXIT

Figure 9. Program Flow Diagram