DEPARTMENT OF MECHANICAL ENGINEERING AND MECHANICS
COLLEGE OF ENGINEERING AND TECHNOLOGY
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23529

GRAPHICS AND FLOW VISUALIZATION OF COMPUTER GENERATED FLOW FIELDS

By

M. Kathong, Graduate Research Assistant

and

S. N. Tiwari, Principal Investigator

Progress Report
(A Supplementary Report)
For the period ended July 31, 1987

Prepared for the
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23665

Under
Research Grant NCC1-68
Dr. Robert E. Smith, Technical Monitor
ACD-Computer Applications Branch

August 1987

DEPARTMENT OF MECHANICAL ENGINEERING AND MECHANICS
COLLEGE OF ENGINEERING AND TECHNOLOGY
OLD DOMINION UNIVERSITY
NORFOLK, VIRGINIA 23529

GRAPHICS AND FLOW VISUALIZATION OF COMPUTER GENERATED FLOW FIELDS

By

M. Kathong, Graduate Research Assistant

and

S. N. Tiwari, Principal Investigator

Progress Report
(A Supplementary Report)
For the period ended July 31, 1987

Prepared for the
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23665

Under
Research Grant NCC1-68
Dr. Robert E. Smith, Technical Monitor
ACD-Computer Applications Branch

Submitted by the
Old Dominion University Research Foundation
P.O. Box 6369
Norfolk, Virginia 23508

August 1987

FOREWORD

This is a supplementary progress report on the research project "Numerical Equations for Closed-Bluff Bodies," for the period ended July 31, 1987. Special attention has been directed toward "Graphics and Flow Visualization of Computer-Generated Flow Fields." This work was supported by the NASA Langley Research Center (Computer Applications Branch, Analysis and Computation Division) through the cooperative agreement NCC1-68. The cooperative agreement was monitored by Dr. Robert E. Smith of the Computer Applications Branch, Analysis and Computation Division.

GRAPHICS AND FLOW VISUALIZATION OF COMPUTER GENERATED FLOW FIELDS

By

M. Kathong[1] and S. N. Tiwari[2]

ABSTRACT

Flow field variables are visualized using color representations described on surfaces that are interpolated from computational grids and transformed to digital images. Techniques for displaying two- and three-dimensional flow field solutions are described. Several methods of illustrating flow structure are addressed. This report describes the transformations and the use of an interative graphics program for CFD flow field solutions, called PLOT3D, which runs on the color graphics IRIS Workstation. An overview of the IRIS Workstation is also described.

[1] Graduate Research Assistant, Department of Mechanical Engineering and Mechanics, Old Dominion University, Norfolk, Virginia 23529.
[2] Eminent Professor, Department of Mechanical Engineering and Mechanics, Old Dominion University, Norfolk, Virginia 23529.

TABLE OF CONTENTS

LIST OF FIGURES

TABLE OF CONTENTS - (Concluded)

LIST OF FIGURES - (Concluded)

1. INTRODUCTION

Complex fluid flow fields can be simulated on large-scale computrs through the numerical solution of the governing partial differential equations and boundary conditions [1-2].* Typically, a spatial grid with well defined relationships between neighboring node points is defined on the field, and finite difference or finite element algorithms are applied to the governing equations at node points. With recent increases in supercomputer performance [3], improved numerical algorithms and the emergence of multiple grid schemes to handle complex geometries, the generation of large three-dimensional computational fluid dynamics (CFD) solutions will soon be a commonplace. It is quite apparent that a large volume of numerical data is generated in flow field simulation, and its analysis is complex. One of the major issues is the difficulty of displaying a limited and proper kind of data that will lead to a better understanding and evaluation of the flow being modeled. An effective approach to the qualitative analysis of computer-generated flow fields is the creation of digital images of the flow field variables, i.e., pressure, temperature, velocity vectors, streak lines etc., and the display of these images using color raster graphics [4-6].

The principles of creating a digital image of a flow field variable is to transform node points from the computational grid in the object space to the image plane and fill the void between the image node points with color that varies according to the magnitude of variables. The image space is discretized into a rectangular matrix of pixels [7] where each pixel is a small unit area which can be assigned a color. The first step toward the assignment of a color to a pixel is the creation of a color table which is a one to one relation between a sequence of integers in the interval between

*The numbers in brackets indicate references.

zero and the number of different colors that can be displayed. Pixels that are exterior to the grid surfaces are assigned an integer which corresponds to a background color. Pixels that are interior to the grid surfaces are assigned magnitudes of the scalar variable by linear or cubic interpolation of variables at the node points. The magnitude of the scalar variable is scaled and truncated to the interval of integers and therefore has an associated color through the assignment in the color table. Grid lines provides a visual clue of the curvature of a surface and can be displayed by setting the pixels along lines connecting node points to be the integer associated with background color or another color that is not predominant on the surface. In a similar manner velocity vectors or streak lines can be drawn on the image. The number of pixels and the number of color that can be displayed depend upon the available equipments.

Quantities describing a flow field are characterized by magnitude and direction. A variety of displays are used to extract this information. Scalar quantities such as pressure, density, Mach number, entropy, etc., are often displayed by profiles, carpet plots, contour lines and surfaces, and variation of color intensities. In three dimensions, color shadings are often helpful in maintaining clarity. Flow directionality can be illustrated by displaying vector fields and vector traces. Seeding particles into the flow and tracing their paths is an especially effective way of observing the direction of the flow. Information on both flow directionality and magnitude can be obtained by tracing the particle over a fixed time interval or by displaying vector fields of direction and length. In three dimensions, such displays can become confusing if too much information is shown.

This report describes the transformations and the experience with an interactive analysis and flow visualization program, PLOT3D, which works off the color graphics IRIS workstation. The guide to the use of PLOT3D is provided in Appendix A.

2. IMAGE PROCESSING TECHNIQUES

The principle of creating a digital image of a result from CFD calculations is to transform node points from the computational grid in the object space to the image plane and fill the void between the image node points with color that varies according to the magnitude of the variables. A digital image can be represented by a number of small unit areas each assigned a particular color. Each unit area is referred to as a picture element or a pixel. Each pixel is described by its address in a two-dimensional array and by an integer number. The integer represents one of the number of color that can be displayed by the Raster Display Unit (RDU). The RDU assigns a color to a pixel by mapping the associated integer into a percentage of primary colors (red, green, blue) through a color table. High resolution (visual quality) of an image can be obtained if a large number of pixels is used.

This section describes crucial steps involved in the creation of an image representing a flow field variable. Three steps are described. Section 2.1 briefly describes the transformations of an object from the object space to the image plane. Section 2.2 discusses the creation of the color table. The procedures to fill the void between the image node points with color are described in section 2.3.

2.1 Transformations

In general, an arbitrary object is described in the object space called world coordinates [7]. It is necessary to transform an object from world coordinates to the image space. For illustration, an object may be described in a three-dimensional coordinates system, but the image space is only in two-dimensional coordinates. The transformations between the object space and the image plane are either an orthographic projection or perspective view [7]. The relationship between neighboring node points is invariant under the transformations [8]. The transformations are translation, rotation and scaling. Several transformations can be combined to a single transformation.

A point in a three-dimensional cartesian coordinate system P(x,y,z) is represented by the row vector

$$\{P\} = \{\ x\ y\ z\ 1\ \} \tag{2.1}$$

If [A] is a transformation matrix, a transformed point P' is

$$\{P'\} = \{P\}\ [A] \tag{2.2}$$

The transformation matrix which translates point P to point P' is

$$[T] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ D_x & D_y & D_z & 1 \end{bmatrix} \tag{2.3}$$

where D_x, D_y and D_z are the components of translation in the x, y and z-directions, respectively.

For rotation, an axis of rotation must be specified. Rotations described here are for a left-handed cartesian coordinate system (Fig. 1). The transformation matrices that describe the rotations about the origin are

$$[R_x] = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(A) & -\sin(A) & 0 \\ 0 & \sin(A) & \cos(A) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

$$[R_y] = \begin{bmatrix} \cos(B) & 0 & \sin(B) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(B) & 0 & \cos(B) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

$$[R_z] = \begin{bmatrix} \cos(C) & -\sin(C) & 0 & 0 \\ \sin(C) & \cos(C) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.6}$$

where A, B and C are measured clockwise about the origin when looking at the origin from a point on x, y and z-axis, respectively [7].

Scaling about the origin is performed by mutiplying a point by the scaling matrix [S] defined as

$$[S] = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.7}$$

where S_x, S_y and S_z are scaling factors in the x, y and z-directions, respectively.

The rotation and/or scaling about any arbitrary point other than the origin can be performed by translating the point to the origin, performing

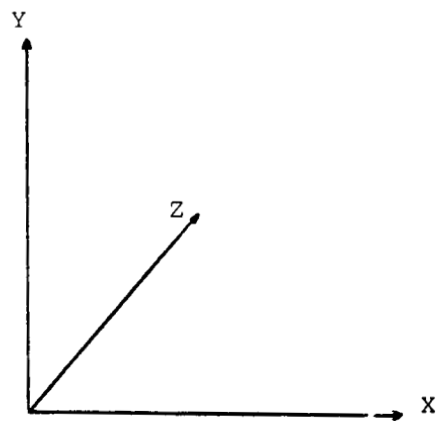

Fig 1. Left-handed Coordinate System

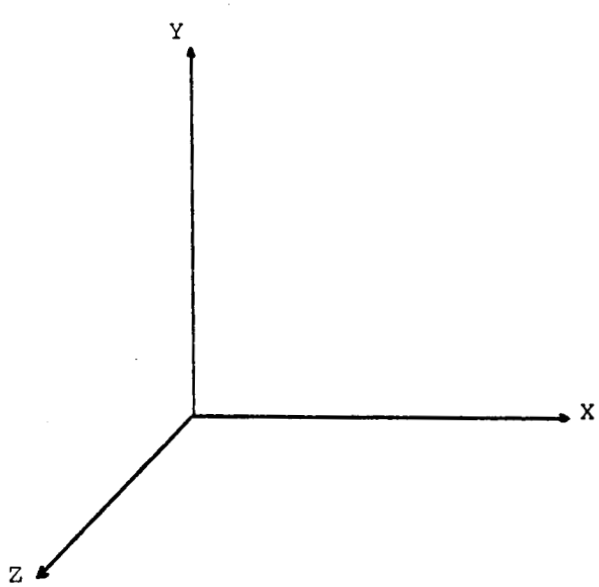

Fig 2. Right-handed Coordinate System

the rotation and/or scaling, then translating back to the original position. The right-handed coordinate system is shown in Fig. 2.

The successive applications of two or more transformations can be combined into a single transformation.

$$[T] = [t1]\ [t2]\ [t3]..... \qquad (2.8)$$

Note that the order of application of the transformations must be preserved when the transformation matrices are multiplied together.

The concept of transformations is essential for dynamics displays described in Section 4.3. These transformations are interpreted in hardware on the IRIS (See Section 3.2).

2.2 Creation of Color Table

As mentioned previously, each pixel is associated with an integer number. In order to display the integer representation in a pictorial form, it is necessary to establish a table between the integers and possible colors that can be displayed by the RDU. The color table is a useful tool for the display of a flow field variable. There are infinitely many ways of creating the color table. This section suggests one way in doing so.

A RDU assigns a color to a pixel with a component from each of the primary colors (red, green, blue). The totality of possible colors that can be displayed is thought of as an ordered set of discrete points *uniformly* spaced in a three-dimensional cartesian coordinate system (color cube) [7]. Each axis of the color cube corresponds to a primary color (Fig. 3). The main diagonal of the cube with equal amounts of each primary color represents the gray level. In order to relate a scalar variable to visual

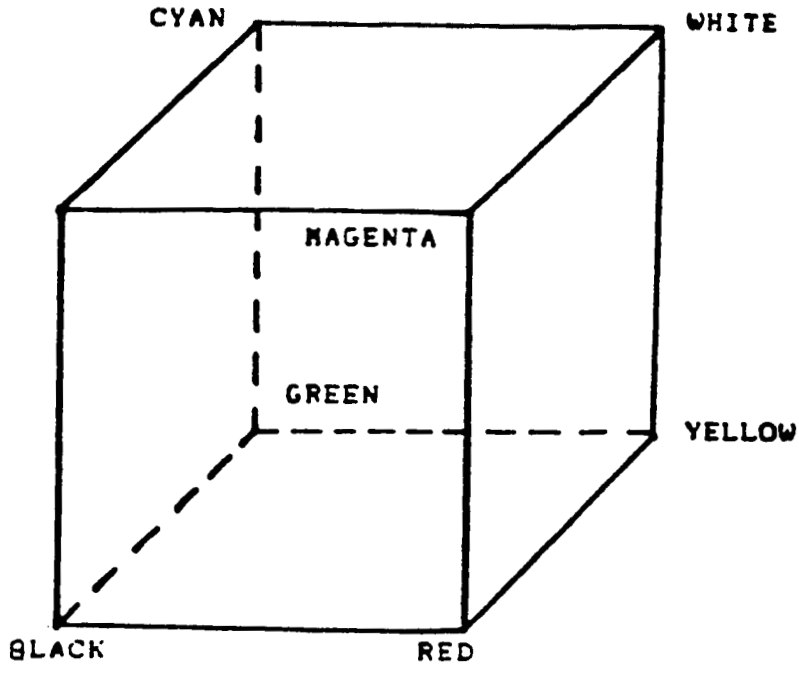

Fig. 3 Color Cube

composite of the three primary colors, a set of ordered integers is associated with a subset of the points in the color cube [9]. Figure 4 shows the color table for a 256 color scheme with integers 0-255. Seven edges and all eight vertices of the cube are mapped in uniform increments onto the 256 integers. The variable to be displayed in color is scaled between 0 and 255, and the value at a pixel location is interpolated linearly into the integer.

The color table on the IRIS contains 4096 colors (integers 0-4095).

2.3 Image Creation

This section describes a few methods in filling the void between the image node points with color after a color table has been decided upon. This filling is essential in color-contour plotting and in plotting grid surfaces in three-dimensional space.

As stated in section 2.1, the relationship between neighboring node points is invariant under the transformation. Thus, filling the void between node points in the image space will yield the correct representation of the variables in the object space. There are many ways to fill these voids. This section describes the procedures used on the IRIS in doing so.

For grid surfaces, the unit normal vectors are computed at node points by a user. The linear interpolations of the magnitude of these vectors onto the color table give the magnitude of intensities at node points. The dot product between the intensity vectors and the intensity vector emanating from a light-source yield the correct color intensities at node points. Being supplied the values of color intensities at node points, the IRIS hardware assigns a color to a polygon by the interpolation of these node point intensities. The same concept is applied to the color-contour

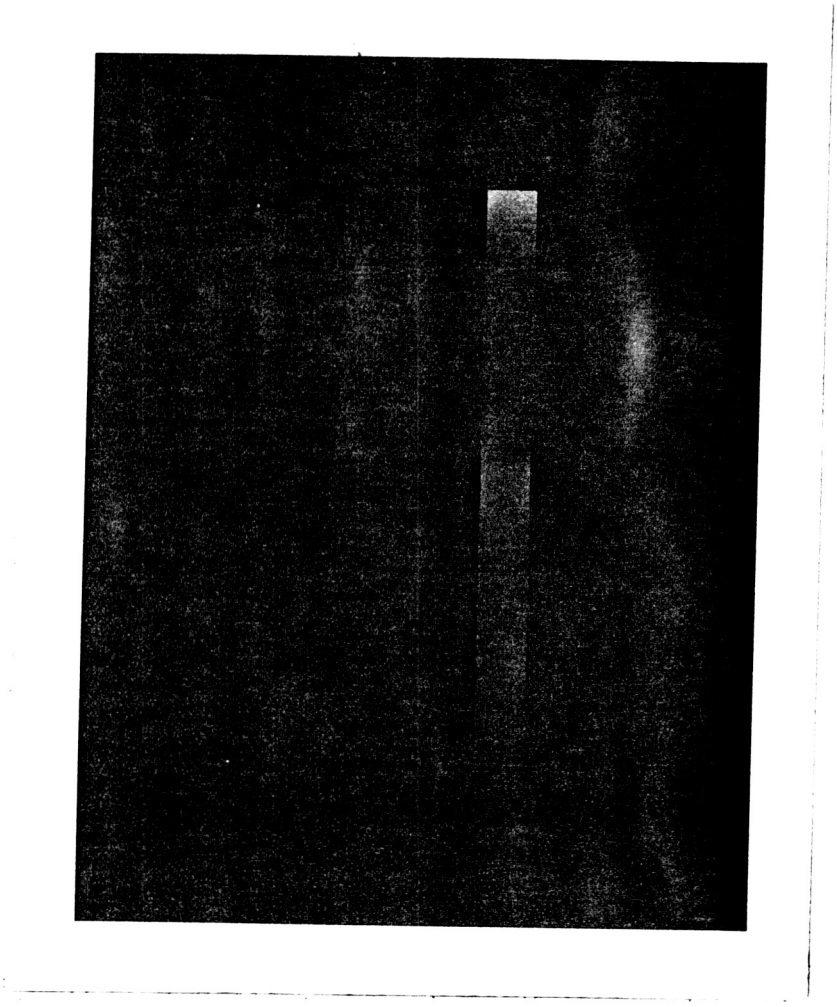

Fig. 4 Color table (256 Scheme)

plotting of the variables except that the values of the variables are used instead of the unit normal vectors. Figures 5 and 6 illustrate this concept. The hidden surfaes are obtained by plotting the furthest polygon first, then the next to the furthest and so on, until the nearest polygon.

Using a color code to describe a variable on a curved surface does not always convey to an observer the concept of curvature. Curvature is perceived by superimposing grid lines connecting visible image node points. Grid lines are displayed in a color chosen to contrast with the surface colors. In similar manner velocity vectors can be superimposed on the image of a solution variable.

3. THE IRIS WORKSTATION

3.1 Introduction

The IRIS (Integrated Raster Imaging System) is a high-performance, high resolution color computing system for 2D and 3D computer graphics. It provides a powerful set of graphics primitives (basic elements for defining a picture, consisting of moving, drawing, etc.) in a combination of custom VLSI (Very Large Scale Integration) circuit, conventional hardware, firmware, and software.

The IRIS Terminal or Workstation is a graphics system consisting of a general-purpose microprocessor, the Geometry Engine system, a raster subsystem, a high-resolution color monitor, keyboard, and graphics input devices. The IRIS Workstation runs on the UNIX Operating System, and can operate with or without a local disk. The various IRIS system configurations make it an extremely versatile high-speed graphics workstation.

The IRIS can communicate with another computer through an Ethernet interface. The heart of the IRIS is the Geometry Engine, a custom VLSI chip

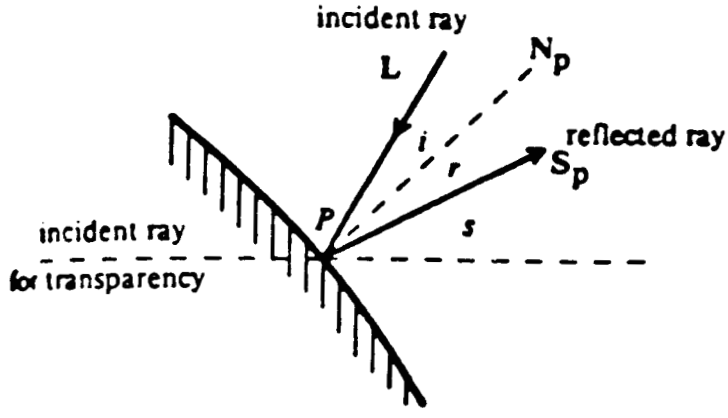

Fig 5. The shading of a point P.

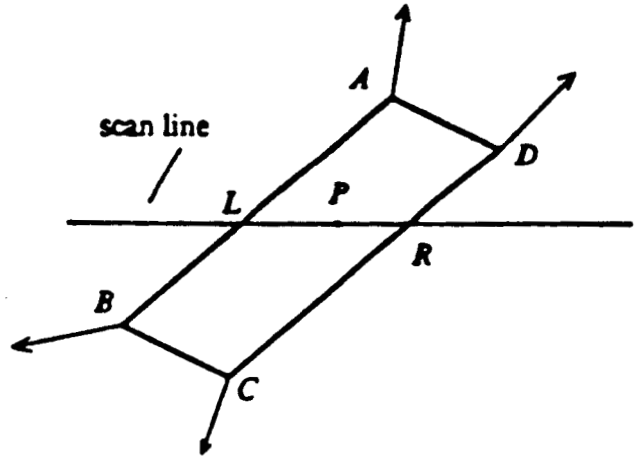

Fig 6. Gouraud shading of a polygon with vertices ABCD.

developed and patented at Stanford University. The Geometry Engine accepts points, vectors, polygons, characters, and curves in a user-defined coordinate system, transforming them to screen coordinates with arbitrary rotations, scaling, and other transformations. The Geometry Engine system performs 2D and 3D floating point transformations, clipping, and mapping of graphical data to the screen at a rate in excess of 86,300 coordinates per second.

3.2 IRIS Hardware Organization

The IRIS terminal consists of a 68000-based applications processor which controls a pipeline made up of two subsystems: a graphics transformation unit featuring a custom-designed graphics processor called the Geometry Engine and a raster subsystem that controls the display. The processor can address up to sixteen megabytes of user RAM (Random Access Memory) per process, manage display lists, interpret graphics commands and run application programs. The IRIS mouse is two valuators, one for horizontal and one for vertical motion.

The Geometry Engine (GE) is a custom integrated circuit that performed the floating point operations needed for three-dimensional transformations. The GE supports geometric primitives for drawing vectors, polygons, characters, and parametric and rational world coordinates are interpreted in hardware.

The raster subsystem receives its instruction from the GE subsystem. It scan converts vectors, rectangles, and polygons, draws characters from raster fonts and refreshes the screen. The image is generated in the available bitplanes and displayed with or without a color map. The raster subsystem features a 2903 bit-slice microprocessor and other custom hardware

that control up to 24 bitplanes of image memory. The bitplanes may be used in either singe or double buffer mode, using a 4096-entry color map to generate eight-bit intensities of red, green and blue. The color map can also be configured to be sixteen sets of a 24-bitplane system directly controls the red, green and blue values.

The raster subsystem draws vectors, areas and characters of arbitrary color. Both vectors and polygons may be generated with arbitrary, user-defined textures with no loss of speed. Vectors are drawn at the rate of three pixels per microsecond. While the setup time for a vector is six microseconds, pipelining allows setup and drawing to be overlapped. Polygons are filled at the rate of up to 44 million pixels per second.

3.3 IRIS Software Organization

The IRIS Graphics Library makes the IRIS graphics hardware available to applications programmers. This general display processing system allows an applications programmer to describe and manipulate objects in world space.

The graphics software divides a program into two processes. The applications process is written by the user and issues graphics commands. Commands are messages to the display process, which controls the IRIS graphics hardware.

Graphics commands are issued in one of two modes. In immediate mode, the commands are immediately interpreted and drawn on the screen; in display use mode, they are stored in data structures and drawn later. In both modes, the drawing commands specify coordinates in a user-defined coordinate system. Before drawing commands are interpreted by the IRIS, a viewport and a viewing window must be defined to convert these coordinates into screen

coordinates. The IRIS screen is 1024 x 768 pixels, with windowing commands assume a right-handed system with y-axis pointing upward (Fig. 2).

The applications process has access to a library of subroutines callable from a high-level programming language. Procedure calls to this library cause messages to be sent to the display process, where the graphics processing is done. The IRIS software environment supports C, FORTRAN77 and Pascal compilers which have been optimized to generate 68020 instructions.

4. INTERACTIVE FLOW FIELD DISPLAYS

Historically, a technique called flow visualization has been very important to fluid dynamics research. When performing a water tunnel test, the scientist introduces liquid dye and observes the patterns of the dye as water flow through the tank [10]. He can then visualize the fluid flow. Photographing the fluid flow is a principal diagnostic tool for wind tunnel tests. One major goal in CFD is to use computer graphics to provide the capability to visualize numerical flows. Use of interactive graphics displays in CFD research is increasing dramatically with the availability of color terminals and workstations that can perform three-dimensional perspective and viewing transformations in hardware. This capability allows the real-time rotation and translation of 3D images (graphical "object") on the display screen. It proves invaluable in visualizing and understanding the geometry as well as the resulting solutions of CFD calculations.

The interactive computer program called PLOT3D was developed and is widely used at NASA Ames Research Center. It has gained popularity at NASA Langley Research Center and at other Government, industry, and university sites as well. PLOT3D works off the IRIS Workstation. It has the capability of dynamically displaying flow field quantities resulting from CFD calculations as well as their geometries. PLOT3D is currently being

improved to handle multiple grid solution that result from using zonal grid refinement [11], or overlapped grids. Some elements of PLOT3D are described below to give an idea of current graphics display capabilities.

There is one thing worth mentioning, though, PLOT3D was developed and widely used at NASA Ames Research Center. It has not been used at NASA Langley Research Center for very long. So, there isn't a real expertise here, at NASA Langley, to consult a user when he runs into some difficulties. It took the author a few months to convince people that there was a "bug" in a version of PLOT3D used at NASA Langley, so that the person who was responsible for it would contact its creator to correct and provide the new version to NASA Langley.

4.1 Flow Field Quantitites

A CFD solution to the Euler or Navier-Stokes equations results in the values of density, momentum (three components), and energy (or other basic variables), at discrete points in the field about the geometry of interest. Additional quanitities can be calculated from these basic variables, including pressure coefficient, Mach number, temperature, and vorticity, etc. These quantities are considered to be in one of two categories: scalar or vector. Various techniques can be used to display these variables. For illustrations, most figures shown in this report were displayed using PLOT3D with the numerical solutions to the Euler equations about the fighter-aircraft configuration [12]. The flow field composes of four grid systems. This illustrates the capability of PLOT3D in dealing with multiple grid solutions.

4.1.1 Scalar Quantities

The typical type of display used for scalar quantities is contours. In two-dimensional viscous flow over an airfoil, pressure contours can be plotted to indicate the shock location and stagnation point (Fig. 7) [13]. Contours are used to indicate the value of quantity, but they also highlight large gradient regions by their convergence. In Fig. 8, contours of Mach number show the build up of the boundary layer on the airfoil, the shock, and the wake behind the airfoil. A clear and detailed picture of the flow structure is obtained from this and similar plots. In three dimensions, slices of the solution can still be plotted as contour lines. However, the 3D grid may not be nicely oriented for graphical display. An alternative way is to interpolate the solution on to a better grid for viewing. But this is not trivial, and requires attention to preserve such things as the details of the boundary layer and geometry shape from the original computation.

As a slightly different way of presenting contour information on a limited number of slices of a three-dimensional grid, color shading can be used. Here each grid cell is colored according to its average value. Color shading is used in Fig. 9 to show the density variations on the surface of the fighter aircraft configuration. High density is seen at the nose and low density region is seen near the leading edge of the wing where shock wave is formed. Figure 10 shows the same variations as in Fig. 9 but at the different view point. This results from the dynamic display capability discussed in Section 4.3. From these figures, one can clearly see the significance of color shading.

To present information throughout the field, contour lines can be plotted at many planes (slices) of the grid. Color is essential in

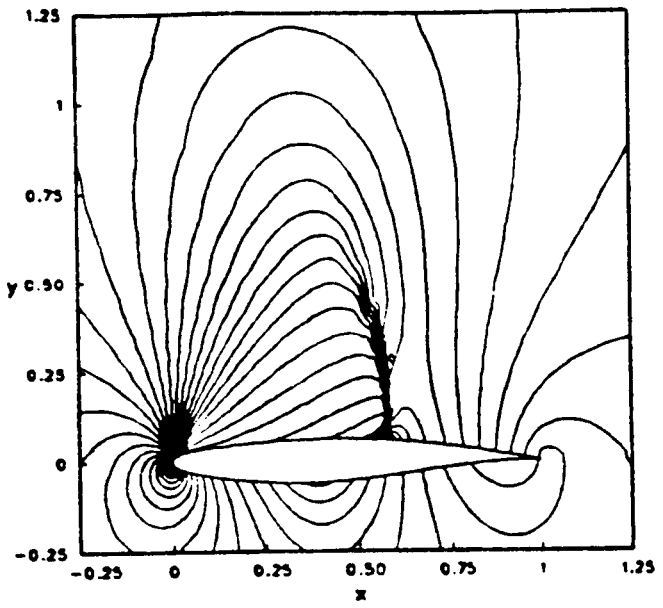

Fig 7. Pressure contours about an RAE 2822 airfoil, $M = 0.73$, $\alpha = 2.79^{o}$

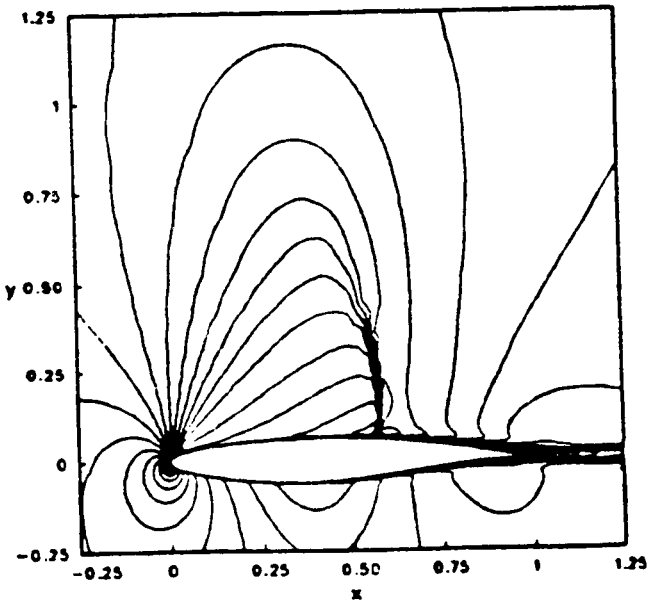

Fig 8. Mach contours for the case of Fig 7.

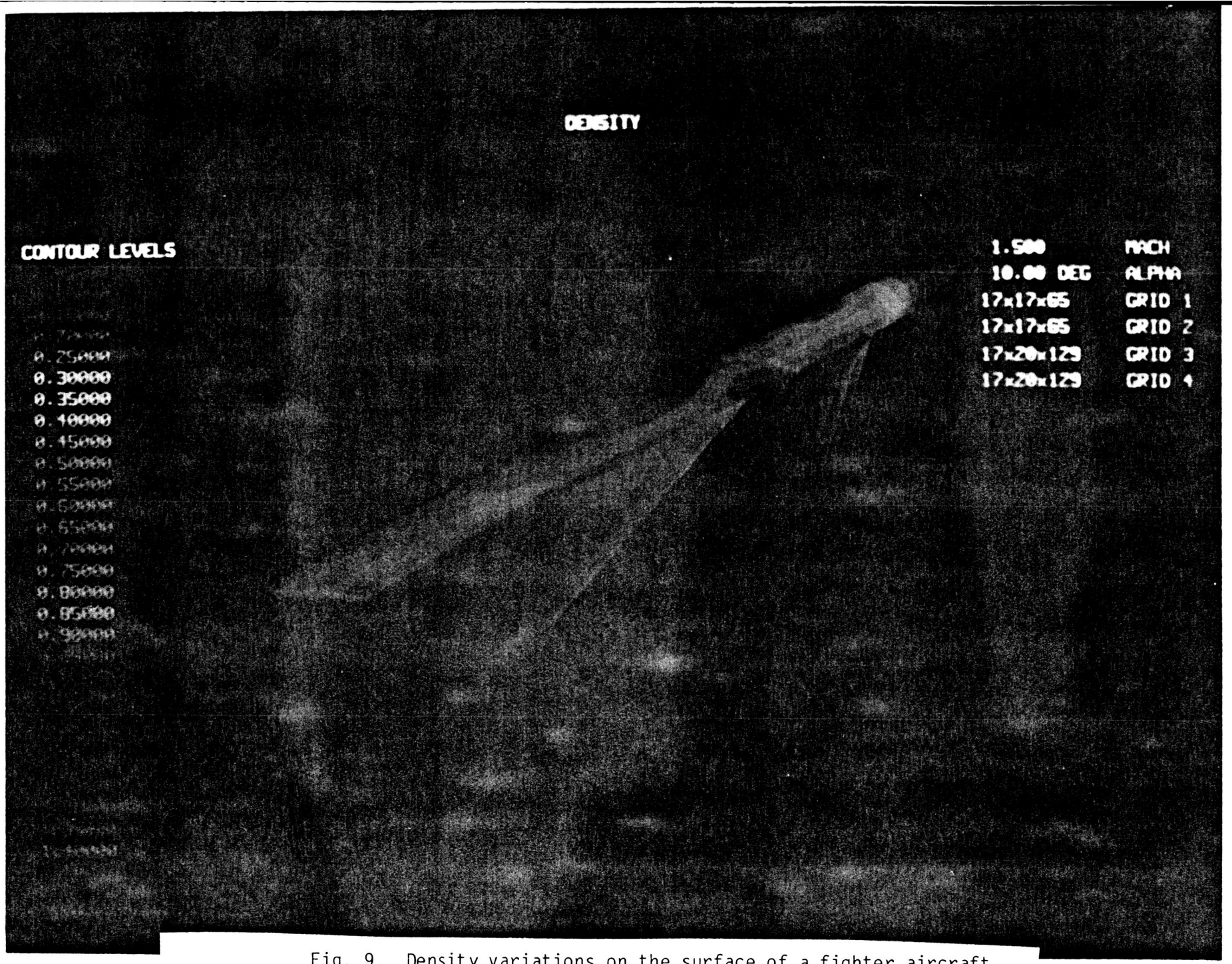

Fig. 9. Density variations on the surface of a fighter aircraft.

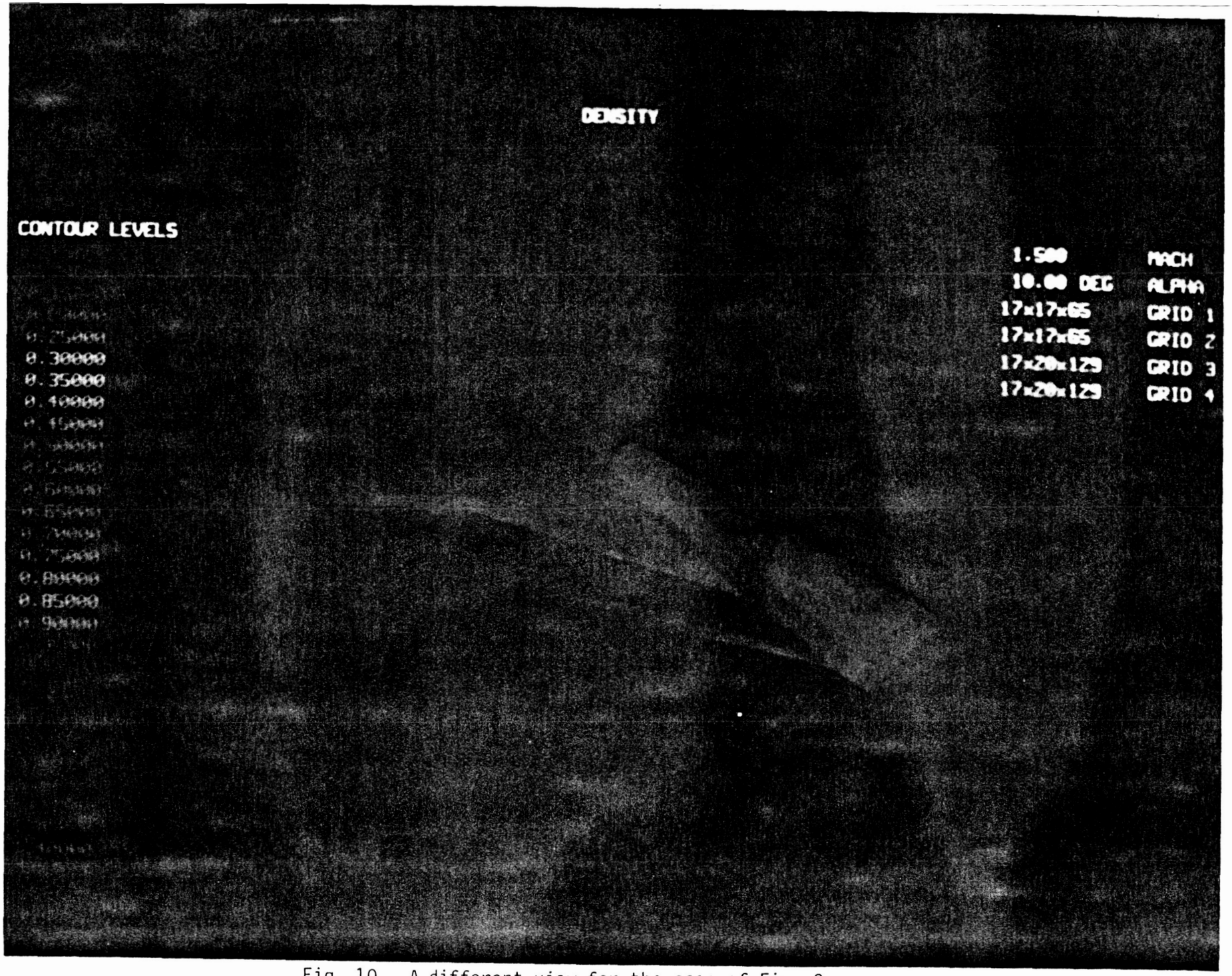

Fig. 10. A different view for the case of Fig. 9.

distinguishing the different contours. If more than a few contour levels are used, though, the plot can be too confusing to be understood.

4.1.2 Vector Fields

When displaying a vector field, both direction and magnitude must be presented. Velocity vectors are a typical example, both of a vector field, and of a displaying technique. The existence of flow separations behind the shock in Fig. 8, for instance, could be easily determined by plotting velocity vectors. In three-dimensional flows, interpretation of vector is more difficult, as only a 2D projection of the vectors can be viewed at a time. By displaying the same slice from different view points, some important information about the flow may not be detected. This can lead to a misinterpretation of results. Several flow field quantities, such as vorticity and skin friction, are vector fields in three-dimensional flows, while in two-dimensional flows these variables have only one nonzero component, and are therefore threated as scalars.

4.2 Flow Field Structure-Particle Tracing

Plots of scalar variables or vector fields often do not provide a clear picture of the structure of the flow field, including flow separations, vortices, and shocks. This can be particularly frustrating, as such features are beautifully illustrated by photographs, smoke or dye injection techniques, or oil flow patterns in wind tunnel experiments [10]. In CFD, several analogues techniques can be employed to provide the equivalent "flow visualization" for computed results. These methods have proved extremely valuable, particularly that of particle tracing.

The technique of particle tracing involves following the local tangent through the vector field. This technique has proved to be very valuable in

understanding the structures of calculated flows, and it mimics flow visualization techniques used in the wind tunnels. It tells us where the fluid is going. Figures 11 and 12 illustrate this technique from a different viewpoint. Here a group of particles are injected near the leading edge of the connard. In Fig. 13, another group of particles are injected near the leading edge of the wing. Vortices are clearly seen from these figures.

From above figures one can see that an important task when plotting particle traces is the placement of (a limited number of) particles, in such a way that they clearly show the flow structure without becoming a tangled mass of spaghetti.

4.3 Dynamic Display and Diagnostic Information

In using flow visualization techniques, the ability to rotate and zoom a three-dimensional display is extremely important. The relative positions and shape of contour lines, particle traces, or grid features could not be evaluated without this capability. Most figures in this report are shown at different viewpoints for the same variables. These figures resulted from the dynamic display capability. The significance of this capability is revealed.

The chief use of graphics has been to provide understanding of flow field phenomena and the effects of various configurational changes. In CFD, however, graphics displays are also used extensively to assess the accuracy of a calculation. Graphics displays are also used when developing new codes or algorithms to look for errors, inconsistencies, or improper modeling. Sometimes graphics provides the only practical way of detecting errors in a complex code. For example, graphics is the only way to detect whether a complex multiple grid system is created correctly. Figure 14 shows grid

Fig. 11. Particle traces for flow over an aircraft.

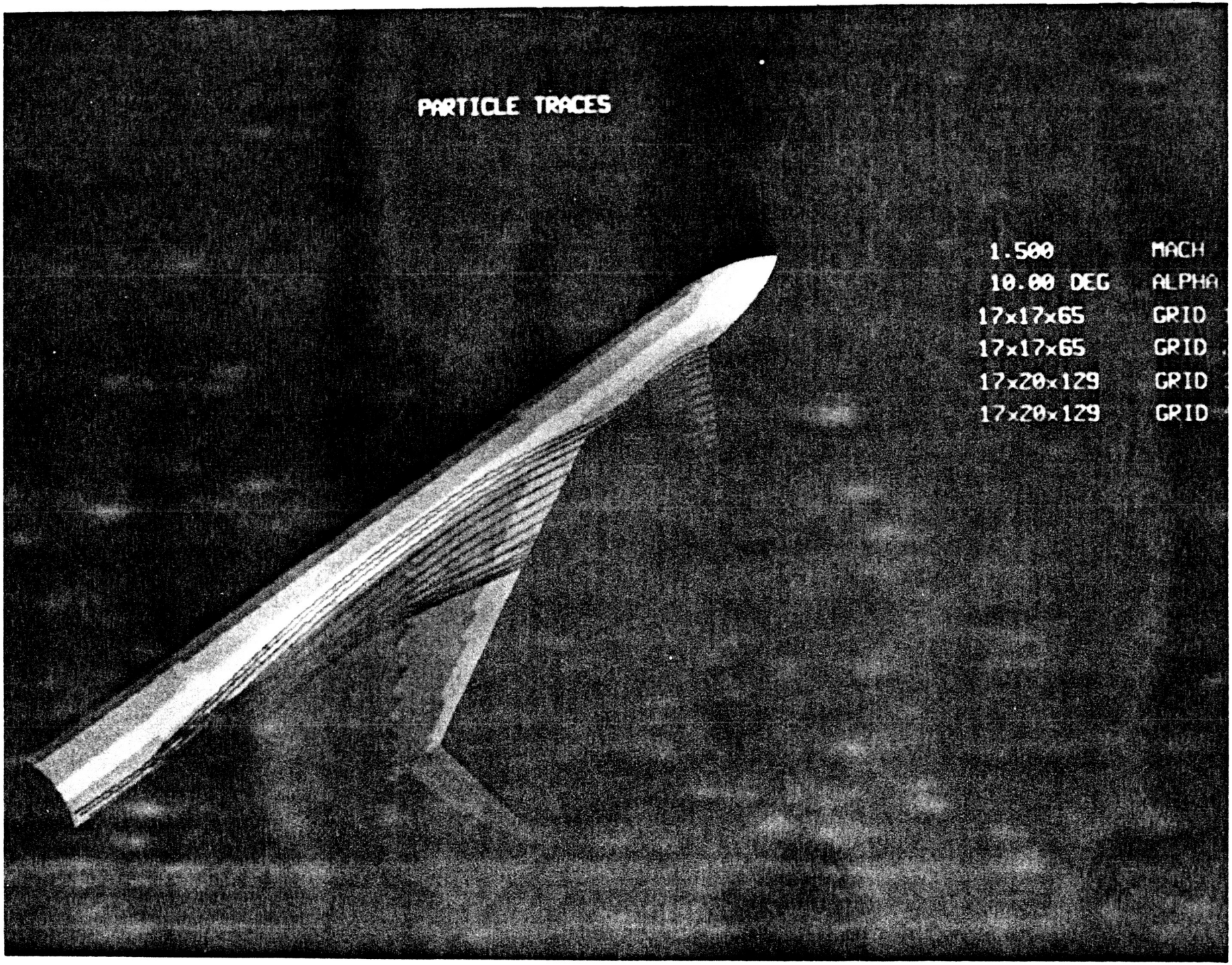

Fig. 12. A different view for the case of Fig. 10.

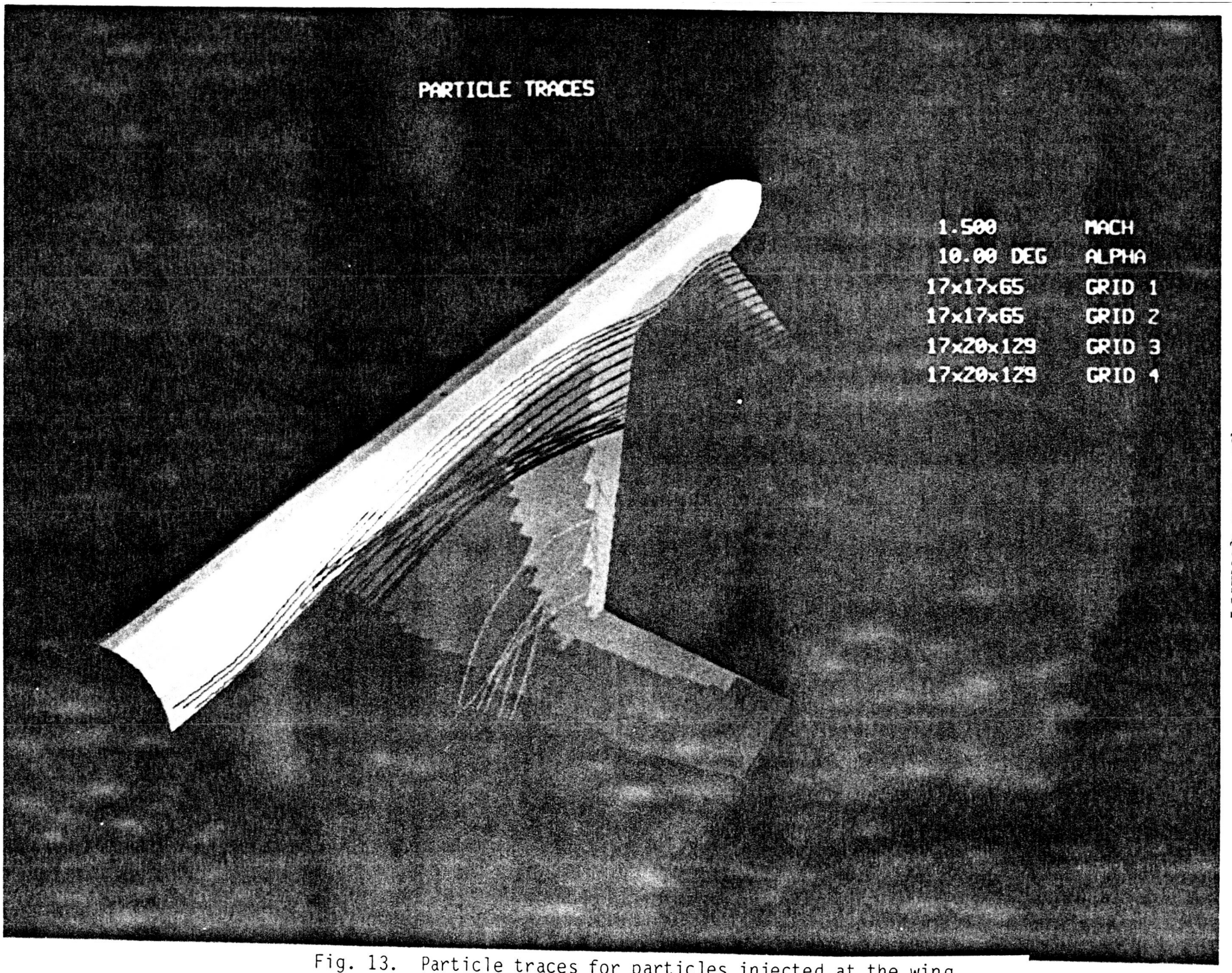

Fig. 13. Particle traces for particles injected at the wing leading edge.

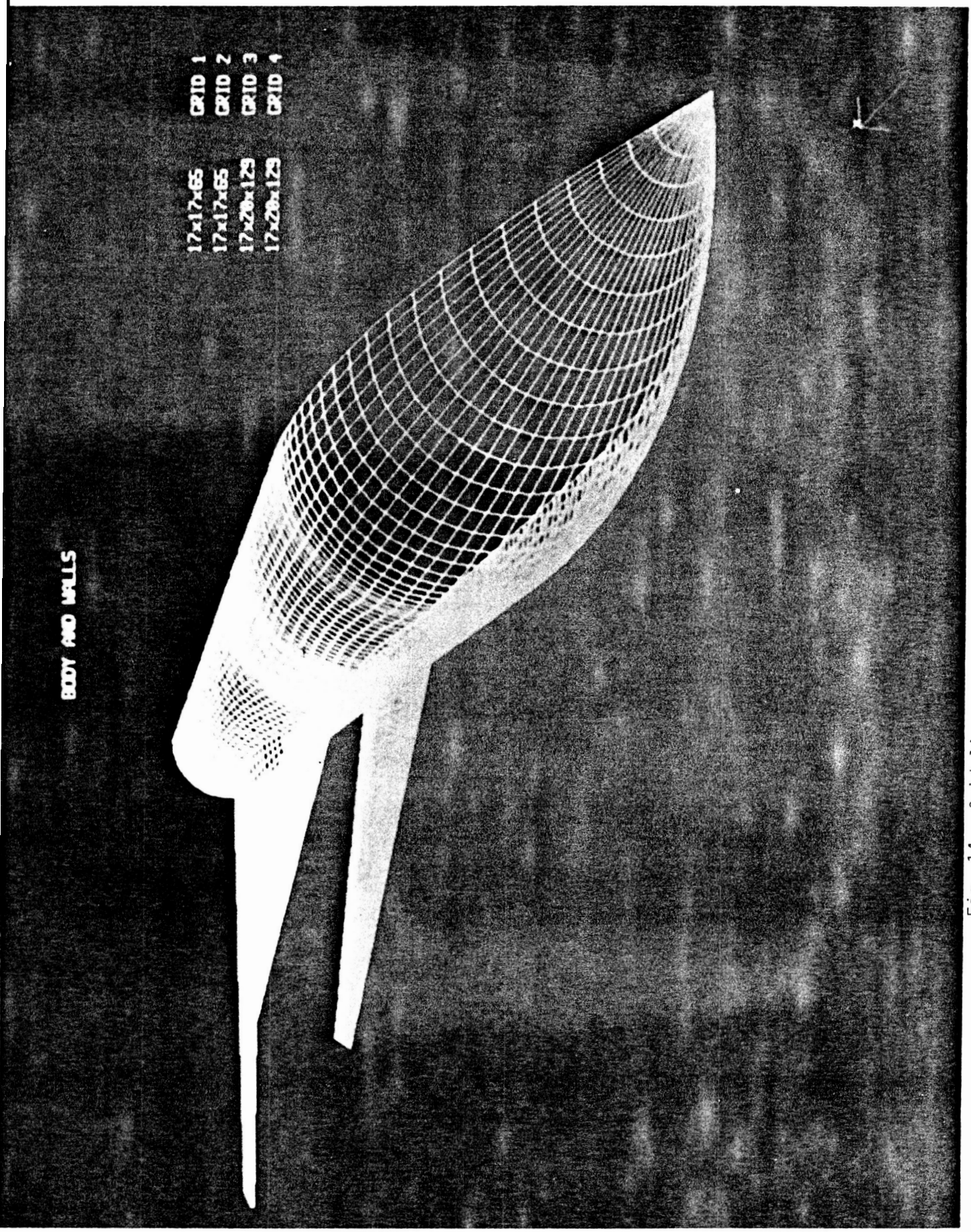

Fig. 14. Grid lines of an aircraft configuration.

lines of the aircraft configuration. The configuration consists of four grid systems. If the grid lines of different grid systems do not connect properly, one would detect the problem immediately by viewing at the displays.

5. CONCLUSIONS

A process for the color visualization of flow field variables using digital imaging techniques has been presented. The process offers a researcher a qualitative tool to analyze his results from the CFD computation. A flow visualization code referred to as PLOT3D has been developed and widely used for displaying grids and flow fields quantities in either static or dynamic mode. This code is run on a color IRIS workstation. The overview of the IRIS workstation was described as well. The digital imaging technique reduces a large volume of numeric data to a pictorial from which can be quickly evaluated by visual inspection. It also enhances data presentation and documentation.

REFERENCES

1. MacCormack, R. W. and Lomax, H., "Numerical Solution of Compressible Viscous Flows, "Annual Reviews in Fluid Mechancs, Vol. 11, 1979, pp. 289-316.

2. Chapman, D. R., "Computational Aerodynamics Development and Outlook," AIAA Journal, Vol. 17, No. 12, March 1979.

3. Graves, R. A., "Computer Fluid Dynamics the Coming Revolution," AIAA Journal, Vol. 20, No. 3, March 1982.

4. Smith, R. E. and Speray, D. E., "Color Visualization for Fluid Flow Prediction," Computer in Flow Prediction and Fluid Dynamics Experiments, ASME Winter Annual Meeting, Washington, D.C., December 1981.

5. Smith, R. E. and Speray, D. E., "Color Visualization for Fluid Flow Prediction," Mechanical Engineering, Vol. 104, No. 3, March 1982.

6. Edwards, C. L., Meissner, F. T., and Hall, J. B., "The Use of Computer Generated Color Graphic Images for Transient Thermal Analysis," NASA Technical Paper 1455, July 1979.

7. Foley, W. M. and Van Dam, A., Fundamentals of Interactive Computer Graphics, Addison-Wesley, Reading, Massachusetts, 1982.

8. Smith, R. E. and Speray, D. E., and Everton, E. L., "Visualization of Computer-Generated Flow Field," Flow Visualization III Proceeding of the Third International Symposium on Flow Visualization, September 6-9, 1983, University of Michigan, Ann Arbor, Michigan, Hemisphere Publishing Corporation, Washington, D.C., 1985.

9. Park, S. K., "A Conceptual Model for the Logical Assignment of Color to a Two-Dimensional Display of Scalar Data," Private Communications.

10. Van Dyke, M., An Album of Fluid Motion, The Parabolic Press, Stanford, California, 1982.

11. Rai, M. M., "A Conservative Treatment of Zonal Boundaries for Euler Equation Calculations," AIAA 22nd Aerospace Scienes Meeting, Paper 84-0164, January 1984.

12. Erikson, L-E., "Flow Solution on a Dual Block Grid Around an Airplane," Paper presented at the First World Congress on Computational Mechanics, Austin, Texas, September 22-26, 1986.

13. Pulliam, T. H. and Steger, J. L., "Recent Improvements in Efficiency, Accuracy, and Convergence for Implicit Approximate Factorization Algorithms," AIAA 85-0360, AIAA 23rd Aerospace Sciences Meeting, Reno, Nevada, January 1985.

APPENDIX A

A GUIDE TO THE USE OF PLOT3D

Introduction

PLOT3D is a flow visualization code created at NASA Ames and runs on UNIX operated IRIS Workstations. To use PLOT3D requires only a little knowledge about the UNIX operating system. PLOT3D has a capability of displaying grids, flow field quantities, particle tracing, etc. It displays these quantities in either statics or dynamics mode. In the dynamics mode, a user can translate, rotate or zoom the image through the use of a mouse. PLOT3D is a useful tool for a CFD researcher. A CFD researcher can either analyze or diagnose his data utilizing PLOT3D. This report is a quick guide for new users who would like to obtain experience with PLOT3D.

Since, PLOT3D also has the capability of dealing with multiple grids (several grid systems combined in a flow field, it is called "Zonal Grid" by some). This report is divided into two parts, single grid and multiple grids. Single grid is outlined first. There are only a few additional things needed for multiple grids.

Single Grid - The first question that comes to a new user's mind is, "What is needed in order to display flow field data utilizing PLOT3D?".

In general, there are two sets of data needed for PLOT3D. They are data sets for grid points (x-y-z) and flow quantities (ρ, ρu, ρv, ρw, E_t). If the user is interested only in displaying grid points, however, data sets for flow quantities are not needed. Data sets can be created on the IRIS, but a user would need to know more about the IRIS (Compilation, etc.) and UNIX. Also, in realistic terms, a user runs his CFD code on NOS or VPS-32

and stores his results in a file (or files) residing in NOS memory. So, it makes sense for this report to describe how to get these data sets (on NOS) ready for PLOT3D.

PLOT3D takes data files in FORTRAN FORMATTED or UNFORMATTED form. However, under IRIS/UNIX, I/O is done much more efficiently using BINARY files. For this reason, BINARY is the default under PLOT3D, and it is recommend that the user write his data as BINARY files. Since BINARY formats may be different for different machines, a conversion file has been written to convert a FORTRAN FORMATTED file into BINARY file suited for PLOT3D. The user needs to use this file to convert his FORTRAN FORMATTED data files into BINARY files (unless he wishes to write his conversion code himself), and send them from NOS to the IRIS. Below is a brief procedure for getting the data files ready for PLOT3D. Note that the user should write the data in the right-handed coordinate system, otherwise unexpected errors may result.

First - write the data files resulting from a CFD code as follows:

Grid file (x-y-z file)

```
      WRITE(unit,10) IDIM,JDIM,KDIM
      WRITE(unit,20) (((X(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
      WRITE(unit,20) (((Y(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
      WRITE(unit,20) (((Z(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
10    FORMAT(.....)
20    FORMAT(.....)
```

Flow variable file (Q-file) ..... Should be in a separate file

```
      WRITE(unit,10)  IDIM,JDIM,KDIM
```

```
WRITE(unit,20)  FSMACH,ALPHA,RE,TIME
WRITE(unit,30)  ((((Q(I,J,K,NX),I=1,IDIM),J=1,JDIM),K=1,KDIM),NX=1,5)
```

where FSMACH = Mach Number

ALPHA = Angle of attack

RE = Reynolds Number

TIME = Time

$Q(I,J,K,1) = \rho$ = Density

$Q(I,J,K,2) = \rho u$ = x-momentum

$Q(I,J,K,3) = \rho v$ = y-momentum

$Q(I,J,K,4) = \rho w$ = z-momentum

$Q(I,J,K,5) = E_t$ = Energy

If the user is interested in displaying density only, he needs to fill up the rest of Q, i.e., Q(I,J,K,NK),NK = 2,5, with some dummy values.

Note that certain reference conditions are assumed in calculating some functions.

ρ_∞	=	1	(freestream density)
C_∞	=	1	(freestream speed of sound)
P_∞	=	1/gamma	(freestream pressure)
$\lvert V_\infty \rvert$	=	$M_\infty * C_\infty$	(freestream velocity magnitude)
gamma	=	1.4	(ratio of specific heats)
R	=	1	(gas constant)

It is a good idea for a user to change his reference conditions in accordance with the above condtions before writing out his data file.

Second - Use the conversion routine, CONVXEQ, to convert the data files into the BINARY format. In order to use this routine some information is

needed at the beginning of the file. The information consists of optional comments and required format specifications. For details see Appendix B. Below is an example of a typical PLOT3D file.

```
C-EXAMPLE PLOT3D FILE HEADING
C .........
F3
(315)
F1000
(5F12.7)
10      10      10
.0000000          .1000000      .2000000      .3000000      .4000000
........          ........      ........      ........      ........
........          ........      ........      ........      ........
```

From the example, F3 and (315) and F1000 and (5F12.7) are referred to as format specification pairs. Some warning needs to be made here.

-- There may not be more than 10 format specification pairs in a file.

-- There may not be more than 10 numeric character strings on a line. i.e.,

```
F11
(11F12.5)
```

This will not work.

-- A format that results in the uneven number for each line should not be used, i.e.,

```
F11
```

```
(5F12.5)
1.00000     .......     .......     .......     .......
.......     .......     .......     .......     .......
.......
```

The last line has a different number of strings than the rest (1 instead of 5). This will result in an unexpected error. If the user can not avoid it, two format pairs may be used. The above example may be written as

```
F10
(5F12.5)
F1
(1F12.5)
```

After these headings have been put in a data file, the user can convert his data file to BINARY form by the following procedures.

```
GET,CONVXEQ/UN=276519C                    < CR >
CONVXEQ,filename1          < CR >    (filename1 = datafile)
REPLACE,TAPE4=filename2  <   CR >   (filename2 = name to be saved as
                                      NOS permanent file)
```

Now the user has a BINARY file in NOS memory. To transfer the file to the IRIS, log off from NOS. Type 71 for ENTER RESOURCE and BRIDGE for USER NAME and the rest is just simply answering questions. For example, see APPENDIX C. The procedure here is easy. However, there is a common mistake many new users have made. Recall that under the UNIX system uppercase and lowercase letters are distinct. For example, on most terminals the letters are set as uppercase by "Caps Lock", if the user forgets to release this key, he may not be able to transfer his files to the IRIS. Headaches may result before finding it out.

The user, now, has data files on the IRIS ready for PLOT3D. Log onto the IRIS and after getting a prompt type : plot3d, this will lead the user into the interactive PLOT3D program (similar to TBGG, an interactive grid generation program). Since this report is intended to give just a brief outline to get a new user started on using PLOT3D, details of how to use PLOT3D will not be given here. A demonstration will be sufficient. Here, an outline of some common options is given. For more details see Appendix D.

The first option which a user should select is "read." PLOT3D reads in both grid and flow variables files. If the user is interested only in displaying grid points, just hit < CR > when asked for Q file. The following are a few common options a user may select.

% walls

-- specifies which parts of the grids should be drawn for all plots, indicating the grid "geometry" or "configuration."

-- walls can be added from the previous walls by selecting "walls/add."

If one selects just "walls," however, the previous walls will be deleted.

%function

-- allows the selection of the function number(variables) to be plotted. For example "function 100" will select density to be plotted when "plot" option is type. For more details of function see Appendix D.

% contours

-- selects contour levels or ranges of the function (variables) to be plotted.

-- This option is useful when one would like to display a scalar functions.

% subsets

-- specifies a plane (or planes) on which a function (variable) is plotted.

-- note that subsets is set to every grid point if this option is never chosen. So, if the user attempts to plot a function without selecting subsets, the image on the screen may look very confusing and too much time may be consumed.

For particle traces, select "function 300," then "rakes."

% rakes

-- specifies the starting locations and attributes of particle traces.

-- If too many locations are chosen, too much computational time is needed, also the particle traces may look like a mess of spaghetti.

-- Note that one can restrict traces to remain in physical or computational planes. If the user is interested in tracing the particle for the whole flow field, however, he needs to select "subsets" to be every grid points (if "subsets" have been chosen otherwise).

After walls, function, subsets, contour or rakes have been decided upon, one types "plot" to display the image.

Multiple Grids - There are only a few variations in displaying flow field variables resulting from multiple grid calculations. These variations exist simply because one needs to, somehow, tell PLOT3D that he is working with multiple grids.

% Blanking

An array IBLANK can be included in the x-y-z grid file which supplies PLOT3D with information on grid topology. This can be used to indicate that some points in the grid are "turned off" (so contours are not plotted through this region), or how multiple grids are connected (to aid in particle tracing). The corresponding grid point is assumed turned off where IBLANK is zero, and data at that point (xyz, Q) are never used. The normal value for IBLANK in the interior of a grid is 1.

IBLANK can be used in single grid as well as multiple grids. IBLANK is particularly useful for particle tracing (for multiple grid). IBLANK value of "-n" means that physical trace leaving the current grid here will continue in grid "n." If a surface of a grid volume element is common to more than one grid system, different IBLANK may be used at different grid points of that surface.

For multiple grids, write statements for data files following the example below:

```
Grid file (x-y-z file)
     WRITE(unit,10) NGRID
     WRITE(unit,20) (IDIM(IGRID),JDIM(IGRID),KDIM(IGRID),IGRID=1,NGRID)
     DO 30 IGRID=1,NGRID
     WRITE(unit,40) (((X(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID),K=1,KDIM
                    (IGRID))
     WRITE(unit,40) (((Y(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID),K=1,KDIM
                    (IGRID))
     WRITE(unit,40) (((Z(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID),K=1,KDIM
                    (IGRID))
```

```
      WRITE(unit,50) (((IBLANK(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID)
                      ,K=1,KDIM(IGRID))
30    CONTINUE
```

Note that IBLANK array cn be neglected if not needed.

Flow variable file (Q-file)

```
      WRITE(unit,10)  NGRID
      WRITE(unit,20)  (IDIM(IGRID),JDIM(IGRID),KDIM(IGRID),IGRID=1,NGRID)
      DO 30 IGRID=1,NGRID
      WRITE(unit,40)  FSMACH,ALPHA,RE,TIME
      WRITE(unit,40) ((((Q(I,J,K,NX),I=1,IDIM(IGRID)),J=1,JDIM
                        (IGRID)),K=1,KDIM(IGRID)),NX=1,5)
30    CONTINUE
```

The conversion to BINARY files and transferring them to the IRIS are the same as in single grid and will not be described again.

Here are a few options in PLOT3D which are different from single grid.

```
% read/mgrid
--   instead of just "read."
% read/mgrid/blank
--   if IBLANK is array is added.
% walls/grid=N
--   where N = grid number
--   if the user chooses just "grid," it will assume N=1.
% subset/grid=N
```

-- same as in "walls."

-- note: when one choose to plot contour on particular surfaces of any particular grids, he needs to turn off subsets of grids which he is not interested in plotting. Recall that subsets are set equal to every grid point of each grid. If one is interested in tracing a particle for the whole flow field, he needs to set these subsets to every grid point of each grid.

Miscellaneous

-- options will stay on PLOT3D until the user changes them. For example, one can do contour plotting of different functions by just hange "function." Here "contours" is not changed, different functions will be plotted with the same contour levels, etc.

-- MOUSE

- Right button -- translation
- Middle button -- zooming
- Left button -- rotation
- The transformations are performed slowly if a lot of grid points are to be displayed. However they are performed so fast that the user can not have the good control of the MOUSE, if too few grid points are to be displayed.

APPENDIX B

NOS → IRIS DATA CONVERSION

Mike Fischbein

Version 2.0, 6 March 1986

The routine CONVXEQ translates a file of numeric character strings to a packed binary file suitable for direct transfer to the IRIS workstations. Preceeding the numeric information in the data file are two items: an optional comments section, and a required format section. The information in the comments and format sections is written to the binary file after the numeric data. The routine CONNXEQ, with a calling sequence and input requirements identical to CONVXEQ, writes only the numeric data and should be used if it is necessary to have only valid numeric data in the output file.

COMMENTS

Comment lines are optional. If used, they must be the first lines in the data file and must start with a capital letter 'C'. They will follow the binary data in the output file. Up to 10 comment lines of 80 characters each are permitted.

FORMAT SPECIFICATION

Each group of data must have a two line format specification at the beginning of the file. There may be up to ten format specification pairs grouped together after the comments (if any) and before the data. They are copied to the output file after the comments section.

The first line of each format specification must start with a capital letter 'F' and be followed by the number of entries to which the specification applies. The second line contains an I, F, G, or E FORTRAN format specification in parentheses. For example, if a thousand numbers were written with a 5F10.2 format, the format block should look like:

```
F1000
(5F10.2)
```

As a second example, a typical plot3d file consists of one line of three integers followed by thousands of real numbers. The start of this file might be:

```
C EXAMPLE PLOT3D FILE HEAD
C FOR THE DOCUMENTATION
C
F3
(3I4)
F130680
(6E12.5)
33  11  120
  .00000E+00     .62500E-01     .12500E+00     .18750E+00     .25000E+00     .31250E+00
  .37500E+00     .43750E+00     .50000E+00     .56250E+00     .62500E+00     .68750E+00
  .75000E+00     .81250E+00     .87500E+00     .93750E+00     .10000E+01     .10625E+01
  .60676E+00    -.65733E+00    -.70789E+00    -.75845E+00    -.80902E+00    -.85958E+00
 -.91014E+00    -.96071E+00    -.10113E+01    -.10618E+01    -.11124E+01    -.11630E+01

 ...
```

USE OF CONVXEQ

Call the tape library (x2438) to get a tape. It is recommended that a mnemonic name be used. Login to the C machine, as it is the preferred machine for tape usage. GET your data file(s). Get the utility CONVXEQ with:

```
GET,CONVXEQ/UN=276519C<cr>
```

Have the system operator mount the tape (this may take some time. if all the tape drives are in use) by the command:

LABEL.TAPE4.VSN=*tapename*.D=1600.F=L.LB=KU.PO=W<cr>

where *tapename* is the name you gave to the tape library and they wrote on your tape reel.

When the tape is mounted and you are prompted for your next line, type CONVXEQ.*inputfile* where *inputfile* is the name of your data file. To write another file on the same tape, type REWIND.CONVXEQ (do *not* REWIND.* or you will write over your first file) and type CONVXEQ.*inputfile2*. Repeat as necessary. A single 2400' tape reel can hold about 9 million numbers, so please don't try to write any more than that.

To read the tape to the IRIS, the program myftp should be used. Have one of the icase system people mount your tape. Then login to the icase VAX (or have someone else do it) and type

/usr/msf/myftp/myftp larc<cr>

You will be prompted for your IRIS login and password. If you want to place the files somewhere other than your home directory, cd to that directory. Then type

send <cr>

You will be prompted for the local file name. Type

/dev/rmt12<cr>

exactly; you will then receive a prompt for the remote file name. Type what you want the file to be called on the IRIS, and a <cr>. If you have more than one file on the tape, type send again and continue as above.

When you are finished, quit the ftp program and either continue working on the icase VAX or logout.

For the curious, data input to CONVXEQ are strings in NOS six bit numeric characters, (prefaced by optional comments and the required formatting information). Output is a packed file of binary data with integers represented in two's complement, 32 bit form, and floating point numbers in IEEE standard 32 bit with one sign bit, eight bit excess 127 exponent, and 23 bit normalized binary characteristic. CONVXEQ does no error checking and will produce garbage data if fed numbers that do not meet the limits of these formats. In particular, if a floating point number has an exponent of greater than about 38 or less than about −38 (but is not zero) the result will be garbage. It should be noted that up to three significant bits (of 24) may be lost in the data conversion process.

If you have any questions, or are not completely sure of all of these instructions, see Mike Fischbein in Room 140, extension 2396.

Note : This portion is not needed now. Instead, a user types;

REPLACE,TAPE4=filename

where filename is the name of the BINARY file to be saved as NOS permanent file.

ORIGINAL PAGE IS
OF POOR QUALITY

APPENDIX C

MOVING A FILE FROM MACHINE TO MACHINE

```
NASA LANGLEY CENTRAL COMPUTERS
ENTER RESOURCE 71
GO

 NASA Langley Research Center MicroVAX/VMS V4.3      - CMB

 Username: BRIDGE
 ************************************************
 *                  ALL SYSTEMS NORMAL                  *
 ************************************************
 Bridge version v2.1.
 ************************************************************
 New version of Bridge.
 You can now print on the Cybers (if you have a NOS account).
 Call Lee (x2812) if there are problems.
 ************************************************************

 Move or Print a file? (M/P/?) [M]> m
 transfer mode: interactive or batch? (I/B/?) [I]> i
 from node (? for list, Q to quit)> ?

 Available nodes:
  122905      AIRLAB1     AIRLAB5     AMB         **Note; BILL is the IRIS residing in
  AMB2        AME         AMELIA      BILL                ACD's graphics Lab. CLYDE and
  CLYDE       CMB         CMBVAX      CSAB                FLASH are the other IRISs
  CSAB01      CYBA        CYBC        CYBD                residing at CMB (next to ICASE)
  CYBY        CYBZ        FLASH       HARVEY
  ICASE       NAVIER      PAM         PRANDTL
  PTR         SAMUEL      SSD1        SSD2
  SSC

 from node (? for list, Q to quit)> cyby
 cyby username> 469381e
 >cyby password>
 cyby charge number> >

 to node (? for list, Q to quit)> bill
 bill username [469381e]> monchai
 >bill password> >

 from file (<name>/<!-command>/<CR> to quit/?)> grid12b
 to file (<name>/?) [grid12b]> grid12b.bin
 File Type: Text (formatted) or Binary? (T/B/?) [T]> b
 transferring grid12b from cyby...

   FLINN Operation initiated.
   FLINN Complete.
   SIZE:    59755 Bytes
   TIME:     13.3 Seconds
   RATE:    35340 Bits/Second

 transferring grid12b.bin to bill...

 from file (<name>/<!-command>/<CR> to quit/?)>
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
NASA LANGLEY CENTRAL COMPUTERS
ENTER RESOURCE 71
GO

 NASA Langley Research Center MicroVAX/VMS V4.3     - CMB

 Username: BRIDGE
 ************************************************
 *                  ALL SYSTEMS NORMAL                  *
 ************************************************
 Bridge version v2.1.
 ************************************************************
 New version of Bridge.
 You can now print on the Cybers (if you have a NOS account).
 Call Lee (x2812) if there are problems.
 ************************************************************

 Move or Print a file? (M/P/?) [M]> m
 transfer mode: interactive or batch? (I/B/?) [B]> i
 from node (? for list, Q to quit)> ?

 Available nodes:
  122905      AIRLAB1     AIRLAB5     AMB          **Note; BILL is the IRIS residing in
  AMB2        AME         AMELIA      BILL                 ACD's graphics Lab. CLYDE and
  CLYDE       CMB         CMBVAX      CSAB                 FLASH are the other IRISs
  CSAB01      CYBA        CYBC        CYBD                 residing at CMB (next to ICASE)
  CYBY        CYBZ        FLASH       HARVEY
  ICASE       NAVIER      PAM         PRANDTL
  PTR         SAMUEL      SSD1        SSD2
  SSC

 from node (? for list, Q to quit)> cyby
 cyby username> 469381e
 >cyby password>
 cyby charge number> >

 to node (? for list, Q to quit)> bill
 bill username [469381e]> monchai
 >bill password> >

 from file (<name>/<!-command>/<CR> to quit/?)> grid12b
 to file (<name>/?) [grid12b]> grid12b.bin
 File Type: Text (formatted) or Binary? (T/B/?) [T]> b
 transferring grid12b from cyby...

   FLINN Operation initiated.
   FLINN Complete.
   SIZE:    56755 Bytes
   TIME:     13.3 Seconds
   RATE:    35340 Bits/Second

 transferring grid12b.bin to bill...

 from file (<name>/<!-command>/<CR> to quit/?)>
```

APPENDIX D
DETAILS OF PLOT3D PROGRAM

Help Library file: plot3d

HELP

PLOT3D Version 3.5 27 April 1987

Format: HELP [keyword [keyword [keyword ...]]]

The HELP command prints information on a list of keywords. "*" is a wildcard, while "*..." matches anything at the current level or below. Thus "HELP *..." prints all HELP information available. When responding to a "Topic?" or "Subtopic?" prompt, a "?" causes information for the current level to be repeated; a <RETURN> pops HELP up one level, and an end-of-file terminates the HELP session.

General_information

In this documentation, information which can be included on a command line which is optional is enclosed in [square brackets]. Control characters are denoted by a preceding caret (^) (i.e., ^D for control-D).

The basic information needed for making a plot (after reading in the XYZ and Q data files) are: (1) the function to plot (FUNCTION command), and (2) the plot ranges, i.e. XMIN, XMAX, etc. (MINMAX command). A contour plot of Mach number can thus be made using the commands

```
READ/2D/XYZ=xyzfile/Q=qfile
FUNCTION 154
MINMAX -1 2 -1.5 1.5
PLOT/2D
```

Arguments can be delimited by either blanks or commas, and real numbers entered as integers (no decimal point) will be interpreted correctly.

The VT100 "green screen" and TEKTRONIX terminals are treated (and respond) identically.

Versions for the following machines/graphics libraries are available:

- o Silicon Graphics IRIS/Unix (GL2-W3.5)
- o Silicon Graphics IRIS/Unix (GL1-W2.4)
- o VAX/VMS (V4.4) and DISSPLA (V10.5)
- o CRAY/COS (1.14) and DISSPLA (V9.2)
- o Apollo/Aegis and GMR3D (V1.0)

If you have any suggestions or find any anomalies, please contact:

Pieter G. Buning
MS 258-2
Computational Fluid Dynamics Branch
NASA Ames Research Center

Moffett Field, CA 94035
(415) 694-5194 or FTS 464-5194

Journal_file

During a PLOT3D session, a "journal file" named PLOT3D.JOU is kept of all commands entered to the program. In the case of an abnormal termination (i.e., crash), this file is saved. The file can also be saved by exiting from PLOT3D using EXIT/SAVE (or QUIT/SAVE). Normally the journal file is deleted when the program terminates.

Initialization_file

When PLOT3D first starts up, it looks for a file "PLOT3DINI.COM" in the current directory. If the file is there, PLOT3D reads commands from it first, before taking input from the normal input device (the terminal).

Installation_of_PLOT3D

Installation notes for PLOT3D Version 3:

1. Check dimension of WORK in MAIN. It should be at least 13 times the maximum number of grid points.
2. Change help library filename (HLBFIL) in SUBROUTINE HLPCMD.
3. COMMON blocks /WALL/ and /SUBS/ currently use ~400,000 words each, due to the maximum number of (multiple) grids being set to 50. To reduce this storage, change all occurrences of "MGRID=50" to "MGRID=5" (or whatever).
4. Look at SUBROUTINE PLTCMD for current plotting device names, SUBROUTINE PLTDEV for calls to routines which initialize devices.
5. Operating system-specific routines are at the end of the source file, and start with BLOCK DATA BCHPK. Check all these routines, but ones which will certainly need to be changed are

BLOCK DATA BCHPK	# characters/word.
SUBROUTINE FILNAM	form of filenames on system, usually of the form "FILENAME.TYPE".
SUBROUTINE GETLUN	sets available unit numbers.
LOGICAL FUNCTION INTERA	interactive or not (see VMS version for what I really want to do).
CHARACTER*80 FUNCTION STDDEV	sets standard I/O device names.

6. For the IRIS 2000 series terminals and workstations (GL2): subroutine IRIS contains a PARAMETER (IZNEAR=m,IZFAR=n) statement relating to z-buffering. IZNEAR is currently set to 0 and IZFAR to 32766 (for a 32 bitplane system), and should be set to 0 and 4095, respectively, for a 28 plane system. Z-buffering should not be attempted on a system with fewer than 28 bit planes: IZNEAR and IZFAR should be set to zero.

Release_notes

3.2

New features in version 3.2:

1. /CHECK/NOCHECK option on reading Q files.
2. /BINARY option (default) on reads for IRIS/Unix, as well as /FORMATTED and /UNFORMATTED.
3. Default file types are .FMT for formatted, .DAT for unformatted, and .BIN for binary.
4. Maximum number of grids increased from 10 to 50.
5. Faster initial search for RAKE/X particle points.
6. Use of C (binary) I/O for faster reading and writing of particle trace files on IRIS/Unix.
7. "P" for picture save (screen image dump) on IRIS version.
8. "Up-arrow" and "down-arrow" to increase or decrease mouse sensitivity on IRIS version.
9. Default contour colors changed from "WHITE" to "BLUE CYAN GREEN YELLOW RED MAGENTA".
10. Highlighting on surface-shading disabled.

3.3

New features in version 3.3:

1. 3D particle traces no longer require right-handed (i,j,k) system.
2. PLOT/SCRIPT option for movie scripts.
3. EXIT command (same as QUIT).
4. PLOT/IRISSINGLE and color maps for IRIS systems with fewer than 24 bit planes.
5. Special command-line characters explained in HELP.
6. Command files calling command files and returning fixed.
7. VPOINT/ANGLES fixed.

3.4

New in version 3.4:

New functions implemented:
- Scalar functions:
  - Cross-flow velocity
  - Stagnation pressure coefficient
  - Pitot pressure
  - Pitot pressure ratio
  - Dynamic pressure
  - Normalized 2D stream function
  - Swirl
  - Speed of sound
  - Shock function
- Vector functions:
  - Perturbation velocity
  - Pressure gradient

Functions corrected:
Normalized stagnation density
Normalized stagnation temperature

Commands/options implemented:
PLOT/2D - 2D plots implemented.
PLOT/LABELS - works for 2D contour plots using DISSPLA.
PLOT/SURFACE (or /CARPET or /LINE) - for 2D line plots or 3D function surface plots.
FSURFACE command - to define properties of line or function surface plot.
VIEW command - for 2D plots or 3D function surface plots.
MINMAX/XSCALE=scale/YSCALE=scale/ZSCALE=scale - to scale axes independently.
PLOT/AXES/NOAXES - axes and axis labels added for IRIS too.
PLOT/FIGURE=(areax,areay,char ht) - for figure-quality plots from DISSPLA.
READ/PLANES - to read data one IJ plane per record.
Symbols and arrowheads have been implemented for the IRIS.
CONTOUR grid-lines - for grid lines colored by contour value.

Defaults/initial values changed:
SUBSETS/NOATTRIBUTES is now the default.
Contour color map default is now "BLUE CYAN GREEN YELLOW RED MAGENTA" instead of "WHITE".
MINMAX 0 10 -5 5 -5 5 is the initial MINMAX box.
VPOINT/ANGLES 150 20 20 is the initial viewpoint for 3D plots.

IRIS features incorporated:
PLOT3D runs under the MEX window manager (but does change the color map) (IRIS 2xxx only). You need a .mexrc file, with commands "reservebut 13", "bindfunc hogwhiledown 13", and "bindfunc menu 101" in it. See the Window Manager section of the IRIS User's Guide for more information.
Keyboard attention interrupts (^C or <BREAK>) return PLOT3D to the command level prompt.
$system-command may work, depending on system swap space.
Q key - dumps the screen to a file for printing on the QMS Lasergrafix 1200 printer (IRIS 2xxx only).
Z key - toggles z-buffering mode (IRIS 2xxx only).
3,6,N keys - change to 30Hz, 60Hz, NTSC video out, respectively (IRIS 2xxx only).

IRIS bugs fixed:
TBL6 6-plane color table fixed.
IRIS 2xxx PXL screen dump and PLOT3DPIC fixed.

DISSPLA device drivers rearranged:
PLOT/VER now produces files PARM.PLV and VECTR1.PLV for plotting on a VERSATEC connected to a VAX. At NASA Ames, see "$HELP PLOT" on how to submit these files to be printed.
PLOT/DIP produces a .DIP file for plotting on a VERSATEC connected to a PDP 11. At NASA Ames, a .DIP file can be

converted to a .QMS file for printing on the QMS Lasergrafix printer with the "$DIPQMS" command. PLOT/QMS produces a .QMS file directly, but this file may be very large.

VAX/VMS features restored:
HELP, keyboard attention interrupts (^C), and error recovery now work under VMS 4.2.

ARCGraph GRAFIX interface written - a PLOT3D version (commonly known as PLOT3X) for NASA Ames device-independent graphics files has been written.

3.5

New_features

New commands/options:

- o RAKE/MAXPOINTS=n option added to limit length of particle traces.
- o RAKE/SCALAR_FUNCTION option added for coloring particle traces by a scalar function.
- o PLOT/LN03 and PLOT/IMAGEN devices added to DISSPLA version.
- o PLOT/UP=axis option added.
- o PLOT/BACKGROUND=color option added (not in DISSPLA; background color must be set in GAS for GRAFIX).
- o SHOW PLOT command added.
- o SHOW TEXT command added.
- o SHOW FUNCTION also shows the (i,j,k) location of function min and max.
- o LIST command added, supporting UNFORMATTED<->FORMATTED<->BINARY XYZ and Q data file conversion.
- o VECTOR attributes added to allow many different types of vector displays.
- o READ prints an informational message when reading a file, including grid size.
- o FSURFACE/NOCONTOUR option name changed to FSURFACE/GRID.

New features/versions:

- o Particle tracing uses 2nd-order Runge-Kutta time-advance, still 5 steps per grid cell. Slow, and oil flow ain't always what we'd want.
- o Removed XY,XZ,YZ restriction capability for particle tracing.
- o CONTOUR plane orientation doesn't matter anymore if subset is just a plane.
- o COLOR_CONTOURS now fill in between contour lines, instead of filling each grid cell with only one color.
- o Contour GRID_LINES change color at contour levels, rather than having only one color for each grid line segment.
- o SHADED_SURFACE walls that are just a single line will be drawn and filled as a single polygon. This can be used to fill in an airfoil outline, for instance.

- o HIDDEN_LINE attribute enabled, as polygons filled with the background color, then outlined. (Ordering of the polygons is thus still important.)
- o Polygon shininess attribute enabled. See "HELP Attributes Shininess" for an explanation.
- o Journal file capability added--file PLOT3D.JOU saved on abnormal exit or EXIT/SAVE.
- o Auto-initialization file capability added--file PLOT3DINI.COM will be read and executed on startup of PLOT3D, if it is found in the current directory.
- o HELP utility vastly improved.
- o APOLLO version written, using AEGIS and GMR3D; it doesn't do everything, but it's workable.

New functions:

- o Functions "Velocity x vorticity" and "Velocity x vorticity magnitude" added.
- o Function "Entropy measure s1" added, a scaled, normalized version of exp(s).

IRIS improvements:

- o IRIS screen saves rewritten to use run-length-encoding for smaller files, and to use the ARCGraph RAS file format. This format encodes RGB instead of color map entries, so that pictures can be restored in any mode. A new program RASP restores the screen saves, replacing the program PLOT3DPIC.
- o Seiko color printer screen save added to the IRIS version. See "HELP PLOT IRIS_interface Seiko_color_printer_screen_save".
- o More colors available for 6, 8, and 10-plane plots on IRIS.
- o Up to 8 simultaneous graphics windows under window manager on IRIS.
- o Increased WORK array size from 2.5M to 3Mwords on IRIS. This will reduce maximum image complexity.
- o Backface polygon removal implemented for IRIS, tied to "B" key during display.
- o Unix SETENV symbols will be translated in PLOT3D when enclosed in 'single primes'.

DISSPLA improvements:

- o 2D polygons now use shading in DISSPLA.

CRAY 2 improvements:

- o The CRAY 2/IRIS version of PLOT3D now tries to start up PLOT3D_LCL on the workstation.

Bugs_fixed

Commands/options:

- o READ/PLANES for XYZ file bug fixed (blanking vs. no blanking calls were reversed).
- o READ/X="xfile"/Q="qfile" bug fixed (was getting misinterpreted).

o PLOT/AXES/NOAXES option fixed (wasn't always getting set right).

Features:

o Last contour level coming out white fixed, I hope for good (a roundoff problem).
o "RGB 0 0 0" color now treated correctly.
o Displaying IK planes in 2D plots should now work correctly (used J indices instead of K).

Functions:

o Function SWIRL fixed.
o ENTROPY calculation scaling changed to include specific heat cv.

IRIS interface:

o Color table problem with IRIS fixed (only red was displayed due to a FORTRAN compiler bug).
o Worked on near clipping problem with z-buffering on IRIS (where z-buffering doesn't work when you're too close to the object).
o Device INPUTC changed to INPTCH on IRIS; commented out PARAMETER initialization of MODECH and INPTCH (these were due to changes in the GL2 FORTRAN include file "/usr/include/fgl.h".
o Mouse buttons should now work correctly under the window manager on the IRIS.
o NTSC display mode for IRIS now correctly maps to a subset of the screen.
o Better handling of keyboard interrupts during graphics initialization under MEX. (Would get message "no more gfports in system!!" twice when attempting another plot; would have to interrupt again. If a third plot was tried, system would hang.)

DISSPLA interface:

o MINMAX/xSCALE=scale for DISSPLA now works.

GRAFIX interface:

o 3D shading fixed (was incorrect if VPOINT was specified as (x,y,z)).
o Axis scaling from MINMAX/xSCALE=scale and PLOT/SURFACE fixed.

Known_problems

Listed are known or potential problems with various versions of PLOT3D.

IRIS

o Z-buffering an image with dashed, dotted, chaindashed, or chaindotted linestyles will hang the IRIS.
o Changing from single- to double-buffer mode or vice-versa (such as when turning z-buffering on or off), when using the window

manager and more than one plot is on the screen can cause the screen to be refreshed incorrectly. Move some windows slightly to force MEX to update the screen.

- o In general, and especially at the command prompt level, PLOT3D does not keep its graphics windows "idling" with a "swapbuffers" (see "Silicon Graphics PIPELINE", Vol. II, No. 1, Fall 1985, p. 12). Thus under the window manager (and double buffering), another graphics program will hang until PLOT3D swaps buffers.
- o For very complicated plots and/or particle trace plots, PLOT3D has been observed to respond VERY sluggishly. Response to the mouse may take on the order of a minute (keep that mouse button down!). Further screen updates may be somewhat faster.
- o PLOT3D will not update the screen correctly if the window manager has been started up using "mex -d" (double-buffered).
- o PLOT3D uses A LOT of (virtual) memory. IRISes at NASA Ames have been reconfigured with 24MB of swap space. This involves repartitioning the system disk (see the "IRIS Series 3000 Owner's Guide, Version 2.0" or your SGI rep.).
- o Under W3.4-GL2, support for the Seiko color printer screen dump may not exist, indicated by unsatisfied externals (isetcolormap, isetname,iclose,iflsbuf,putrow,iopen) when linking. In this case, disable the Seiko color printer screen dump by commenting out the call to SCRDMP in EXEQUE.
- o Under W3.5-GL2, the INCLUDE file "/usr/include/fgl.h" does not define the Graphics Library function ISMEX to be type LOGICAL. If, in a future release, this is fixed, all "LOGICAL ISMEX" declarations in IRIS2 will have to be commented out.

VMS

- o To link PLOT3D under VMS (with the current dimensions in MAIN), your page file quota must be something over 40000 pages. To run PLOT3D, it must be at least 20000 pages. You need special priviledge to change this:

```
$SD SYS$SYSTEM:
$RUN AUTHORIZE
MODIFY username /PGFLQUOTA=20000
```

COS

- o The CRAY/COS version of PLOT3D is really not worth much due to the high memory overhead in PLOT3D and the relatively small amount of physical memory on the XMP. See "HELP Installation_of_PLOT3D" for tips on reducing the memory requirements of PLOT3D.

DISSPLA

- o Divide-by-zero errors are sometimes generated by DISSPLA while making 2D contour plots with labels. The traceback may point to QQTLB(?), CONTUR, or ENDCO2. This is a DISSPLA internal bug. Try changing MINMAX or plotting without contour labels

("PLOT/NOLABELS")
- o Spurious lines have been observed on QMS and LN03 output plots. The cause is unknown.

CRAY_2

The CRAY 2 version of PLOT3D has many idiosyncrasies, due to the poor condition of the CFT77 compiler under UNICOS. Things have improved somewhat under UNICOS 2.0. Some of the remaining problems are:

- o The "$" format descriptor does not work, so all prompts end with a <RETURN>.
- o Multiple line messages, such as those printed by the by the SHOW command, are garbled.
- o ^D end-of-files are often not trapped correctly by the FORTRAN REA

D

statement. Type "EOF" instead (see "HELP End-of-file").
- o The utility programs CREHLPLIB and GETHELP do not work: the HELP Library file must be created on the IRIS.

APOLLO

- o Titles and text have not been implemented.
- o The C I/O routines (in the file CIO.C) have not been tested with PLOT3D; I don't know how to compile and link them; I've indicated something reasonable in the PLOT3DLINK file. They are needed for the READ/BINARY command (but not READ/UNFORMATTED), and for particle tracing. C routines are included because C is faster for reading and writing large chunks of data than FORTRAN, and the C binary files are pure binary, as opposed to FORTRAN unformatted files, which include record length information. This latter feature facilitates the transfer of binary data files from a dissimilar type machine such as a CRAY.
- o The FORTRAN compiler has a maximum number of lines of code that it will accept from one file.
- o GMR3D V1.0 does NOT support filled polygons. This is in beta-test for the next release of GMR3D.
- o BSP tree hidden surface removal is not supported or documented for GMR3D V1.0.
- o I could not get more than 8 colors out of GMR3D in single-buffered mode (but could from GPR). This is a possible GMR3D problem. Onl

y

8 colors are available in double-buffered mode, as documented.
- o Mouse buttons work strangely--you may have to click the button a second time to turn off the rotation, rather than just releasing the button. This is a possible GMR3D problem.
- o Keyboard attention interrupts are handled by PLOT3D using Aegis system calls. However, when interrupting a plot, the textport doe

s

not come back, but the "PLOT3D V3:" prompt does. This makes continued execution difficult.
- o Line styles and filled polygons (beta-test) don't work correctly o

n

the DN580--they originally come up correctly, but when the picture is complete (ready to be manipulated), dashed or dotted lines change to solid, and filled polygons disappear.

o When doing a "grow" or "pop" on the PLOT3D window, the viewport is cleared and redrawn. On the DN580, the viewport is incorrectly cleared to cyan. When the mouse buttons are used to rotate or translate the image (causing the viewport to be refreshed), the picture reappears. The refresh works correctly on a DN570.

o As a further note, when writing programs that write files, FORTRAN I/O requires that a maximum record length (RECL=#bytes) be specified on the OPEN statement when opening ANY new file. This is a BUG. This DOES affect PLOT3D, when doing particle traces. Routines PAR2D and PAR3D must be modified to specify the record length when opening the SCRATCH file. For the rest, current coding uses C I/O routines to do the particle trace reads and writes; equivalent FORTRAN I/O (but WITHOUT the RECL parameter) is commented out beside the C calls.

Special_characters

The following characters are treated specially when encountered on ANY line of input.

- If the last character on a line of input, "-" means to consider the next input line to be a continuation of the current line.
' Expected to surround a symbol, to be translated (by calling an appropriate system routine) into a character string.
" Expected to surround a character string, the contents of which are not to be disturbed. The string is treated as a single character argument.
! Denotes that all input following on this line is to be considered a comment and ignored.
@ If the first (nonblank) character on a line of input, the rest of the line is taken as a file name to be read as input to PLOT3D. When an end-of-file (or EOF) is encountered, input continues from the previous source.
EOF If the ONLY nonblank characters on an input line are EOF, this is equivalent to an end-of-file reading input. Further input is taken from the previous input file (or device).

Blanking

An array IBLANK can be included in the XYZ grid file which supplies PLOT3D with information on grid topology. This can be used to indicate that some points in the grid are "turned off" (so contours are not plotted through this region), or how multiple grids are connected (to aid in particle tracing). See "READ File_formats XYZ with_IBLANK" for information on including IBLANK in XYZ data files.

The use of IBLANK is as follows: where IBLANK is zero, the corresponding grid point is assumed turned off, and data at that point

(XYZ, Q) is NEVER USED. If IBLANK is anything but zero, data at that point is assumed valid. The NORMAL value for IBLANK in the interior of a grid is 1.

Other values are used for particle tracing, and indicate grid topologies: around holes (blanked-out regions) and grid boundaries, an IBLANK value of -n means that physical space "continues" in grid n; in other words, a particle trace leaving the current grid here will continue in grid n. Note that periodic boundaries or C-mesh-type boundaries can be coded the same way: for grid m, use IBLANK=-m on the common boundaries.

Another IBLANK key is the value 2, and indicates the location of a solid wall boundary. This is used to (artificially) restrict particle traces from flowing into the wall and stopping. Particles are forced to remain some fraction of a point away from a wall. This restriction is only implemented for particle tracing through a velocity field.

Attributes

Various attributes of graphical objects can be set in commands such as WALLS, SUBSETS, and CONTOURS. Examples of such attributes are color, line type and thickness, and surface transparency.

Hidden_lines

The HIDDEN_LINE attribute type is implemented as polygons filled with the background color, and outlined with the given line style. As a result, to generate a correct hidden line image, walls or subsets, as well as polygons within the walls or subsets must be ordered correctly (i.e., from back to front). Do this by setting the START,END,INCREMENT values appropriately in the WALLS or SUBSETS command, and/or split a wall or subset into two.

Shaded_surface

Implemented as filled grid cells, giving a faceted shading. Note that hidden surfaces are not by default implemented, so that for a correct hidden surface image, walls and subsets, as well as polygons within walls or subsets mustbe ordered correctly (i.e., from back to front). Do this by setting the START,END,INCREMENT values appropriately in the WALLS or SUBSETS command, and/or split a wall or subset into two.

Shading is done assuming that the light source is at the initial viewpoint (specified with the VPOINT command). The color is ramped linearly from 0.3 times the base color at 90 deg. incidence of the light sorce to the polygon normal vector (angle cosine of 0), to full color at a cosine of (1-shininess). As the cosine increases to 1, the color changes linearly from the base color to white.

Color

Colors may be entered as one of the eight standard (named) colors, BLACK, MAGENTA, RED, YELLOW, GREEN, CYAN, BLUE, and WHITE (D), or as a set of RGB (red-green-blue) components, in the range of 0 to 1. RGB colors are specified in the form "RGB .2 .3 .4".

Line_type

Line type (or style) can be SOLID (D), DASH, DOT, CHAINDASH, CHAINDOT, or NONE.

{DOT, CHAINDASH, and CHAINDOT are not implemented for GRAFIX.}

Line_thickness

This is a real number which multiplies the standard line thickness.

{Only factors of 1 or 2 are supported for IRIS 1000 series terminals and workstations. Not implemented for GRAFIX.}

Symbol_type

Specifies a symbol (by integer number) to be put at each point. Symbols are

-1	dot (single pixel)
0	no symbol (D)
1	+
2	x
3	square
4	circle
5	diamond
6	triangle (Delta)
7	triangle (nabla)
8	filled square
9	filled circle
10	filled diamond
11	filled Delta
12	filled nabla

{DISSPLA: Symbol -1 may not work; symbols 10-12 are implemented as unfilled.
GRAFIX : Symbols 1-12 are not implemented.
APOLLO : Symbol 3 is implemented as a *; symbols 5-12 are implemented as a circle.}

Symbol_size

Real number factor which multiplies the standard symbol size.

Shading_pattern

Shading pattern for polygon fills. Default is SOLID. {Others are

not yet implemented.}

Transparency

Real number between 0 (opaque) and 1 (completely transparent) controlling polygon transparency. {Not implemented for DISSPLA or GRAFIX.}
The HIDDEN_LINE type is flagged via a negative transparency.
A transparency value greater than 1 results in polygons being outlined and not filled.

Shininess

Shaded surface shininess; a real number between 0 and 1.

Shading is done assuming that the light source is at the initial viewpoint (specified with the VPOINT command). The color is ramped linearly from 0.3 times the base color at 90 deg. incidence of the light sorce to the polygon normal vector (angle cosine of 0), to full color at a cosine of (1-shininess). As the cosine increases to 1, the color changes linearly from the base color to white.

Material_coefficient

Polygon reflectivity. {Not implemented.}

@file

Use the given file as input for PLOT3D commands. On an end-of-file, control is returned to the previous source of input. This command may be entered at any time during a PLOT3D session, not necessarily at the main command level.

End-of-file

An end-of-file can be used in many commands to prematurely terminate a set of input, preventing that input from being incorporated into the program. Most computer systems have a control character for entering an "end-of-file" from the terminal; in VAX/VMS, it is ^Z. As an alternative, the letters "EOF" can be entered as an input line, signalling an end-of-file.

$system-command

If a "$" is the first character of a command line (at the main command level), the rest of the line is passed to the operating system as a command (for example $DIR when running under VAX/VMS would result in the "DIR" being sent to VMS). A "$" alone means return to the system for awhile. Upon completion (LOGOUT in VMS, ^D in Unix), return to PLOT3D. {Generally doesn't work for IRIS/Unix due to swap space

limitations.}

CLEAR

Format: CLEAR

Clears all internally computed arrays (such as functions) from the WORK storage area. In general, this is handled automatically.

CONTOURS

Format: CONTOURS [max number of levels]
or CONTOURS/INCREMENT [contour increment]
or CONTOURS/MANUAL [start[,end,increment]]

Select contour levels or ranges.

Qualifiers

/RANGE
Show the range (minimum and maximum) of the current function over all active points in all active grids.
/AUTOMATIC (D)
/INCREMENT
/MANUAL
For the default AUTOMATIC mode, a maximum number of contour levels to be calculated is specified. In INCREMENT mode, a contour level increment is specified. The number of levels will depend on the function range. MANUAL entry of contour levels allows start-end-increment sets to be given.
/ATTRIBUTES (D)
/NOATTRIBUTES
Controls whether attributes such as color range, line type and thickness, and surface transparency are prompted for. {Symbols do not appear for PLOT/2D/LABELS using DISSPLA.}

EXIT

Format: EXIT

Exit out of PLOT3D. Same as QUIT.

Qualifiers

/SAVE
Save the journal file. Default is to delete it.

FSURFACE

Format: FSURFACE

Set the following properties of function surface (or line) plots:
(1) the scale factor of the function axis;
(2) the origin (or offset) of walls drawn on the plot for reference; and
(3) whether a 3D function surface will be drawn as grid lines or contour lines, if the contour attribute type is LINES.

Other properties of a function surface plot are set by other commands:
(1) the VIEW command sets which spatial (x,y,z) axes will be plotted vs. the function;
(2) the CONTOUR command controls whether the function surface will be plotted as lines or a surface (using polygons), as well as the color map and line or surface attributes to be used;
(3) the MINMAX command sets the range of the function to be plotted (XMIN,XMAX, YMIN,YMAX, or ZMIN,ZMAX, depending upon which axis will be used for the function; and
(4) the /SURFACE qualifier on the PLOT command is used to select a function surface plot instead of the (default) /CONTOUR plot.

In 3D, a function surface is a plot of two spatial dimensions (x, y, or z) vs. a scalar function. This is also known as a carpet plot. Which two spatial dimensions are used is chosen using the VIEW command. A right-handed coordinate system is maintained; thus a plot with VIEW XZ would plot x vs. f vs. z, and the axis range for f would be controlled by YMIN and YMAX entered in the MINMAX command.

In 2D, a function surface can degenerate to a simple line plot, depending on the subset(s) active. A good example of a line plot is Cp (pressure coefficient) vs. x for an airfoil. Here the appropriate subset would be just the points on the airfoil surface. In any case, the plot is of one spatial dimension and the function value. (The spatial dimension used is the first axis given to the VIEW command. Currently this spatial dimension will be always be plotted on the horizontal axis, and the function along the vertical axis.)

Qualifiers

/SCALE_FACTOR=scale
/SCALE_FACTOR=AUTO (D)
Allows entry of the relative scaling of the function axis compared to the spatial (x, y, or z) axes. Thus a scale factor of 2 would mean that 1 unit of the scalar function will have the same length as 2 units in x, y, or z. Specifying "AUTO" for the scale factor means that the length of the function axis (from MINMAX) will be made equal to the longer of the other two axes (3D), or equal to

the other axis (2D).

/WALLS_ORIGIN=origin

/WALLS_ORIGIN=AUTO (D)

Allows specification of the origin along the function axis for the walls which are drawn on the function surface or line plot. For example, if a value of 5 was entered for a plot of x vs. y vs. f, a wall at coordinates (x,y,0) would be plotted at (x,y,5) in (x,y,f) space. Specifying "AUTO" means that the origin will be set to zero.

/CONTOUR

/GRID (D)

Controls whether a function surface will be drawn as grid lines or contour lines if the contour attribute type is LINES. This flag is ignored for 2D plots.

FUNCTION

Format: FUNCTION [function number]

Allows the selection of the function number to be plotted. Numbers between 0 and 99 are for grid-type information, between 100 and 199 are scalar functions (suitable for contour plots), and numbers between 200 and 299 are vector fields. Note that currently 200-299 refer to vector plots, while 300-399 generate particle trace-type plots. Functions from 400 on up are special functions (such as shock wave locations).

Grid_functions

0 Walls alone (geometry).
1 Grids.
2 Outline of IBLANK holes.

Scalar_functions

100 Density (or Q1).
101 Normalized density.
102 Stagnation density.
103 Normalized stagnation density.
110 Pressure.
111 Normalized pressure.
112 Stagnation pressure.
113 Normalized stagnation pressure.
114 Pressure coefficient.
115 Stagnation pressure coefficient.
116 Pitot pressure.
117 Pitot pressure ratio.
118 Dynamic pressure.
120 Temperature.
121 Normalized temperature.
122 Stagnation temperature.
123 Normalized stagnation temperature.

```
    130 Enthalpy.
    131 Normalized enthalpy.
    132 Stagnation enthalpy.
    133 Normalized stagnation enthalpy.
    140 (Internal) energy.
    141 Normalized (internal) energy.
    142 Stagnation energy.
    143 Normalized stagnation energy.
    144 Kinetic energy.
    145 Normalized kinetic energy.
    150 u velocity.
    151 v velocity.
    152 w velocity.
    153 Velocity magnitude.
    154 Mach number.
    155 Speed of sound.
    156 Cross-flow velocity.
    157 Normalized 2D stream function.
    160 x-momentum (Q2).
    161 y-momentum (Q3).
    162 z-momentum (Q4).
    163 Stagnation energy per unit volume (Q5).
    170 Entropy.
    171 Entropy measure s1.
    180 x-component of vorticity.
    181 y-component of vorticity.
    182 z-component of vorticity.
    183 Vorticity magnitude.
    184 Swirl.
    185 Velocity x vorticity magnitude.
    190 Shock function based on pressure gradient.


Vector_functions

    200 Velocity.
    201 Vorticity.
    202 Momentum (Q2,Q3,Q4).
    203 Perturbation velocity.
    204 Velocity x vorticity.
    210 Pressure gradient.


Particle_trace_functions

    300 Particle traces.
    301 Vortex lines.

    Particle traces are generated using trilinear interpolation of
    values of the vector function inside a computational cell, and
    second-order Runge-Kutta steps to advance the particle in space.
    The particles are advanced using the velocity (or vector function)
    in real-space, but the step is limited so the particle will
    only advance some fraction of a computational cell.  Currently,
    five steps are taken per computational cell.
```

See the RAKE command for specification of starting points and attributes for the traces.

Shock_waves

400 Shock locations based on pressure gradient.

The current algorithm for "finding shocks" is to look at the Mach number component in the direction of the local pressure gradient. Where this value goes through one, AND the Mach number is decreasing, is plotted as a shock. See "FUNCTION Function_definitions Shock_function" for additional information.

The way that the shock structure is plotted is determined by the FIRST contour specification (attributes, not level). Thus one can specify LINES (and therefore plane orientations) or SURFACES, color, line thickness, surface transparency, etc.

Nondimensionalizations

Certain reference conditions are assumed in calculating some functions (the pressure coefficient, for example). These conditions are set in subroutines SCAFUN and VECFUN.

(1) $\rho_{inf} = 1$ (freestream density)

(2) $c_{inf} = 1$ (freestream speed of sound)

(3) $p_{inf} = 1/\gamma$ (freestream pressure)

(4) $|V|_{inf} = M_{inf} * c_{inf}$ (freestream velocity magn.)

Conditions 3 and 4 follow from 1 and 2.

Fluid_constants

The fluid is assumed to be air. The perfect gas law is also used. The following constants are used in computing functions, and are defined in BLOCK DATA BFLUID.

gamma = 1.4 (ratio of specific heats)

R = 1 (gas constant)

Function_definitions

Density

```
rho    = Q1

rhoinf = 1

                 [      gamma-1          ]**[1/(gamma-1)]
rho0   = rho * [ 1 + ------- * M**2 ]
                 [         2             ]
```

Pressure

```
p    = (gamma-1) * rho * [e0-(1/2)*V**2]

pinf = 1/gamma

               [      gamma-1          ]**[gamma/(gamma-1)]
p0     = p * [ 1 + ------- * M**2 ]
               [         2             ]
```

Pressure_coefficient

```
Cp  = (p-pinf)/(.5*rhoinf*Vinf**2)

Cp0 = (p0-p0inf)/(.5*rhoinf*Vinf**2)

      [ 2 + (gamma-1)*Minf**2 ]**[gamma/(gamma-1)]
      [-----------------------]                    - 1
      [        gamma+1        ]
Cp* = ---------------------------------------------------
                        gamma
                        ----- * Minf**2
                          2
```

Ref: A.H. Shapiro, THE DYNAMICS AND THERMODYNAMICS OF COMPRESSIBLE FLOW, The Ronald Press Company, New York, 1953, Vol. 1, p. 107.

Pitot_pressure

For M < 1, the pitot pressure is the same as the stagnation pressure. Above Mach 1, it is the equivalent stagnation pressure behind a normal shock:

```
pp = p0y = (p0y/px)*px   (x is before shock, y is after)

pitot pressure ratio = pp/pinf

p0y/px = (p0y/p0x)*(p0x/px)

               [ (gamma+1)          ]**[gamma/(gamma-1)]
               [ --------- * Mx**2 ]
               [     2              ]
```

$$= \frac{}{\left[\frac{2\cdot\gamma}{\gamma+1} \cdot Mx^2 - \frac{\gamma-1}{\gamma+1}\right]^{[1/(\gamma-1)]}}$$

Ref: A.H. Shapiro, THE DYNAMICS AND THERMODYNAMICS OF COMPRESSIBLE FLOW, The Ronald Press Company, New York, 1953, Vol. 1, p. 119.

Dynamic_pressure

$q = (1/2)*rho*V**2 = kinetic energy/unit volume$

Temperature

$T = p/(rho*R)$ (perfect gas law)

$T/Tinf = (p/pinf)/(rho/rhoinf)$

$$T0 = T * \left[1 + \frac{\gamma-1}{2} * M^2\right]$$

Enthalpy

$h = gamma * [e0 - (1/2)*V**2]$
$= gamma * ei$
$= [gamma/(gamma-1)] * (p/rho)$

$hinf = gamma*e0inf$
$= [gamma/(gamma-1)] * (pinf/rhoinf)$

$h0 = h + (1/2)*V**2$
$= gamma*e0 - (gamma-1)*(1/2)*V**2$
$= e0 + p/rho$

Energy

$ei = e0 - (1/2)*V**2$
$= [1/(gamma-1)] * (p/rho)$

$eiinf = [1/(gamma-1)] * (pinf/rhoinf)$

$e0 = Q5/rho$

$e0inf = [1/(gamma-1)]*(pinf/rhoinf) + (1/2)*Vinf**2$

$ek = (1/2)*V**2$
$= q/rho$

Mach_number

M = V/c

c = sqrt(gamma*p/rho)

Cross-flow_velocity

Vcross-flow = sqrt(v**2+w**2)

2D_stream_function

Calculated by integrating the mass flow across a coordinate line. THE STREAM FUNCTION IS ASSUMED TO HAVE THE VALUE ZERO AT THE FIRST POINT IN EACH SUBSET. The values are normalized by the freestream Mach number. Only x and y, and u and v are used! See subroutine STREAM.

Momentum

Momentum = (Q2,Q3,Q4)

Entropy

$$s = \frac{c}{v} * \ln\left[\frac{p}{pinf}\right] + \frac{c}{p} * \ln\left[\frac{rhoinf}{rho}\right]$$

$$= \frac{c}{v} * \ln\left[\frac{(p/pinf)}{(rho/rhoinf)^{**}gamma}\right]$$

$$s1 = \left[\frac{(p/pinf)}{(rho/rhoinf)^{**}gamma}\right] - 1 \text{ , another measure of entropy}$$

Vorticity

$$omega_1 = xi\ w + eta\ w + zeta\ w - xi\ v - eta\ v - zeta\ v$$
$$y\ xi \quad y\ eta \quad y\ zeta \quad z\ xi \quad z\ eta \quad z\ zeta$$

$$= d(w)/d(y) - d(v)/d(z)$$

$$omega_2 = xi\ u + eta\ u + zeta\ u - xi\ w - eta\ w - zeta\ w$$
$$z\ xi \quad z\ eta \quad z\ zeta \quad x\ xi \quad x\ eta \quad x\ zeta$$

$$= d(u)/d(z) - d(w)/d(x)$$

$$omega_3 = xi\ v + eta\ v + zeta\ v - xi\ u - eta\ u - zeta\ u$$
$$x\ xi \quad x\ eta \quad x\ zeta \quad y\ xi \quad y\ eta \quad y\ zeta$$

= d(v)/d(x) - d(u)/d(y)

Swirl

Swirl = (omega dot V)/(rho*V**2)

Shock_function

The shock function based on pressure gradient, of which some parts of the level 1 contour are the shock waves, is computed as follows, where V is the velocity vector:

$$\text{Shock function} = \frac{V}{c} \cdot \frac{\text{grad}(p)}{|\text{grad}(p)|}$$

There may be a switch which sets the shock function to zero if |grad(p)| < 0.1. This avoids picking up preshock oscillations (see subroutine SHOCKP). It is agreed that this is not the proper place to do this. Another approach is to put the test in the contour finding/testing routines (SHK2L, SHK2LB,SHK3L,SHK3LB,SHK3S), but this requires carrying the pressure gradient along.

Perturbation_velocity

V' = V - Vinf

LIST

Format: LIST [XYZ or Q or FUNCTION]

List the XYZ, Q, or current function data. Output can be directed to the screen (default) or to a file. FORMATTED, UNFORMATTED, or BINARY files suitable for reading into PLOT3D can be produced as well.

Qualifiers

/TEXT (D)
/FORMATTED
/UNFORMATTED
/BINARY
Select the type of LIST output desired. TEXT output includes column headings and suitable for viewing on the screen or printing out. FORMATTED, UNFORMATTED, and BINARY options produce files which can be read into PLOT3D using the READ command.
/OUTPUT=file
Redirect the list output to a file or device. A filename is required for FORMATTED, UNFORMATTED, or BINARY lists, and will be prompted for in case the /OUTPUT qualifier has not been included in the command

line.

MAP

Format: MAP

Produce a map of WORK array usage, including grid number, variable names, and source file names.

MINMAX

Format: MINMAX [xmin,xmax,ymin,ymax[,zmin,zmax]]
or MINMAX/INCREMENT [xmin,xmax,xinc,ymin,ymax,yinc[,zmin,zmax,zinc]]

Set the plot ranges, or general region of interest for the plot.

Qualifiers

/X (D)
/NOX
/Y (D)
/NOY
/Z (D)
/NOZ
Controls which set of axis limits are to be changed. To change only the y-axis limits, for instance, type "MINMAX/Y ymin,ymax".
/INCREMENT
Allows the specification of a tick mark increment. An increment of zero implies automatic scaling of tick marks and possibly rounding of axis limits. (Axis drawing can be suppressed by using PLOT/NOAXES.)
/XSCALE=scale
Sets x-axis scale factor. Default is one.
/YSCALE=scale
Sets y-axis scale factor. Default is one.
/ZSCALE=scale
Sets z-axis scale factor. Default is one.

PLOT

Format: PLOT

Initiate a plot. When the plot is completed, type <RETURN> to return to PLOT3D command level.

Qualifiers

/2D
/3D (D)

```
  Type of plot.  Note that 2D data can be displayed in a 3D plot too.
/TEKTRONIX (D for DISSPLA)
/TK41XX
/ENVISION
/VERSATEC
/DIP
/QMS
/LN03
/IMAGEN
/DICOMED
/COMPRESS
/IRIS (D for IRIS)
/IRISRGB
/IRISSINGLE
/GRAFIX (D for GRAFIX)
/APOLLO (D for Apollo)
/APOLLOSINGLE
  Output device.

  DISSPLA ONLY devices: TEKTRONIX (4014) (default)
                        TK41XX (Tektronix 41xx series terminals--
                           will prompt for terminal type)
                        ENVISION (215)
                        VERSATEC (connected to a VAX; produces
                           files PARM.PLV and VECTR1.PLV -- see
                           $HELP PLOT for info on submitting to
                           Versatec)
                        DIP (NASA Ames Device Independent Plot file;
                           produces file xxx.DIP*)
                        QMS (QMS 1200 laser printer; produces file
                           xxx.QMS*)
                        LN03 (DEC LN01S or LN03 Plus laser printer;
                           produces file INTSCRT1.TMP)
                        IMAGEN (IMAGEN 8/300 laser printer;
                           produces file IMAGEN.IMP)
                        DICOMED (D48 film recorder; produces file
                           xxx.D48*)
                        COMPRESS (DISSPLA device independent file
                           to be used with POP; produces file
                           POPFIL.DAT)

     (On the CRAY, only DIP and DICOMED are available; at NASA Ames,
      use the COS JCL
        AMESLIB.
        DISSPLA,HELP.
      for information on DISSPLA output files.)

  IRIS ONLY devices:    IRIS (double buffered) (default)
                        IRISSINGLE (single buffered)
                        IRISRGB (RGB mode)

  GRAFIX ONLY devices:  GRAFIX (produces xxx.GRA*) (default)

  APOLLO ONLY devices:  APOLLO (double buffered) (default)
                        APOLLOSINGLE (single buffered)
```

* For those devices that generate a file (such as DIP), the base filename used will be the Q filename (or the XYZ filename, or simply "PLOT3D") with an appropriate three letter extension (such as ".DIP"). Exceptions are PLOT/COMPRESS, which produces a file "POPFIL.DAT", PLOT/LN03, which generates "INTSCRT1.TMP", and PLOT/IMAGEN, which generates "IMAGEN.IMP".

If the device specified does not apply to the version of PLOT3D being run, the default for that version is used instead.

/CONTOUR (D)
/SURFACE
/CARPET
/LINE

If this is a plot of a scalar variable, set whether the plot will be a CONTOUR plot (the default) or a function SURFACE plot. A CARPET plot is another common term for a function surface, and in 2D a function surface degenerates to a LINE plot; thus CARPET and LINE are synonyms for SURFACE. See HELP FSURFACE for more information on function surface plots.

/FULLSCREEN
/NOFULLSCREEN (D)

Allows for an enlarged plotting area to be used, at the expense of titles and text. Very helpful for 3D plots. (For the IRIS or GRAFIX, FULLSCREEN simply removes all text from the display.)

/AXES (D)
/NOAXES

Controls whether axes are put on the plot or not.

/FIGURE=(areax,areay,charht)
/NOFIGURE (D)

For /FIGURE, the plot size and character height is specified (in inches), title, contour bar, and additional text are not plotted, and other measures are taken to pretty up the plot. (Most of this can not be done for the IRIS or GRAFIX.)

/BACKGROUND=color

Set the background color of the plot. Note that if "color" is of the form "RGB r,g,b", it must be enclosed in parentheses (e.g., /BACKGROUND=(RGB .5,.5,.5)) so the entire string is associated with the /BACKGROUND qualifier. The default is BLACK.
(Not implemented for DISSPLA. The display program (e.g. GAS) sets the background color for GRAFIX.)

/UP=axis

Specify which axis will be considered generally vertical for a plot. Valid "axes" are X, Y, Z, +X, +Y, +Z, -X, -Y, or -Z. (A right-handed system is always assumed.) A viewpoint specified in spherical coordinates (VPOINT/ANGLES) sets the angles phi, in the horizontal plane, and theta, above the plane. The default is /UP=Z for a 3D plot, /UP=Y for 2D.

Note that for a 2D plot, the "y-axis" really means the second axis on the plot: if VIEW XZ had been selected, /UP=Y would put the Z-valued axis up. Similarly, if VIEW ZX had been selected, /UP=Y would put the x-axis up! There is the possibility of confusion here!

Example: We have a 2D airfoil, and would like velocity profiles with u horizontal and y vertical. We select proper subsets and function, type "VIEW YX" so the line plot will use Y as the spatial axis,

"MINMAX 0 .2 -.5 1" so y will range from 0 to 0.2, u from -0.5 to 1. The we say "PLOT/2D/LINE/UP=X": we're making a 2D line plot, and we want the axis which WOULD have been horizontal (and corresponds to the first set of MINMAX values) to be "up". Thus y will be up, u horizontal.

/SCRIPT
Further input will be interpreted as SCRIPT commands for controlling the orientation of the display on the screen. STOP will signify the end of SCRIPT commands. See PLOT SCRIPT for information on these commands. (Not implemented for DISSPLA or GRAFIX.)

IRIS_interface

PLOT3D will run with or without the IRIS window manager MEX. Under MEX, PLOT3D can open multiple graphics ports. By default, each plot is NOT deleted when a <RETURN> is entered and the program returns to the PLOT3D command prompt. Subsequent plots are displayed as additional graphics ports connected to PLOT3D. Mouse and keyboard input can be attached to the different ports by using the window manager "attach" menu item while mouse and keyboard manipulation is active in PLOT3D (i.e., while a picture is being displayed).

To delete a graphics window, attach to the window (again, while a picture is being displayed, NOT in PLOT3D command mode) and type <DELETE>.

To make different graphics ports have the same viewpoint:
(1) attach to the port with the desired viewpoint,
(2) type "=", and
(3) attach to the port(s) to be modified.

NOTE: a "hogwhiledown" keyboard button can be identified in the user's ".mexrc" file to allow attaching to different PLOT3D graphics windows. The .mexrc file lines

```
reservebut 13
bindfunc hogwhiledown 13
```

bind the "no scroll" key to the hogwhiledown function. For more information on the .mexrc file and hogwhiledown, read the Window Manager section in the IRIS User's Guide.

Mouse

On the IRIS, the mouse can be used to manipulate the picture on the graphics screen. The mouse buttons are used to activate different degrees of freedom for manipulation, while the mouse (x,y) displacement controls the RATE of motion in the respective degree of freedom.

While the left mouse button is being depressed, x-motion of the mouse is associated with rotation (about the center of the MINMAX box) in phi, while y-motion controls rotation in theta. (Phi and theta are cylindrical coordinate angles, using the DISSPLA convention.)

The middle mouse button activates zooming (toward the center of the

MINMAX box), controlled with the y-motion of the mouse. X-motion is not used.

Depressing the right button allows the picture to be dragged left or right (x-motion of the mouse), and up or down (y-motion) on the screen. REMEMBER, the POSITION of the mouse controls the RATE of motion (speed) of the object.

The sensitivity of the mouse to zooming and screen dragging is a function of the distance of the VPOINT initial viewpoint from the center of the MINMAX box. The closer the viewpoint to the MINMAX box, the less sensitive the mouse. While the plot is on the screen, this can be adjusted by typing "up-arrow" (to increase the sensitivity by a factor of 2) or "down-arrow" (to decrease it).

More than one button can be depressed at one time. The mouse position will control the rates associated with the last button pushed, while the motions associated with the other buttons are maintained, as long as they remain depressed.

Type <RETURN> to return to PLOT3D command level.

Seiko_color_printer_screen_save

Type "S" when the picture on the screen is what you would like to save.

In any mode, a file "xxx.N.sei" will be created, where xxx is the Q filename (or XYZ filename, or simply "PLOT3D"), and N is the frame number (starting from 1 for each PLOT3D run). These files can be printed by typing the Unix command "lp -dseiko xxx.N.sei".

This screen save option is a fixed-up version of the routine SCRSAVE supplied by Silicon Graphics under W3.5-GL2 and documented in the GL2-W3.5 Release Notes, p. 4-38. It still references routines in the library "/usr/people/gifts/mextools/imglib/libimage.a".

RAS_screen_save

On the IRIS, type "P" when the picture on the screen is what you would like to save. Saving the screen image takes a few minutes; the terminal will beep and a message ("Screen image saved as frame n in file xxx.ras") will be printed when it is done. The filename used is the Q filename (or XYZ filename, or simply "PLOT3D") with a .RAS extension. Once the image has been saved, the picture can again be manipulated with the mouse.

One file is opened per PLOT3D run. Frames are added to the existing file, if there is one (so we don't delete it by creating a new one!).

Screen images can be restored by running the program RASP.

The ARCGraph GRAFIX RAS file format is used. See the file "arcgraph/graifs.doc", ARCGraph Internal Functional Specifications

for a description of the format.

Screen_dump_to_QMS

On the IRIS, type "Q" when the picture on the screen is what you would like to save. Saving the screen image takes a few minutes; the terminal will beep and a message ("Screen image saved as page n in file xxx.qms") will be printed when it is done. The filename used is the Q filename (or XYZ filename, or simply "PLOT3D") with a .QMS extension. This is intended for line drawings only--anything that is not the background color (usually black) will be colored black on the QMS (where the background is, of course, white).

Once the image has been saved, the picture can again be manipulated with the mouse.

A new file is opened for every PLOT3D run. Thus if the file already exists, it is deleted!

Backfacing_polygon_removal

On the IRIS (2000 or 3000 series), typing "B" toggles the removal of backfacing polygons on or off. The default is off. See the IRIS User's guide for a description of backface polygon removal and its uses. Note that this is not an exact procedure, and some polygons may not get properly removed due to screen resolution. (The polygon normal is determined in screen coordinates.)

This is a new feature, and as such is not yet polished. The clockwise or counterclockwise ordering of polygon points in walls can be changed by using a negative increment for one of the indices. This is true for shocks and contour surfaces as well (I believe), by using a negative increment in the subset definition. Point ordering for color contours does NOT change with the sign of the increment, however.

Z-buffering

***** WARNING ***** Under GL2-W3.5 (as of 18 March 1987), z-buffering does not work for any displays using nonsolid linestyles! The machine may crash.

IRIS 2000 or 3000 series machines with 28 or more bitplanes support hardware z-buffering to generate hidden surface pictures. Type "Z" to toggle z-buffering on and off. The default is off. The MINMAX command should be set to include everything of importance, as the near and far clipping planes are activated with z-buffering, and are set to include somewhat more than the volume specified by the MINMAX command.

Z-buffering cannot be done from RGB mode (PLOT/IRISRGB); also, if the display is in double buffer mode, a side-effect of typing "Z" is that the display mode will change to single buffer mode. The

mode will change back when z-buffering is turned off.

Parameters IZNEAR and IZFAR must be set correctly in subroutine IRIS.

Script

These commands, when entered following a PLOT/SCRIPT command, control the motion of the objects on the display screen. (Not implemented for DISSPLA or GRAFIX.)

CENTER

Format: CENTER [x,y,z]

Move the center of the display (equivalent to the center of the MINMAX box) to the given (x,y,z) point. Objects on the screen will rotate about this point.

Qualifiers

/FROM=(x,y,z)
Specify what point to move the center from. Default is the previous CENTER, or center of the MINMAX box.
/IN=n
Move the center to the new point in n frames.

GO

Format: GO [number of frames]

"GO n" means advance the display n frames. The default value for n is one.

HOLD

Format: HOLD [number of frames]

"HOLD n" means hold the display still for n frames. Default value for n is one.

INTERACTIVE

Format: INTERACTIVE

Temporarily enable interactive manipulation of the display. Typing <RETURN> returns control to the SCRIPT input. See

"PLOT Mouse" for a description of interactive control of the display.

STOP

Format: STOP

Signals the end of SCRIPT commands.

VPOINT

Format: VPOINT [x,y,z]
or VPOINT/ANGLES [phi,theta,radius]

Change the viewpoint to the given (x,y,z) or (phi,theta,radius) position. Viewpoint is always looking toward the CENTER point.

Qualifiers

/ANGLES
Viewpoint (as well as FROM viewpoint, if supplied) is in (phi,theta,radius) spherical coordinates. Default is (x,y,z) cartesian coordinates.
/FROM=(x,y,z)
Viewpoint from which to move. Specified as (x,y,z) or (phi,theta,radius), depending on /ANGLES. Default is to move from previous viewpoint.
/IN=n
Move the viewpoint to the new viewpoint in n frames.

QUIT

Format: QUIT

Exit out of PLOT3D. Same as EXIT.

Qualifiers

/SAVE
Save the journal file. Default is to delete it.

RAKES

Format: RAKES

Specify the starting locations and attributes of particle traces (and

vortex lines, both of which are traced through a vector field).

Qualifiers

/IJK (D)
/XYZ
Traces can be started at (i,j,k) grid points or at (x,y,z) physical coordinates.
/ADD
Rake specifications are to be added to the end of existing rake specs. This does not work when the particle traces are read in with /READ=file.
/ATTRIBUTES (D)
/NOATTRIBUTES
Controls whether attributes such as color, line type and thickness, and surface transparency are prompted for.
/READ=file
/WRITE=file
Particle traces can be written out or read in to PLOT3D, allowing traces to be replotted later, or calculated elsewhere. See "File_format" for information on file structure. The default file type is .PAR.
/+TIME (D)
/-TIME
/+-TIME
Controls whether traces in this rake are followed in the plus, minus, or both directions of the vector field.
/MAXPOINTS=n
Set the maximum number of points for each particle trace segment, allowing a crude control on the length of the trace. Each trace MIGHT be made up of more than one segment, as a new segment is started when a trace continues in another grid. Currently, a particle takes 5 steps to cross a computational cell; therefore, setting /MAXPOINTS=15 means that each trace will be roughly 3 cells long.

The default value (and maximum) is /MAXPOINTS=1000.
/SCALAR_FUNCTION=function number
/NOSCALAR_FUNCTION (D)
Sets the function to be used to determine the color of the traces. Each trace will be colored according to the value of this function (which must be a scalar function) and the color map(s) specified using the CONTOUR command. If /NOSCALAR_FUNCTION is selected, the color attribute color will be used. This is the default.

Restriction

Traces can be restricted to remain in computational planes. This is useful for plotting particle traces on symmetry planes or for simulating oil flow. For oil flow pictures, traces are often restricted to remain one computational plane above the surface (as the velocity on the surface is zero).

Valid restrictions are IJ, IK, or JK.

File_format

Trace files are read and written using FORTRAN UNFORMATTED I/O (BINARY on IRIS/Unix) as follows:

```
   10 CONTINUE
      READ(unit,END=20) IRAKE,ITRACE,N,IGRID
      READ(unit) (X(I),I=1,N),(Y(I),I=1,N),(Z(I),I=1,N),(T(I),I=1,N),
     C           (RI(I),I=1,N),(RJ(I),I=1,N),(RK(I),I=1,N)
      ...
      GOTO 10
   20 CONTINUE
```

IRAKE is the rake number, and will be used to determine the attributes of the trace when it is drawn. ITRACE is the trace number inside this rake, and N is the number of points to be read in the following record. IGRID is the grid number for this part of the trace. The physical location (x,y,z) and time t is recorded for each point in the trace, as well as the (i,j,k) coordinates, saved as real numbers to indicate both the cellnumber and position within the cell of the point.

A trace may appear in the file as several segments, all with the same rake and trace number.

READ

Format: READ

Read input data files.

Qualifiers

/1D
/2D
/3D (D)
  Controls whether the input files are expected to be 1-, 2-, or 3-dimensional.
/XYZ=file (D)
  Input standard grid file.
/Q=file (D)
  Input standard Q file, including density, momentum, and stagnation energy per unit volume.
/FUNCTION=file
  Input an arbitrary function. {Currently there is no way to plot an arbitrary function.}
/MDATASET
  For multiple datasets in one file. {Not implemented.} (Intended to support time-varying solutions.)
/MGRID
  If the files contain multiple grids, in the multiple grid form.

/FORMATTED
/UNFORMATTED (D)
/BINARY (D for Unix)
Controls whether the files are expected to be in FORTRAN FORMATTED or UNFORMATTED form. For character form, free-formats (READ(unit,*)) are used. Under IRIS/Unix, I/O is done much more efficiently using C and straight BINARY files (which have no record information). Thus BINARY is the default under Unix. (Note that BINARY I/O can be done from FORTRAN too, using OPEN(...,FORM='BINARY').)
/PLANES
/WHOLE (D)
Specifies whether the data is to be read in planes (one plane per record) or the whole thing at once. See "READ File_formats" for some examples of reads with /PLANES and /WHOLE. Note that /PLANES has no effect for reading other than 3D datasets.
/CHECK (D)
/NOCHECK
Controls whether the default checking of Q data for zero or negative density or pressure is performed. Bypassing this check can speed up the Q file reading significantly.
/JACOBIAN
/NOJACOBIAN (D)
Indicates whether Q variables are scaled (divided) by the determinant of the metric Jacobian, in which case this determinant is expected as an additional Q variable. {This option may disappear, leaving the default of NOJACOBIAN.}
/BLANK
/NOBLANK (D)
Signals whether the XYZ file contains an integer array IBLANK as a fourth (or rather n-dimension plus one) variable. At (i,j,k) points where IBLANK=0, data will NEVER BE USED.

File_formats

XYZ

Listed are some samples...

Single_grid

1D:

```
      READ(unit) IDIM
      READ(unit) (X(I),I=1,IDIM)
```

2D:

```
      READ(unit) IDIM,JDIM
      READ(unit) ((X(I,J),I=1,IDIM),J=1,JDIM),
     C           ((Y(I,J),I=1,IDIM),J=1,JDIM)
```

3D (/WHOLE):

```
      READ(unit) IDIM,JDIM,KDIM
      READ(unit) (((X(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     C           (((Y(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
     C           (((Z(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
```

```
      3D (/PLANES):
            READ(unit) IDIM,JDIM,KDIM
            DO 10 K= 1,KDIM
               READ(unit) ((X(I,J,K),I=1,IDIM),J=1,JDIM),
           C               ((Y(I,J,K),I=1,IDIM),J=1,JDIM),
           C               ((Z(I,J,K),I=1,IDIM),J=1,JDIM)
          10    CONTINUE


   with_IBLANK

      3D (/WHOLE):
            READ(unit) IDIM,JDIM,KDIM
            READ(unit) (((X(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
           C           (((Y(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
           C           (((Z(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM),
           C      (((IBLANK(I,J,K),I=1,IDIM),J=1,JDIM),K=1,KDIM)
      3D (/PLANES):
            READ(unit) IDIM,JDIM,KDIM
            DO 10 K= 1,KDIM
               READ(unit) ((X(I,J,K),I=1,IDIM),J=1,JDIM),
           C               ((Y(I,J,K),I=1,IDIM),J=1,JDIM),
           C               ((Z(I,J,K),I=1,IDIM),J=1,JDIM),
           C          ((IBLANK(I,J,K),I=1,IDIM),J=1,JDIM)
          10    CONTINUE


   Multiple_grid

      3D (/WHOLE):
            READ(unit) NGRID
            READ(unit) (IDIM(IGRID),JDIM(IGRID),KDIM(IGRID),IGRID=1,NGRID)
            DO 10 IGRID= 1,NGRID
               READ(unit)
           C (((X(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID)),K=1,KDIM(IGRID)
),
           C (((Y(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID)),K=1,KDIM(IGRID)
),
           C (((Z(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID)),K=1,KDIM(IGRID)
)
          10    CONTINUE
       3D (/PLANES):
            READ(unit) NGRID
            READ(unit) (IDIM(IGRID),JDIM(IGRID),KDIM(IGRID),IGRID=1,NGRID)
            DO 10 IGRID= 1,NGRID
            DO 10 K= 1,KDIM(IGRID)
               READ(unit)
           C ((X(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID)),
           C ((Y(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID)),
           C ((Z(I,J,K),I=1,IDIM(IGRID)),J=1,JDIM(IGRID))
          10    CONTINUE


Q

   Listed are some samples... The Q header information (extra
```

information at the beginning of each grid) includes the following:

```
Freestream Mach number (FSMACH)
Angle-of-attack        (ALPHA)
Reynolds number        (RE)
Time (not used)        (TIME)
```

Single_grid

```
(without Jacobian)
1D:
      READ(unit) IDIM
      READ(unit) FSMACH,ALPHA,RE,TIME
      READ(unit) ((Q(I,NX),I=1,IDIM),NX=1,3)
2D:
      READ(unit) IDIM,JDIM
      READ(unit) FSMACH,ALPHA,RE,TIME
      READ(unit) (((Q(I,J,NX),I=1,IDIM),J=1,JDIM),NX=1,4)
3D (/WHOLE):
      READ(unit) IDIM,JDIM,KDIM
      READ(unit) FSMACH,ALPHA,RE,TIME
      READ(unit) ((((Q(I,J,K,NX),I=1,IDIM),J=1,JDIM),K=1,KDIM),NX=1,
5)
3D (/PLANES):
      READ(unit) IDIM,JDIM,KDIM
      READ(unit) FSMACH,ALPHA,RE,TIME
      DO 10 K= 1,KDIM
         READ(unit) (((Q(I,J,K,NX),I=1,IDIM),J=1,JDIM),NX=1,5)
   10    CONTINUE
```

Multiple_grid

```
(without Jacobian)
3D (/WHOLE):
      READ(unit) NGRID
      READ(unit) (IDIM(IGRID),JDIM(IGRID),KDIM(IGRID),IGRID=1,NGRID)
      DO 10 IGRID= 1,NGRID
         READ(unit) FSMACH,ALPHA,RE,TIME
         READ(unit)
       C ((((Q(I,J,K,NX),I=1,IDIM(IGRID)),J=1,JDIM(IGRID)),K=1,KDIM(IG
RI
       C                    NX=1,5)
   10    CONTINUE
3D (/WHOLE):
      READ(unit) NGRID
      READ(unit) (IDIM(IGRID),JDIM(IGRID),KDIM(IGRID),IGRID=1,NGRID)
      DO 10 IGRID= 1,NGRID
         READ(unit) FSMACH,ALPHA,RE,TIME
         DO 10 K= 1,KDIM(IGRID)
            READ(unit)
       C    (((Q(I,J,K,NX),I=1,IDIM(IGRID)),J=1,JDIM(IGRID)),NX=1,5)
   10       CONTINUE
```

SHOW

Show the current status of some commands. Applicable commands are CONTOURS, FSURFACE, FUNCTION, MINMAX, PLOT, RAKES, SUBSETS, TEXT, VECTORS, VIEW, VPOINT, and WALLS.

SUBSETS

Format: SUBSETS

Allows parts of grids to be selected as "active".

Qualifiers

/GRID=grid number
Specifies which grid these subsets are to be associated with (default is grid 1).
/ADD
These subsets are to be added on to the current list, rather than starting from subset number 1.
/ATTRIBUTES
/NOATTRIBUTES (D)
Controls whether attributes such as color, line type and thickness, and surface transparency are prompted for. Subset attributes are applied to GRID plots. {In the future, if NOATTRIBUTES is specified for certain types of plots, subset attributes would be used instead.}

TEXT

Format: TEXT ["line 1"[,"line 2"]]

Allows the entry of up to two additional lines of text (besides the title), to be centered above future plots. This text will continue to be used until another TEXT command is given. Current text can be "erased" by entering a <RETURN> in response to the prompt for text. (Text is prompted for if not entered on the command line.) If text is entered on the command line, be sure to enclose each line in double quotes.

VECTORS

Format: VECTORS

Sets vector plot characteristics, such as arrowheads, vector length, and associated scalar function for vector colors.

Qualifiers

/SCALAR_FUNCTION=function number
/NOSCALAR_FUNCTION (D)
Sets the function to be used to determine the color of the vectors. Each vector will be colored according to the value of this function (which must be a scalar function) and the color map(s) specified using the CONTOUR command. If /NOSCALAR_FUNCTION is selected, the color attribute color will be used. This is the default.
/ATTRIBUTES (D)
/NOATTRIBUTES
Sets various vector attributes. NOATTRIBUTES resets all everything to default values. Attributes are:

Vector length scale factor	- default is 1 (a vector function of unit length will be drawn as one unit long.
Vector scaling	- whether or not the vector is scaled with the magnitude. Default is scaled.
Arrowhead type	- LINES (D), FILLED, or NONE.
Arrowhead size factor	- default value of 1 corresponds to 3/10ths the vector length for scaled arrowheads, 1/50th the longest axis if not scaled. For DISSPLA, only size factors of 1-6 are supported. The default is 1.
Arrowhead scaling	- whether or not the arrowhead is scaled with the vector length. Default is scaled. In DISSPLA, arrowhead scaling is NOT supported.
Line type	-
Line thickness	-
Symbol type	- causes a symbol to be put at the base of the vector.
Symbol size	-
Shading pattern	- for filled arrowheads.
Shading parameters	-

See HELP ATTRIBUTES for information on line type, etc.

VIEW

Format: VIEW [view]

The VIEW command is used for two cases: (1) to specify which axes are to be horizontal and vertical for a 2D plot, and (2) to specify which two spatial axes are to be used for 3D function surface (carpet) plots. Choices are: XY,XZ,YZ,YX,ZX,ZY,TOP,SIDE,FRONT. TOP,SIDE,FRONT are equivalent to XY,XZ,YZ, respectively.

VPOINT

Format: VPOINT [x,y,z]
or VPOINT/ANGLES [phi,theta,radius]

Enter a viewpoint for a 3D plot.

Qualifiers

/XYZ (D)
/ANGLES
Controls whether the viewpoint will be interpreted as an (x,y,z) point in space, or as phi,theta,radius values, corresponding to DISSPLA conventions.

WALLS

Format: WALLS

Specifies which parts of the grids should be drawn for all plots, indicating the grid "geometry" or "configuration" (hence the name WALLS).

Qualifiers

/GRID=grid number
Specifies which grid these walls are to be associated with (default is grid 1).
/ADD
These walls are to be added on to the current list, rather than starting from wall number 1.
/ATTRIBUTES (D)
/NOATTRIBUTES
Controls whether attributes such as color, line type and thickness, and surface transparency are prompted for.

Programmed STOP