VIS-5D VERSION 3.3

1. OVERVIEW OF VIS-5D

VIS-5D is a software system for visualizing data made by numerical weather models and similar sources. VIS-5D works on data in the form of a five-dimensional rectangle. That is, the data are real numbers at each point of a "grid" or "lattice" which spans three space dimensions, one time dimension and a dimension for enumerating multiple physical variables. Of course, VIS-5D works perfectly well on data sets with only one variable or one time step (i.e. no time dynamics). However, your data should have some depth in all three spatial dimensions.

The VIS-5D system includes the vis5d visualization program, several programs for managing and analyzing five-dimensional data grids, and instructions and sample source code for converting your data into its file format. We have included the VIS-5D source code so you can modify it or write new programs. We have also included sample data sets from the LAMPS model and from Bob Schlesinger's thunderstorm model, so you can work through our examples.

VIS-5D version 1.0 was written by Bill Hibbard and Dave Santek of the University of Wisconsin Space Science and Engineering Center, supported by the NASA Marshall Space Flight Center, and by Marie-Francoise Voidrot-Martinez of the French Meteorology Office. Later version enhancements were written by Bill Hibbard, Brian Paul, and Andre Battaiola. Dave Kamins and Jeff Vroom of Stellar Computer, Inc. provided substantial help and advice in using the Stellar software libraries. Simon Baas and Hans de Jong of the Netherlands ported VIS-5D to HP workstations. Pratish Shah of Kubota Inc. ported VIS-5D to the Kubota Alpha/Denali workstation.

VIS-5D is offered under the terms of the GNU General Public License, which you can find in the file "NOTICE". As the notice states, there is no warranty for the VIS-5D system, but we would be interested in hearing about your questions and problems. Also, if you would like to be added to the VIS-5D mailing list, send email to:

Bill Hibbard (email: whibbard@macc.wisc.edu) or
Brian Paul (email: bpaul@macc.wisc.edu) at:

Space Science and Engineering Center
University of Wisconsin - Madison
1225 West Dayton Street
Madison, WI 53706

This document was written to make it easier for you to use VIS-5D. The next section tells you how to install VIS-5D on your system. Section 3 gives you a sample data conversion program and tells you how to modify it to convert your data into the VIS-5D file structures. Section 4 describes how to prepare your data for visualizing using the VIS-5D analysis programs. Section 5 describes how to visualize your data using the vis5d program. Section 6 leads you through using VIS-5D on the two sample data sets included on the tape. Section 7 tells you something about writing your own analysis programs that fit into the VIS-5D structure. Section 8 is a history of the versions of VIS-5D. The last section has a few words about the relation of VIS-5D to our much larger McIDAS system.

2.1 SYSTEM REQUIREMENTS

VIS-5D currently works with the following systems:

1. Silicon Graphics workstations:
   32MB RAM
   24-bit color and Z-buffer (or 8-bit Indigo)
   IRIX version 4.0.1 or higher.
   Multiple processors are used when present.

2. IBM RS/6000 workstations*:
   32MB RAM
   Model 320H or higher
   AIX version 3 or later is suggested.
   8 or 24-bit color display
   A 3-D graphics accelerator such as the GTO is recommended but not required.

3. HP series 7000 or 9000 workstations*:
   32MB RAM
   8 or 24-bit color display
   HP-UX A.09.01 or later

4. Sun Sparc workstations*:
   32MB RAM
   8 or 24-bit color display
   SunOS Release 4.1.x or later

5. DECstation workstations*:
   32MB RAM
   8 or 24-bit color display
   ULTRIX V4.2 or later

6. Kubota / DEC Alpha 3000 workstations:
   32MB RAM
   Denali Graphics
   OSF/1 V1.3 or higher
   KWS V1.3.3 or higher
   NPGL Run-time license

7. Stardent GS-1000 or GS-2000 workstations:
   32MB RAM
   True-color display (i.e. 32 planes).
   Optional SpaceBall is supported.
   Stellix version 2.3 or later is suggested.

*Note: for these workstations, VIS-5D does all 3-D rendering using a modified version of the VOGL library using X Windows. Because of this, rendering is rather slow. We would like to encourage anyone with one of these workstations having 3-D graphics hardware to modify VIS-5D to utilize that hardware.

If you would like to port VIS-5D to a new graphics system or workstation read the "PORTING" file which gives more information. If you succeed, please inform us so that we may add your work to the distribution.

2.2 INSTALLING VIS-5D

VIS-5D is obtained via anonymous ftp. If you don't have Internet access, you can obtain VIS-5D on tape by sending us a blank tape and a note explaining what you need.

Here are the installation instructions:

1. Go to the directory in which you want VIS-5D installed:
   % **cd /usr/mydir**

   NOTE: The installation of VIS-5D will result in a new subdirectory named "vis5d-3.3/" being created in the current directory.

   NOTE: Be sure that you have write permission in this directory. If you do not, you should become superuser before proceeding. When finished installing VIS-5D be sure to set the file ownership and permissions accordingly.

2. Start ftp:
   % **ftp iris.ssec.wisc.edu**
   or
   % **ftp 144.92.108.66**

3. Login as anonymous and send your id as the password:
   Name: **anonymous**
   Password: **myname@mylocation**

4. Go to the pub/vis5d directory:
   ftp> **cd pub/vis5d**

5. Transfer files in binary mode:
   ftp> **binary**

6. Get the VIS-5D archive file:
   ftp> **get vis5d-3.3.tar.Z**

7. Get optional files:
   NOTE: The GR3D0001 and GR3D0002 files are sample McIDAS grid files. You will only need them if you want to work through the examples using igg3d, igu3d, and comp5d:

   ftp> **get GR3D0001** [optional]
   ftp> **get GR3D0002** [optional]

8. Exit ftp:
   ftp> **bye**

9. Uncompress and un-tar the archive file:
   % **uncompress vis5d-3.3.tar.Z**
   % **tar -xvf vis5d-3.3.tar**

10. Change to the newly created vis5d directory:
    % **cd vis5d-3.3**

11. Run make:

    % **make**

    Make will print a list of systems supported for VIS-5D. Look for yours on the list and type the appropriate make command. For example, suppose you have an IBM RS/6000 without 3-D graphics hardware. You should type:

    % **make ibm-x**

    VIS-5D and its utility programs will now be compiled. If you do not have C and/or FORTRAN compilers on your system, this step will fail with an error message such as "cc: Command not found." or "f77: Command not found." In this case you will have to get the appropriate archive of executable programs:

    a. Repeat steps 1 through 5 as above.
    b. Then get the archive of executable files for your system:
        ftp> **get vis5d.xxx.bin.tar.Z**
      where xxx is one of the system configuration options listed previously by 'make'.
    c. Exit ftp:
        ftp> **bye**
    d. Uncompress and un-tar the archive:
        % **uncompress vis5d.xxx.bin.tar.Z**
        % **tar -xvf vis5d.xx.bin.tar.Z**

12. Test VIS-5D:

    % **./vis5d COMPLAMP**

    NOTE: To quit, click on the "EXIT" widget button.

13. You may delete the .tar files if desired.

2.3 CUSTOMIZING

After installation and testing you may want to customize the vis5d program by editing the src/vis5d.h file:

1. The visualization program vis5d assumes your system has 32 megabytes of memory. Although you can override this when you invoke vis5d, it may be convenient to change the default if your system has more than 32MB. The default number of megabytes is defined by the value of MBS in the src/vis5d.h include file.

2. If you want to specify a different default topography or map file, you can edit src/vis5d.h and change the values for TOPOFILE and/or MAPFILE.

When finished changing the src/vis5d.h file you must re-make the system by repeating installation step 11 above.

2.4 ORGANIZATION

When you are finished installing VIS-5D you should have a directory named "vis5d" which contains the following:

1. The VIS-5D program: vis5d
2. Utility programs: comp5d, compinfo, gg3d, igg3d, igu3d, help, and listfonts [SGI only]
3. Sample data sets: COMPLAMP, COMPSCHL
4. Map outline files: OUTLHRES, OUTLSUPW, OUTLUSAL, and OUTLUSAM
5. Earth topography file: EARTH.TOPO
6. The VIS-5D documentation (this file): README
7. The GNU general public license file: NOTICE
8. Hints on porting VIS-5D: PORTING
9. The vis5d source code directory: src/
10. The user interface source directory: lui2/
11. The utilities source directory: util/
12. User analysis functions directory with examples: userfuncs/
13. The user contributed software directory: contrib/
14. The VOGL GL-like library: vogl/
15. Optionally, sample McIDAS grids: GR3D0001, GR3D0002

3. PUTTING YOUR DATA INTO VIS-5D FILES

A data set for VIS-5D has a real number at each point in a five-dimensional grid. These data are organized into a series of three-dimensional spatial grids whose dimensions are latitude, longitude and height (i.e. altitude). The three-dimensional grids are organized into short sequences to enumerate the values of multiple physical variables at a single time. The short sequences of physical variables are repeated into a longer sequence which steps through many time steps.

These data are stored in our 3D grid files. A 3D grid file is identified with an integer between 1 and 9999, and contains a sequence of 3D grids, which are also identified by integers. A 3D grid file contains a directory entry for each 3D grid, which describes the size and geographic location of the grid, and the date, time and name of physical variable of the data in the grid array. A five-dimensional data set consists of a sequence of 3D grids in a 3D grid file, all with the same size and geographic locations. The grid sequence repeats the same short sequence of physical variables stepping forward through time. For example, the grid sequence from a weather model could be:

GRID NUMBER	DATE	TIME	PHYSICAL VARIABLE NAME
1	88035	000000	U
2	88035	000000	V
3	88035	000000	W
4	88035	000000	T
5	88035	000000	P
6	88035	010000	U
7	88035	010000	V
8	88035	010000	W
9	88035	010000	T
10	88035	010000	P
11	88035	020000	U
12	88035	020000	V
13	88035	020000	W
14	88035	020000	T

```
    15   88035 020000    P
```

This data set consists of 3 time steps of 5 physical variables. The physical variables are the U, V and W components of the wind vector, the temperature T and the pressure P. The date is February 4, 1988 and the time steps are midnight, 1 AM and 2 AM. Dates are represented by five digit decimal integers YYDDD where YY are the last two digits of the year and DDD is the three digit day within the year, so February 4, 1988 is 88035. Times are represented by six digit decimal integers HHMMSS where HH are the hour, MM are the minute, and SS are the second, so 2 AM is 020000.

The following sample program creates a 3D grid file and fills its 3D grids with data for a five-dimensional data set. It is repeated in the file sample.F, with make file sample.m. The easiest way to read your data into a 3D grid file is to alter the sample.F program. The subroutines it calls are all in the libmain.a library, and their source is in the src subdirectory. Here is a listing of sample.F:

```
  1 C THE MAIN PROGRAM OF YOUR CONVERSION PROGRAM MUST
  2 C BE NAMED SUBROUTINE MAIN0
  3 C
  4       SUBROUTINE MAIN0
  5 C
  6 C THE NEXT TWO COMMENTS ARE PRINTED BY THE 'help sample'
COMMAND
  7 C ? SAMPLE program to convert data to 3D grid files
  8 C ? sample gridf#
  9 C
 10 C DIMENSIONS OF 3D GRID
 11 C NOTE NLATS AND NLONS MUST BOTH BE LESS THAN OR EQUAL TO 150
 12 C NLATS, NLONS AND NHGTS MUST ALL BE AT LEAST 2
 13       PARAMETER (NLATS=31,NLONS=51,NHGTS=16)
 14 C
 15 C NUMBER OF PHYSICAL VARIABLES AND NUMBER OF TIME STEPS
 16 C NOTE EITHER OR BOTH MAY BE EQUAL TO 1.  THAT IS, VIS-5D
DOES
 17 C NOT FORCE YOU TO HAVE MULTIPLE VARIABLES OR TIME DYNAMICS.
 18       PARAMETER (NVARS=5,NTIMES=100)
 19 C
 20 C ARRAY FOR 3D GRID DATA
 21       REAL*4 G(NLATS, NLONS, NHGTS)
 22 C ARRAYS FOR GRID FILE ID AND GRID DIRECTORY
 23       INTEGER ID(8), IDIR(64)
 24 C ARRAY FOR VARIABLE NAMES
 25       CHARACTER*4 CNAME(5)
 26 C
 27 C LATITUDE, LONGITUDE AND HEIGHT BOUNDS FOR SPATIAL GRID
 28       DATA XLATS/20.0/,XLATN/50.0/
 29       DATA XLONE/70.0/,XLONW/120.0/
 30       DATA XHGTB/0.0/,XHGTT/15.0/
 31 C
 32 C STARTING DATE IN YYDDD AND TIME IN HHMMSS
 33       DATA JDAY/88035/,JTIME/020000/
 34 C TIME STEP IN HHMMSS
 35       DATA JSTEP/000100/
 36 C
```

```
C NAMES OF THE FIVE PHYSICAL VARIABLES
      DATA CNAME/'U   ', 'V   ', 'W   ', 'T   ', 'P   '/
C INITIALIZE GRID DIRECTORY TO ZEROS
      DATA IDIR/64*0/
C
C READ GRID FILE NUMBER FROM COMMAND LINE.  IPP WILL
C CONVERT THE PARAMETER # 1 TO AN INTEGER, WITH A DEFAULT
C VALUE OF 0.
      IGRIDF=IPP(1,0)
C IF ILLEGAL GRID FILE NUMBER, PRINT ERROR MESSAGE AND RETURN
      IF(IGRIDF .LT. 1 .OR. IGRIDF .GT. 9999) THEN
        CALL EDEST('BAD GRID FILE NUMBER ',IGRIDF)
        CALL EDEST('MUST BE BETWEEN 1 AND 9999 ',0)
        RETURN
      ENDIF
C
C CALCULATE GRID INTERVALS
      XLATIN=(XLATN-XLATS)/(NLATS-1)
      XLONIN=(XLONW-XLONE)/(NLONS-1)
      XHGTIN=(XHGTT-XHGTB)/(NHGTS-1)
C
C DATE AND TIME FOR FIRST TIME STEP
C IDAYS CONVERTS YYDDD FORMAT TO DAYS SINCE JAN. 1, 1900
      IDAY=IDAYS(JDAY)
C ISECS CONVERTS HHMMSS FORMAT TO SECONDS SINCE MIDNIGHT
      ISEC=ISECS(JTIME)
C
C INITIALIZE GRID IDENTIFIER TEXT TO BLANKS
C NOTE LIT CONVERTS A CHARACTER*4 TO AN INTEGER*4
      DO 10 I=1,8
10    ID(I)=LIT('    ')
C
C SET UP DIRECTORY ENTRY
C
C DIMENSIONS OF GRID
      IDIR(1)=NLATS*NLONS*NHGTS
      IDIR(2)=NLATS
      IDIR(3)=NLONS
      IDIR(4)=NHGTS
C
C LATITUDES AND LONGITUDES IN DEGREES * 10000
      IDIR(22)=4
      IDIR(23)=NINT(XLATN*10000.)
      IDIR(24)=NINT(XLONW*10000.)
      IDIR(25)=NINT(XLATIN*10000.0)
      IDIR(26)=NINT(XLONIN*10000.0)
C
C HEIGHTS IN METERS
      IDIR(31)=1
      IDIR(32)=NINT(XHGTT*1000.)
      IDIR(33)=NINT(XHGTIN*1000.)
C
C CREATE THE GRID FILE
      CALL IGMK3D(IGRIDF, ID, NLATS*NLONS*NHGTS)
```

```
 91 C
 92 C LOOP FOR TIME STEPS
 93       DO 200 IT=1,NTIMES
 94 C
 95 C SET DATE AND TIME IN DIRECTORY ENTRY
 96 C IYYDDD CONVERTS DAYS SINCE JAN. 1, 1900 TO OUR YYDDD FORMAT
 97       IDIR(6)=IYYDDD(IDAY)
 98 C IHMS CONVERTS SECONDS SINCE MIDNIGHT TO OUR HHMMSS FORMAT
 99       IDIR(7)=IHMS(ISEC)
100 C
101 C LOOP FOR PHYSICAL VARIABLES
102       DO 190 IV=1,NVARS
103 C
104 C SET VARIABLE NAME IN DIRECTORY ENTRY
105       IDIR(9)=LIT(CNAME(IV))
106 C
107 C
************************************************************
108 C READ YOUR DATA FOR TIME STEP NUMBER IT AND VARIABLE NUMBER
IV
109 C INTO THE ARRAY G HERE.
110 C NOTE THAT G(1,1,1) IS THE NORTH WEST BOTTOM CORNER AND
111 C G(NLATS,NLONS,NHGTS) IS THE SOUTH EAST TOP CORNER.
112 C MARK A GRID POINT AS 'MISSING DATA' BY SETTING IT = 1.0E35
113 C
************************************************************
114 C
115 C CALCULATE 3D GRID NUMBER
116       IGRID=IV+NVARS*(IT-1)
117 C WRITE DATA IN G AND DIRECTORY IN IDIR TO 3D GRID
118 C NOTE WE PASS THE NEGATIVE OF THE GRID NUMBER (I.E. -IGRID)
119       CALL IGPT3D(IGRIDF,-
IGRID,G,NLATS,NLONS,NHGTS,IDIR,IGNO)
120 C
121 C END OF PHYSICAL VARIABLE LOOP
122 190   CONTINUE
123 C
124 C INCREMENT DATE AND TIME, CONVERT JSTEP FROM HHMMSS TO
SECONDS
125       ISEC=ISEC+ISECS(JSTEP)
126 C IF SECONDS CARRY PAST ONE DAY, ADJUST SECONDS AND DAYS
127       IDAY=IDAY+ISEC/(24*3600)
128       ISEC=MOD(ISEC,24*3600)
129 C
130 C END OF TIME STEP LOOP
131 200   CONTINUE
132 C
133       RETURN
134       END
```

The routines IGMK3D and IGPT3D are the interface to the 3D grid structures. The call to IGMK3D at line 90 creates a 3D grid file. Its parameters are:

1 INTEGER*4 - number of 3D grid file to create

2 array of 8 INTEGER*4 - a 32 byte text ID for the file
3 INTEGER*4 - maximum number of grid points in any 3D grid.

After the 3D grid file is created, IGPT3D is called in line 119 once for each combination of time step and physical variable to put 3D grids into the file. Its parameters are:

1 INTEGER*4 - number of 3D grid file to write to
2 INTEGER*4 - minus the number of the 3D grid to write. This is 0 or positive to indicate write to next empty grid.
3 array of REAL*4 - array of grid points to write
4 INTEGER*4 - first dimension of grid array, # of latitudes
5 INTEGER*4 - second dimension of grid array, # of longitudes
6 INTEGER*4 - third dimension of grid array, # of heights
7 array of 64 INTEGER*4 - directory for 3D grid
8 INTEGER*4 - number of 3D grid actually written, returned by IGPT3D.

VIS-5D allows data sets which span more than one 3D grid file. In this case the grid sequence of repeating variables and repeating time steps continues across grid file boundaries. A single 3D grid file is limited to 100,000,000 grid points (400 megabytes). If your data set contains more than this number of grid points, then you should alter sample.F to create a new 3D grid file (by incrementing IGRIDF and calling IGMK3D) on every Nth time step, where N time steps will fit in one 3D grid file. Note that the comp5d command described in section 4 references data sets as sequences of 3D grid files.

The VIS-5D system processes the gridded data based on the information in the grid directories, which is contained in the IDIR array in the sample.F program. It is a good idea to initialize IDIR to all zeros, as in line 40. The size of the 3D grid is set in entries 1 to 4 of IDIR (lines 72 to 75). Note the restrictions on data set size described in section 4 of this document.

The date and time of the 3D grid are set in entries 6 and 7 of IDIR, as in lines 97 and 99. Note that they are represented in our YYDDD and HHMMSS formats described above. Four functions are available in libmain.a for converting between these formats and a format which makes date and time calculations easy. The IDAYS function converts YYDDD format to days since January 1, 1900, as in line 60. The ISECS function converts HHMMSS format to seconds since midnight, as in lines 62 and 125. This makes it easy to do calculations with dates and times, as in lines 125, 127 and 128. Then the IYYDDD function converts days back to YYDDD and the IHMS function converts back to HHMMSS, as in lines 97 amd 99.

The physical variable name is 4 ASCII characters packed into entry 9 of IDIR, as in line 105. The LIT function in libmain.a converts a CHARACTER*4 to an INTEGER*4.

The spatial location of the grid is described in terms of latitude and longitude in ten-thousandths of a degree, and in terms of height (altitude) in meters. The grid element G(1,1,1) is in the north west bottom corner of the grid, and the grid element G(NLATS,NLONS,NHGTS) is in the south east top corner. The grid latitude and longitude are described in entries 21 to 25 of IDIR, as in lines 78 to 82. The grid heights are described in entries 31 to 33, as in lines 85 to 87. The NINT function is a FORTRAN intrinsic for converting a REAL to the nearest INTEGER. The latitude, longitude and height spacings are simply the distances between between successive grid points. Latitudes are positive in the northern hemisphere, longitudes are positive in the western hemispere, and of course heights are positive above sea level.

The real work in modifying the sample.F program is writing code for getting your data into the G array, in lines 107 to 113. For some data you may want to fake the latitude, longitude and height coordinates. However, if your data is geographical and large scale, then you may want

to describe its location accurately, and it may be necessary to resample your data to a regularly spaced grid in latitude, longitude and height from some other map projection. It may also be necessary to transpose your data array to get the index order to be LAT, LON and HGT, and to invert your data array in some index to make sure G(1,1,1) is the north west bottom corner. Even in faked coordinates, you may need to transpose or invert your data array to get the right 'handedness' in the display. The VIS-5D system allows grid points marked as missing, indicated by array values greater than 1.0E30. If you do fake the latitude, longitude and height coordinates, then the topography and map display of the vis5d program will be meaningless. If you calculate trajectories for your data set, either use accurate coordinates, or take great care to get relative time, distance and velocity scales consistent in the faked coordinates. Otherwaise trajectory paths will not be realistic.

The IPP function in libmain.a returns the value of a command parameter as INTEGER*4, as in line 45. There are similar functions CPP and DPP in libmain.a which return CHARACTER*12 (converted to upper case) and REAL*8 values for command parameters. They get command parameters based on their sequential position in the command line. They all have similar function parameters:

1 INTEGER*4 - sequence number of command parameter
2 (IPP) INTEGER*4 - default value of command parameter
or
2 (CPP) CHARACTER*12 - default value of command parameter
or
2 (DPP) REAL*8 - default value of command parameter.

There is also a mechanism for picking up command parameters based on keywords. This is done with the functions IKWP, CKWP and DKWP in libmain.a. They get command parameters based on position after a keyword of the form '-keyword'. IKWP returns an INTEGER*4, CKWP returns a CHARACTER*12 (converted to upper case) and DKWP returns a REAL*8. They all have similar function parameters:

1 CHARACTER*12 - keyword string in command line
2 INTEGER*4 - sequence number of command parameter after keyword
3 (IKWP) INTEGER*4 - default value of command parameter
or
3 (CKWP) CHARACTER*12 - default value of command parameter
or
3 (DKWP) REAL*8 - default value of command parameter.

The NKWP function in libmain.a returns the number of sequential parameters after a keyword. Its function parameter is:

1 CHARACTER*12 - keyword string in command line.

On the most machines the REAL*4 format is not a subset of the REAL*8 format, so make sure to declare DPP and DKWP as REAL*8, as well as their third function parameters (for default values of command parameters).

If you would rather write your grid conversion program in C instead of FORTRAN, look at the file 'sample.c'. It contains examples of how to easily read and write grid files using C structures and routines in stdio.

To make your own topography files for use with the -topo option there is a maketopo.c program in the vis5d/src/ directory. It explains the format of the file and shows how to create such files.

4. MANAGING AND ANALYZING YOUR DATA

Once your data set is in a 3D grid file, you can list directory information about the grids using the command:

```
igg3d list I J -gr3df N
```

where N is the 3D grid file number, and I and J give the range of grid numbers to list. You can get a quick idea of the data values using the command:

```
igg3d info I J -gr3df N
```

which will list the minimum and maximum values, the mean, the standard deviation and the number of grid points marked for missing data, for grid numbers I to J in 3D grid file number N.

Before you can visualize your data set, you must convert it to a compressed file structure using the command:

```
comp5d N M F
```

where N is the first 3D grid file number and M is the number of grid files in the data set. The M parameter allows data sets which span multiple grid files and should not be confused with the total number of 3D grids in the data set. F is the name of the compressed grid file. You can choose whatever name you want, but note that comp5d will convert the name to all upper case characters. If your data set contains wind vector components you can use the -wind keyword to select a subset of wind components or calculate horizontal wind speed, named 'SPD ', for the compressed file. The longitude, latitude, and vertical components of the wind vector must be named 'U ', 'V ' and 'W ' respectively. If you use the -wind keyword, then only those wind-relevant variables (i.e. U, V, W & SPD) whose names are listed after -wind will be included in the compressed file. For example, to include SPD and W in the compressed file, from a 3D grid file containing U, V and W components, use the command:

```
comp5d N M F -wind SPD W
```

The 'compinfo' program is used to get information about a compressed grid file made by comp5d. For example:

```
compinfo COMPLAMP
```

will print information about the COMPLAMP file including the grid dimensions, number of time steps, number of variables, etc.

There are restrictions on the dimensions of data sets which can be visualized using the vis5d program. Currently, you are limited to a maximum of 30 physical variables and 400 times steps. The vis5d program will also fail if there is a trivial spatial dimension:

NLATS < 2
NLONS < 2

NHGTS < 2

The vis5d program will perform badly, possibly making errors, if the total 5-D size:

NLATS * NLONS * NHGTS * NTIMES * NVARS

is too large. The limit depends on the amount of memory in your system. For a 64MB system, the limit is around 25,000,000, with performance degrading as the data set size exceedes the limit.

VIS-5D provides the gg3d and igg3d programs which can be used to reduce the resolution and scale of a data set to meet these limits. The gg3d program resamples a 3D grid to new array dimensions and new extents in latitude, longitude and height, using the command:

```
gg3d samp N I M J
gg3d ave  N I M J
```

where N and I are the numbers of the source 3D grid file and grid, and M and J are the numbers of the destination 3D grid file and grid. The 'samp' version calculates destination grid point values by linearly interpolating between source grid point values, and is appropriate for increasing resolution. The 'ave' version calculates destination grid points by averaging multiple source grid point values, and is appropriate for decreasing resolution. Without any keywords gg3d will do a straight copy operation. Invoke the gg3d command with the keyword:

-size NLATS NLONS NHGTS

to set the grid dimensions for the destination grid as different from the dimensions for the source grid. Invoke gg3d with the keywords:

-lat XLATS XLATN
-lon XLONE XLONW
-hgt XHGTB XHGTT

to set extents (range bounds) for the latitude, longitude and height for the destination grid as different from the extents for the source grid. The -lat, -lon and -hgt keywords take real arguments.

The igg3d program provides options for copying and deleting 3D grids and for interpolating between 3D grids in time. Sequences of 3D grids are copied using the command:

```
igg3d get N I J M K
```

where N is the source 3D grid file number, I and J are the range of source grid numbers, M is the destination grid file number, and K is the starting destination grid number. A single grid may be copied within a 3D grid file using the command:

```
igg3d copy I J -gr3df N
```

where N is the 3D grid file number, I is the number of the source grid and J is the number of the destination grid. A range of grids may be deleted with the command:

```
igg3d del I J -gr3df N
```

where N is the 3D grid file number and grid numbers between I and J are to be deleted.

The igg3d command provides two different options for time interpolation. The first is:

```
igg3d ave K I J D T -gr3df N
```

where grid number K is produced by interpolating between grid numbers I and J, all in 3D grid file number N. Grid number K will be assigned day D (in YYDDD format) and time T (in HHMMSS format). The relative weighting of grids I and J is calculated from this date and time, assuming linear time interpolation. If grid K is not between grids I and J in date and time, igg3d prints an error message. The igg3d command also provides a more complex time interpolation option:

```
igg3d int I D T -setdel S M -lag U V -gr3df N
```

This will put a grid in the next empty slot of 3D grid file number N, assigned to day D (in YYDDD format) and time T (in HHMMSS format). This grid will be interpolated from a sequence of grids, all in file number N, at grid numbers I, I+S, I+2S, ... , I+(M-1)S. This sequence of grids should be ascending in date and time. igg3d will search the sequence and linearly interpolate between the two consectutive grids from the sequence which bracket day D and time T. Furthermore, the interpolation will be done in a coordinate system moving at constant velocity (U, V), where U and V are in meters per second, with V positive for motion from south to north and U positive for motion from west to east. The two bracketing grids from the sequence will be shifted in latitude and longitude to their positions at day D and time T, and the result interpolated between these two spatially shifted grids. Furthermore, if the grids in the sequence are identified in their directory entries with variable name 'U ' or 'V ', then the corresponding component of the velocity (U, V) will be subtracted from the grid values.

The 'int' option of igg3d may seem complex, but it is just what you need if you want to write a script to re-interpolate a five-dimensional data set to a new sequence of time steps. It is particularly useful if the source sequence does not have uniform time steps, or if the physics are moving through the spatial grid and you want to avoid blurring in the time re-interpolation. You would set M equal to the number of time steps and S equal to the number of physical variables in the source five-dimensional data set. The I parameter would be set equal to the grid number in the first time step of the variable being interpolated. Note that this igg3d option will put the new grid at the end of the grid file containing the source data set, but you can use 'igg3d get' to move it to another grid.

You can use the command:

```
igu3d make N M
```

to create 3D grid file number N, which allows 3D grids of up to M points each. The names of 3D grid files have the form:

GR3Dnnnn

where nnnn is the four digit decimal grid file number, padded with leading zeros if needed to make four digits. Use the UNIX command:

```
rm GR3Dnnnn
```

to delete a 3D grid file.

The help command will list a quick reference to the parameter formats for the VIS-5D utility commands. Just enter:

```
help NAME
```

where NAME is the name of the command such as igg3d, igu3d, or comp5d.

5.1 USING VIS-5D TO VISUALIZE YOUR DATA

This section describes how to use the VIS-5D visualization program, vis5d. It is almost completely controlled using the mouse with a graphical user interface. The best way to learn to use it is to experiment. There is no way to harm your data within from the program.

After you have used comp5d to produce a compressed file, you can interactively visualize your data set with the command:

```
vis5d file [options]
```

where 'file' is the file name of the compressed file (made with comp5d) which you want to visualize. A number of options are available (none are usually needed):

-alpha
: [SGI only] Use alpha blending instead of "screen door" transparency.

-box x y z
: This lets you specify the aspect ratio or proportions of the 3-D box. Default values are 2 2 1.

-font xfont
: [Stardent] Sets the font used in the 3-D window. Use the xlsfonts command to see a list of X fonts on your system.
Example: vis5d COMPLAMP -font fg-30

-font glfont height
: [SGI] Sets the font used in the 3-D window. Use the listfonts command included with VIS-5D to see a list of GL fonts on your system. The height argument is the font height in points where 72 points = 1 inch.
Example: vis5d COMPLAMP -font Helvetica 30
[IBM] Not supported.

-full
: Open the 3-D window as a borderless, full-screen size window.

-map file
: Use a map file other than the default of OUTLSUPW. See section 2.3 to setup a different default.
Example: vis5d COMPLAMP -map OUTLUSAL

-mbs n
: Override the assumed system memory size of 64 megabytes. See section 2.3 to setup a different default value.

-rate ms
: Change the default animation rate. ms is the minimum delay in milliseconds between frames. Default is 100 ms.

-texture rgbfile
: Specify an SGI .rgb file to texture map over the topography. [SGI only]

-topo file
: Use a topography file other than the default of EARTH.TOPO. See section 2.3 to setup a different default.

-wdpy xdisplay
: Put the widgets on a different X display. Useful in combination with -full for making slides and videos.

Example: vis5d COMPLAMP -full -wdpy pluto:0

-wide w

Set width of line segments in pixels (default is 1.0). Again, useful for making videos.

Example: vis5d COMPLAMP -wide 3.0

-wind2 uvar vvar [wvar]

Specify the names of a secondary set of U, V, and (optionally) W wind component variables to use when drawing the H-WIND 2 and V-WIND 2 vector slices. Useful when you have two sets of wind vector components that you want to visualize simultaneously.

Example: vis5d MYDATA -wind2 U2 V2 W2

If you type 'vis5d' without arguments you will get a list of available options.

When you start vis5d it loads your data set and then opens a 3-D window on the right and a control panel on the left of the screen. The 3-D window is used to view and interact with the data. In its upper-left corner is a combination analog/digital clock which indicates the current time step. The control panel contains several groups of buttons.

The first button group contains the following buttons:

```
[ANIMATE]   [STEP]      NEW VAR     EXIT
NEW SURF    TOP         SOUTH       WEST
[TOPO]      [MAP]       BOX         CLOCK
SAVE        RESTORE     GRID #'s    CONT #'s
[PRETTY]    REVERSE     [SAVE PIC]  [PERSPEC]
```

These buttons are used to toggle, activate, or control various functions as described below. Some of the above buttons are enclosed in brackets to indicates that they may be blank upon starting vis5d. This will happen when the button does not apply to the current data set, because the button would conflict with a command line option, or because the feature is not available on your hardware.

The next group of radio buttons are used to determine how the mouse behaves in the 3-D window:

Normal - Normal mouse mode is used to rotate, zoom, and pan the graphics in the 3-D window. See section 5.3.

Trajectory - This mode is used for creating and displaying wind trajectories. See section 5.7.

Slice - This mode is used to reposition horizontal and vertical slices. See section 5.5.

Label - This mode is used to create and edit text labels in the 3-D window. See section 5.8.

Probe - Used to inspect individual grid values by moving a 3-D cursor through the 3-D grid. See section 5.9.

These modes are mutually exclusive; only one may be selected at a time. To the immediate right of these buttons is the mouse button legend. It is there to remind you of the use of each mouse button in the 3-D window for the currently selected mode.

Next, if your data set contains U, V, and W wind component variables there will be a row of four wind slice buttons:

```
H-WIND 1     V-WIND 1     H-WIND 2     V-WIND 2
```

A wind slice depicts the motions of the wind by drawing small arrows which point in the direction of the wind. The length of each line segment indicates it's magnitude. The tails of the line segments are all anchored within a horizontal or vertical plane through the 3-D box. The location of the slice plane can be changed with the mouse while in "Slice" mode. See section 5.5 for more details.

The remaining set of widget buttons is used to control the display of 3-D isosurfaces, horizontal contour slices, vertical contour slices, horizontal colored slices, vertical colored slices, and volume renderings of your data. The buttons are arranged in a matrix. Each row corresponds to a physical variable in your data set. Each column corresponds to one type of graphic listed above.

The display of any graphic is controlled by clicking on its widget button with the left mouse button. Each type of graphic also has a small control window which appears when turned on. The control windows are different for each type of graphic and are explained below. To bring up a graphic's control window without toggling its display, use the middle mouse button. When the graphic is displayed it will be the same color as the widget button, making it easy to distinguish and identify different variables in the display. To change the color of the graphic, click on its widget button with the right mouse button and a small window with four slider widgets will appear. By changing the levels of red, green, and blue you can make any color.

If the control panel window becomes obscured by other windows, you can bring it to the top by pressing the "F1" key while the mouse pointer is in the 3-D window. This is especially useful when using the '-full' option.

5.2 VIS-5D CONTROL BUTTONS

The following is a description of each button in this group. Some of them will be described in more detail later in this document.

ANIMATE — This toggle button turns animation on or off. Use the left or middle mouse buttons for forward animation and the right mouse button for reverse animation. Does not appear when viewing data sets with one time step.

STEP — This button has three possible uses depending on which mouse button is pressed:

- Left Button - Step ahead one time step
- Middle Button - Go to first time step.
- Right Button - Backward one time step.

This button does not appear when viewing data sets with one time step.

NEW VAR — Used to duplicate physical variables or invoke external analysis functions. This is explained further in section 5.10.

EXIT — Exit the program. A verification window will appear to ask you to verify your decision.

NEW SURF — Computes a new 3-D contour surface.

TOP — Depending on which mouse button is pressed:
Left or Middle: Reset the 3-D window to the default top-view.
Right: Set the 3-D window to a bottom-view.

SOUTH — Depending on which mouse button is pressed:
Left or Middle: Set 3-D window to a south-view.
Right: Set 3-D window to a north-view.

WEST — Depending on which mouse button is pressed.
Left or Middle: Set 3-D window to a west-view.
Right: Set 3-D window to an east-view.

TOPO — Toggle the display of topography. This button will not appear if the topography file was not found. Click on TOPO with the right mouse button to edit the topography color.

MAP — Toggle the display of map lines. This button will not appear if the map file was not found.

BOX — Toggle the display of the 3-D box.

CLOCK — Toggle the display of the clock.

SAVE — Save current graphics and colors. After you've setup a variety of isosurfaces, slice, wind trajectories and colors it is useful to be able to save them and restore them the next time the data set is visualized. Click on SAVE with left mouse button to save all graphics and colors. Click on SAVE with the right mouse button to just save all colors.

RESTORE — Restore the data last saved with the SAVE button.

GRID #s — Normally the bounds of the data set in latitude, longitude and kilometers are displayed along the edges of the box. Use this button to display the numbers in grid coordinates instead.

CONT #s — The numbers which are drawn on contour line slices can be toggled on or off with this button.

PRETTY — Toggle the 'pretty' rendering option on or off. When turned on the 3-D graphics will be rendered using whatever high quality options are available on your system:
Stellar: isosurfaces are antialiased and rendered with improved transparency.
SGI: antialiasing is done on systems with an accumulation buffer.
Others: Pretty rendering is not supported on other systems
When PRETTY is turned on, your mouse pointer will turn into a "busy" symbol. Pretty rendering is slow and may take over minute to render a single image. It should normally be turned off. This is used when making slides or printouts.

REVERSE — Normally, the 3-D box and clock are drawn in white on a black background. This option reverses that and draws a black box and clock on a white background. This is used with the xwd(1) program when making print outs.

SAVE PIC — Used to save the image in the 3-D window to a file.

PERSPEC — Toggle between perspective and orthogonal viewing projections.

5.3 NORMAL VIEWING MODE

In 'Normal' mouse mode the mouse is used to view the data in the 3-D window. By pressing the left mouse button and moving the mouse while the cursor is in the 3-D window, the 3-D image can be rotated. At any instant you can only control two of the three degrees of freedom of box rotations. However, by releasing and re-pressing the left mouse button you can change your "grip" on the box. With practice you will learn to control the box through a series of mouse moves, releasing and re-pressing the left button between moves. The center button controls two very different things depending on how the mouse is moved. Holding the center button down and sliding the mouse away from yourself zooms in, making the box get bigger. Sliding the mouse towards yourself zooms out and makes the box get smaller. Holding the center button down and sliding the mouse right moves a plane of invisibility (i.e. a clipping plane) into the box, creating a cut away view of the box contents. Sliding the mouse left brings the clipping plane toward yourself, eventually out of the box altogether. The right mouse button is pressed to translate the box in the window. This is useful if you want to zoom in to something that is not in the center of the box. Note that the center of rotation for box rotations stays at the center of the screen rather than in the center of the box.

5.4 3-D CONTOUR SURFACES

A 3-D contour surface (isosurface) shows the 3-D volume bounded by a particular isovalue. The isosurface has the specified iso-level, the volume inside contains values greater (or less) than the iso-level. The volume outside contains values less (or greater) than the iso-level.

The display of 3-D isosurfaces is controlled with the first column of buttons in the button matrix in the control panel. Clicking on one of these buttons with the left mouse button causes a small window with a slider widget to appear below. It is used to select iso-level values for contour surfaces. The slider window displays the variable name and the range of values for that variable. The slider value is initially set to the minimum of the range. You can select an iso-level by placing the cursor on the slider, holding any mouse button down, and moving the cursor right or left. When you have the iso-level value you want, click on the 'NEW SURF' widget button above. This will cause iso-level contour surfaces to be generated for the selected physical variable for each time step.

Toggling ANIMATE on will let you watch the time dynamics of the iso-level contour surfaces. Note that the surfaces are generated asynchronously with the animation, so you may not see the surfaces for all the time steps as the clock hand makes it revolution. The new surfaces will appear on successive clock revolutions. If you use a slider and the NEW SURF widget to change the value of iso-level contour surfaces, the animation may show a combination of new and old valued surfaces while the system calculates the new surfaces.

Clicking on an isosurface widget button with the middle mouse button will summon a variable's iso-level slider without turning the surface on or off.

To change the color of an isosurface, click on the appropriate isosurface widget with the right mouse button. A window will appear with four sliders labeled red, green, blue, and transparency. The color of the isosurface is a mixture of red, green and blue components. By changing the mixture with the sliders, any color can be made. The transparency value slides between 0.0 for invisible and 1.0 for opaque. On SGI and IBM systems, "screen-door" transparency is used by default and only four levels of transparency are available. On the Stellar and SGI VGX, VGXT, VTX, or RE using the -alpha option, alpha blending is used. Neither method is perfect and may produce unexpected artifacts under certain conditions.

5.5 CONTOUR, COLOR, AND WIND SLICES

Slices allow you to look at planar cross sections of data in the 3-D box. These slices can be oriented either horizontally or vertically and may depict either contour lines, colored slices, or wind vectors.

As described in section 5.1, the last group of buttons on the control panel form a matrix of buttons, the four right-most columns of which control slices. There is a column of buttons for horizontal contour slices, vertical contour slices, horizontal colored slices and vertical colored slices, respectively.

If your data set contains U, V, and W variables, there will also be a row of wind vector slice buttons as described in 5.2. There are two buttons for horizontal wind slices and two buttons for vertical wind slices.

To activate/turn on a slice, click on the appropriate widget button with the left mouse button. The initial position for slices is the middle of the box. The exact slice location in terms of latitude, longitude or elevation is given by a small numeric labels near the one corner of each slice. To print the numbers as grid coordinates instead of geographic coordinates, toggle the "GRID #s" widget button on the control panel.

The position of slices can be changed interactively using the mouse. To do so you must first be in SLICE mode by selecting the SLICE radio button. To move any slice, simply point at the slice's corner with the mouse, press the right mouse button and drag it to a new position. Vertical slices may also be moved in a perpendicular motion by "grabbing" the middle of the top or bottom edge and dragging it. A slice may be moved while in animation mode, however, some jumpiness may occur because new slices are computed asynchronously.

When a slice is turned on, a small control window will appear as well. The function of the control window depends on the type of slice:

Contour slice: the control window contains a type-in widget used to specify the interval between contour lines. To enter a new value, first point at the value with the mouse. The field will blank to white. You can now type in a new value and press RETURN. If you don't press RETURN the old value will be restored when you move the mouse pointer away. The BACKSPACE key can be used to correct typing mistakes. Decreasing the interval will cause denser contour lines to be generated, increasing the interval will result in sparser lines.

If you enter a negative interval then all contour lines with a negative value will be drawn with dashed lines while positive values will be drawn with solid lines.

Optionally, after the interval value you may specify a range of values (a,b) which will cause only contour values between a and b to be drawn. For example, suppose you enter "-10 (-30,20)" (without quotes). This will result in contour lines for values between -30 and 20 at intervals of 10 with negative lines drawn as dashed lines.

Color slice: the control window contains a 3-function graph which maps data values to colors. To change the red, green, or blue function press the left, middle, or right mouse button, respectively, and drag the mouse to draw a new function. By default, low data values are mapped to blue and high data values are mapped to red. If you need to restore the default mapping, press 'r' while the mouse pointer is in the window.

Instead of using the mouse to draw a new function you can use the keyboard cursor (arrow) keys to modify the shape and position of the default function curves.

Press the left/right keys to move the curves left or right. Press the up/down keys to change the shape of the curves.

Wind vector slice: the control window contains two type-in widgets to control the scale and density of wind vectors. The scale parameter is used to multiply the length of vectors drawn. If you want to double the length of all vectors, enter 2.0. If you want to halve the lengths, enter 0.5. The density parameter controls how many wind vectors are displayed. This value can only be between zero and one. To make one-half the number of vectors, enter 0.5, for one-fourth enter 0.25, etc. The default values for both parameters is 1.0.

As with 3-D surfaces, the middle mouse button can be used to toggle the control window without changing the slice's on/off status.

To match a slice to its respective widget button, look at the colors:

Contour line and wind vector slices: the color of the widget button is the same as the slice color. To change the color, select the widget button using the right mouse button. A window with four sliders: red, green, blue, and transparency will appear. By changing the amount of red, green, and blue you can make any color. The transparency slider has no effect.

Colored slices: the color of the widget button is the same as the slice's position label. To change the color, click on the appropriate widget button using the right mouse button. Use the red, green, and blue sliders as previously described.

5.6 VOLUME RENDERING

If you are running VIS-5D on an SGI workstation with a VGX, VGXT, VTX, RE or RE2 graphics system or a Kubota/Denali system you can make volumetric renderings of the variables in your dataset. In this case there will be a sixth column of buttons will be present in the main button matrix. To get a volume rendering of any variable just click on that variable's button in the

sixth column using the left mouse button. Note that only one volume rendering can be viewed at a time.

You can change the mapping of data values to colors and opacity with the color window which appears when the volume is first displayed. As with colored slices (described above), you can change the mapping by "drawing" them with the mouse. Use the left mouse button to modify red, middle mouse button to modify green and the right button to modify blue. The fourth curve, which is drawn in white, controls opacity. Hold down any of the <shift>, <control> or <alt> keys with any mouse button to change the opacity curve. High values are opaque and low values are transparent. Also, you can use the cursor keys to modify the curves. Holding down any of the <shift>, <control> or <alt> keys causes the cursor keys to modify the opacity curve.

The volume rendering is made as follows:

1. Examine the current viewing transformation to determine which axis of the 3-D box is most nearly parallel to the view direction.
2. Create a number of colored slices perpindicular to that axis which map data values to colors and opacity.
3. Render the colored slices in back to front order. The alpha values at vertices are interpolated and blended by the graphics hardware to make smooth transitions between and within slices.

Note that the volume rendering is completely recomputed whenever a different variable is selected, animation is turned on, or when the 3-D box is rotated to a new orientation.

Despite the simplicity of the algorithm, most fields are rendered acceptably. Those that aren't can be improved by adjusting the color and opacity mappings. While more attractive volume rendering techniques are known, none are fast enough to support interactive visualization as needed in VIS-5D.

5.7 WIND TRAJECTORIES

If your data set contains U, V, and W wind component variables, you can create wind trajectories. Wind trajectories trace the motion of air through the 3-D volume much line smoke trails in a wind tunnel. To enter trajectory mode select the TRAJECTORY radio button on the control panel. A window titled "Interactive Wind Trajectories" will appear near the bottom of the screen and a 3-D cursor will appear inside the 3-D view box. This 3-D cursor is used to specify where a new wind trajectory should be made. The STEP button on the main control panel is also important because it is used to select the time step at which to create the trajectory.

Wind trajectories are dealt with in sets. Currently, eight sets are available. Each set is represented in the trajectory window with a button labeled Set1, Set2, ..., Set8. Each set can be individually displayed, colored, or deleted. As you create new trajectories you may want to group them in sets corresponding to location, time, etc.

The first step in creating a trajectory is to select a position with the 3-D cursor. Use the right mouse button to drag the 3-D cursor around inside the 3-D box. The 3-D cursor will move in 2-D in a plane parallel to the plane of projection. That is, the cursor will stay at a constant distance of depth. By alternately rotating the view box with the left mouse button and placing the cursor with the right mouse button, the 3-D cursor can be placed anywhere inside the view box. The TOP, SOUTH, and WEST buttons as explained in section 5.2 can also be useful when making trajectories.

Second you should select a time step with the STEP button on the control panel. When the trajectory is made, it will be traced forward from the current time step to the last time step and will be traced backward through time to the first time step.

Finally, to make a trajectory at the current cursor location and current time step, press the middle mouse button when pointing inside the 3-D window. The trajectory will appear as a line segment. By turning on the ANIMATE button, you can observe how the trajectory travels through time and space. Typically, you will repeat the process of positioning the 3-D cursor and clicking the middle mouse button to create a set of trajectories.

Interesting results can be seen by making a trajectory when the ANIMATE button is turned on: a trajectory will be created for every time step instead of just one. This will show you the path of every air parcel which passes through a single point in space.

Here is a summary of the various trajectory functions:

1. To position the 3-D cursor, use a combination of rotating the view box with the left mouse button and dragging the 3-D cursor with the right mouse button.

2. Use the STEP button or ANIMATE option to select a time step.

3. Press the middle mouse button to create a trajectory at the current cursor location and time step.

4. To toggle the display of a trajectory set on or off, click on the set button with the left mouse button.

5. Select the current trajectory set by clicking on the set button with the middle mouse button.

6. Change the color of a trajectory set by clicking on the corresponding trajectory set button with the right mouse button. A color selection window will appear on the left of your screen. By adjusting the red, green, and blue sliders, you can change the trajectory set's color.

7. A trajectory set may be deleted with the 'Delete Set' button in the trajectories window. You will asked to verify your decision.

8. You can delete the last trajectory made by clicking on the 'Delete Last' button in the trajectories window.

The ability to make individual sets of trajectories can used to group them according to position or time criteria.

Wind trajectories can be depicted in two ways: as line segments or as ribbons. You can select ribbons by clicking on the RIBBON button in the trajectory window. Toggling the RIBBON button will not effect trajectories you have already made but rather controls how new trajectories will be displayed.

The trajectory window also contains two type-in widgets labeled STEP and LENGTH. The STEP value is used to control the step size used in the trajectory tracing algorithm. 100 is the default. The LENGTH value is used to control the length of trajectories. It is simply a scaling value which multiplies the length of the trajectory. For example, to double the length of new trajectories enter a value of 2, to triple the length enter 3.

5.8 TEXT LABELS

Text labels are used to annotate the image in the 3-D viewing window. Typically this is used as the final step before presenting the output. You could add a title, your name, the date, highlight a particular feature of the data, or document the meaning of the data seen in the window.

To enter text labeling mode select the LABEL radio button on the control panel.

To create a text label position the mouse pointer somewhere in the 3-D window and press the left mouse button. A vertical bar cursor will appear at that location and you can now type in the text. The <Backspace> key can be used to correct errors. When you are finished, press <Return>.

To move a text label to a new position, point at it with the mouse, hold down the middle mouse button and drag the mouse. As you move the mouse an outline of the text will be dragged with the pointer until you release the mouse button.

To delete a text label, pointing at it with the mouse and pressing the right mouse button. Be careful, you will not be asked for verification before deleting a label. Once it's deleted you can only restore it by retyping it.

The SAVE button on the control panel will save any text labels you have made.

Use the '-font' option to select a different font.

5.9 DATA PROBE

Sometimes it's useful to be able to inspect individual data values at various locations in the 3-D volume. You can do this with the data probe. Click on the PROBE mode button on the control panel. A 3-D cursor appears in the 3-D box which you can move around using the right mouse button. For each physical variable the value for the current time step is printed along the left edge of the 3-D window.

5.10 MAKING NEW VARIABLES

The NEW VAR button on the control panel is used to add new physical variables in vis5d. There are two kinds of new variables you can add:

1. Cloned variables: these are copies of existing variables. You can use a cloned variable to make two different isosurfaces of the same variable simultaneously, for example.

2. External function variables: you can invoke an external function (which you write) to compute a new variable as a function of the existing ones.

When you click on the NEW VAR button a window appears which lists the variables which you can clone and external functions which you can invoke. After you select one, a new row of buttons will be added to the control panel for the new variable. You can use then make isosurfaces, contour slices, etc. of the that variable.

5.10.1 CLONED VARIABLES

Suppose you want to clone the U wind component variable so that you can make both +20 and -20 isosurfaces of it. First, click on NEW VAR and then select U from the pop-up window. The cloned variable will be named U'. You can then treat U' as any other variable and make an isosurface of it.

5.10.2 EXTERNAL ANALYSIS FUNCTIONS

An external analysis function is a function written by you in FORTRAN which is called by VIS-5D to produce a new variable as a function of the existing variables. As an example, there is included a function SPD3D which computes wind velocity as: SPD3D = SQRT(U*U + V*V + W*W). Be aware that the external function feature is intended for experienced VIS-5D users who are also proficient FORTRAN programmers.

All external functions must be placed in a directory named "userfuncs". This is relative to the current directory when you run vis5d. For example, suppose you always run vis5d while in "/usr/jones/data", then your analysis functions must be in "/usr/jones/data/userfuncs". Also, this directory contains a script "externf" which is used to compile your function.

To write an external function it's best to copy one of the supplied examples and then modify it. The included "userfuncs/example.f" is fully commented for this purpose. Later, when you call your function from within vis5d, the function will be invoked once for each time step. The arguments passed to the function include:

1. the number of physical variables in the data set
2. the name of each variable
3. the size of the 3-D grid
4. the 3-D grids of data for each variable
5. the date and time of the time step

Your function will have to scan the list of variables to find the ones it needs for the computation. Then it must do the actual computation, generating a new grid of data. The examples we've included demonstrate how to do this.

Suppose you want your function to be named "delta". Then the name of the FORTRAN program must be "delta.f". You would compile the function by typing "externf delta". If there are no errors, an executable file "delta" will be written. Then in vis5d when you select NEW VAR, "delta" should appear in the list of functions in the pop-up window.

There are two places for vis5d to get the grid data which it passes to your external function: from the original, uncompressed McIDAS file or the compressed VIS-5D file. The uncompressed McIDAS data is better because it has more precision. In order for vis5d to find the McIDAS file, the compressed file must have been made with the version 3.1 comp5d and the McIDAS file must be in the current directory. If the McIDAS file can't be found, then the compressed data which vis5d has in memory will be passed to your external function. Note that this has no bearing whatsoever on the construction of your external function.

5.11 KEYBOARD FUNCTIONS

The following keyboard functions can be invoked while the mouse pointer is inside the 3-D viewing window:

Key	Function
F1	Raise or lower the control panel widget window. This is useful with the -full option.
F2	Toggle display of system information including memory used and number of graphics to be computed.
P	Print the current window image. A PostScript printer must be available. Set the PRINTER environment variable from your shell to specify which printer to use.

If you want to program your own keyboard functions look the in the file src/gui.c for the func1(), func2(), func3(), etc functions. They are called when the corresponding function key is pressed.

5.12 SPACEBALL SUPPORT

Version 2.1 and later of VIS-5D for the Stardent also supports the SpaceBall made by Spatial Systems, Inc. If you have a SpaceBall make sure the environment variable 'SPACEBALL' is defined. This can be done by adding a line similar to 'setenv SPACEBALL /dev/tty04' (make sure you use the correct tty number) to your .cshrc file. If vis5d recognizes the SpaceBall it will emit two short beeps upon startup. See your SpaceBall documentation for more information about setup.

Twisting the SpaceBall causes an appropriate rotation of the 3-D box. Pressing the SpaceBall up, down, left, or right is analogous to translation with the mouse. Pressing the SpaceBall toward or away you causes the box to zoom in or out, respectively. The SpaceBall function keys are defined as follows:

(1) Top view	(2) South View	(3) West View	(4) Unused
(5) Bottom View	(6) North View	(7) East View	(8) Rezero Ball

These assignments can be changed in the src/sb.c source file.

5.13 COMMON QUESTIONS (AND ANSWERS)

Q: Why doesn't the map and/or topography work for me?

A: If the "MAP" and "TOPO" buttons don't appear on the control panel, the default files were not found. By default, vis5d looks in the current directory for the OUTLSUPW and EARTH.TOPO iles. Change the src/vis5d.h file to specify absolute path names or copy the map and topography files to your current directory.

Q: How can I print the image in the window?

A: Just press P as mentioned in section 5.11. Also, if available, you should first turn on the "PRETTY" button to get a good quality image and select the "REVERSE" button. If you do not have a color printer you should make all your VIS-5D colors shades of gray to get a good idea of what results you can expect. Alternatively, you can save the window image to a file and then manipulate it with other programs before printing.

Q: How do I make photographic slides?

A: A 35mm camera on a tripod works good. First turn off all lights in the room and close all window shades to prevent glare. Setup your image and turn on "PRETTY" if available. Take several pictures at various exposures to find the best setting for your camera.

Q: How do I make videos?

A: First you'll need a scan converter to convert the RGB video signal sent to your monitor into an NTSC (or PAL) signal which can be recorded by any standard VCR. There is no scripting language in VIS-5D so you must compose your video "on the fly" and edit it later if you have two VCRs. You can even make video titles with vis5d! Start vis5d with '-full' and a large font with '-font', go into 'Label' mode, hide the control panel window, and then make your titles.

Q: Does VIS-5D support non-uniform grid height spacing and/or other map projections?

A. Wait for the next version of VIS-5D...

5.14 FINAL NOTES

The SGI and Stardent versions of VIS-5D are parallel programs; after you select a new isosurface, move a slice, etc. you are free to do other things such as rotate the box. The other versions does not allow this; user input is blocked while graphics are being computed. [A way to set up multiple threads of execution within a process is required to solve this problem.] For a large data set, generating isosurfaces or slices for all the time steps may take a while.

VIS-5D uses a bounded memory management system to avoid using more memory than what is available, thereby preventing virtual memory thrashing which leads to poor performance. When the memory limit is reached, the least recently displayed graphics are deallocated. You can monitor the amount of memory used by pressing the F2 key. If you try to read in a grid file which will not fit into memory, or not leave enough working memory free, you will get an error message saying so. A system with 32MB of RAM can read with files of up to approximately 12.5 million data points, with a 128MB system you can visualize 50 million data points, etc.

When you save the current graphics and/or colors with the SAVE button, the information is written to a file with the extension ".SAVE". For example, if you do a SAVE while visualizing the "COMPLAMP" data set, the save file will be named "COMPLAMP.SAVE". This file can be rather large. When it's no longer needed you can delete it.

The vis5d user interface may be complex to describe in words, but we have tried hard to make it simple in reality. After a little practice using the sample data sets we hope it feels natural.

Since version 3.2 of VIS-5D there is a user-contributed software directory: contrib/. See the README file in that directory for a description of current contributions.

6. VISUALIZING AND ANALYZING THE SAMPLE DATA SETS

We have included two sample data sets. To visualize one of Bob Schlesinger's thunderstorm simulations, enter the command:

```
vis5d COMPSCHL
```

Then to view an isosurface of QL (moisture content):

1. Click on the QL button in the left column of the button matrix.
2. On the slider, select a value near 1.0, then click on the NEWSURF button.
3. Turn on animation with the ANIMATE button.

To view a vertical contour line slice of QL:

1. Turn off animation by clicking on ANIMATE again.
2. Click on the QL button in the third column.
3. Move the slice by first selecting the SLICE radio button. Then use the right mouse button to drag any corner of the slice along the edges of the 3-D box.

To visualize a LAMPS (Limited Area Meso-Scale Prediction System) model simulation of an extratropical cyclone, enter the command:

```
vis5d COMPLAMP
```

To view an isosurface of wind speed over a topography with map lines:

1. Click on the TOPO and MAP buttons.
2. Click on the SPD button in the first column. Then select a value near 45.0 on the slider and click on NEWSURF.
3. Turn on ANIMATE and you will see an animation of the 45 m/s wind isosurface.

To make some interactive wind trajectories:

1. Turn off the wind speed isosurface by clicking on the SPD button again
2. Select the TRAJECTORY button.
3. Move the mouse pointer into the 3-D window and press the middle mouse button. You will get a series of white wind trajectory lines passing through the 3-D cursor location.
4. Move the 3-D cursor by dragging it with the right mouse button then click the middle button to make more trajectories.
5. Select RIBBON and then the SET 2 button and try making some yellow ribbon trajectories.

These data sets are also available in the form of 3D McIDAS grid files name GR3D0001 and GR3D0002. See section 2 if you don't have these files. To list the thunderstorm grids and to see statistics about them, enter the commands:

```
igg3d list 1 190 -gr3df 1
igg3d info 1 190 -gr3df 1
```

The COMPSCHL compress file was made from the GR3D0001 file with the command:

```
comp5d 1 1 COMPSCHL -wind spd u v w
```

To try out the compinfo program, type:

```
compinfo COMPSCHL
```

Then, as before, the data can be visualized with:

```
vis5d COMPSCHL
```

To list the LAMPS grids and to see statistics about them, enter the commands:

```
igg3d list 1 189 -gr3df 2
igg3d info 1 189 -gr3df 2
```

The COMPLAMP compressed grid file was produced with the command:

```
comp5d 2 1 COMPLAMP -wind spd u v w
```

7. WRITING YOUR OWN DATA ANALYSIS PROGRAM

The source files igg3d.F and gg3d.F, in the src subdirectory, are both good models for analysis programs. They use the IGGT3D routine to read arrays from 3D grids, and the IGPT3D routine to write arrays to 3D grids. Section 3 of this document describes many of the routines in libmain.a which may be useful for writing analysis programs.

8. VERSION HISTORY

This is a version history of VIS-5D.

1.0 (December 1988)

This was the original version of VIS-5D for the Stellar GS-1000. It was used to give demonstrations at the ECMWF in December 1988 and at the AMS conference in Anahiem in January 1989. It had the following features:

- Depict time series of multivariate 3-D grids by animated isosurfaces and horizontal contour line slices.
- World topography map with map boundaries.
- Wind trajectory tracing with the traj5d program.

2.0 (Fall 1991)

This version was only available for the Stellar GS-1000/2000 and introduced the following features:

- Faster isosurface generation.
- Horizontal and vertical slices moved interactively with the mouse.
- Colored slices.
- Interactive wind trajectory creation.
- Ribbon trajectories.
- Label / text annotations.
- "Pretty" rendering option.

The format of the compressed grid file was changed slightly with version 2.0. Specifically, the trajectory files of version 1.0 were eliminated, trajectories are now stored in the compressed grid file itself. Also, the internal storage representation for surfaces and slices has been changed.

2.1 (February 1992)

This is the first version of VIS-5D available for the SGI and IBM workstations. It was also modified to use less memory during isosurface generation.

2.2 (April 1992)

This version of VIS-5D runs on the base SGI Indigo with 8-bit color though some features not available. It also has the following improvements:

- The -box option for changing the proportions of the 3-D box (SGI and Stardent only).
- User topography files. VIS-5D now uses the EARTH.TOPO file instead of TOPOHRES to make the map. The maketopo.c program shows how to make new .TOPO files. (SGI and Stardent only)

3.0 (August 1992)

This version features the following improvements:

- Horizontal and vertical wind vector slices.
- Improved SAVE and RESTORE functionality.
- New trajectory widget options.
- Separate map and topography controls.
- CLONE option added.
- Simultaneous colored and contour line slices.
- Improved transparency, PRETTY option on SGI.
- Same source code for SGI, Stardent, and IBM.
- Improved portability and porting guide added.
- New video and hardcopy convenience features.

3.1 (July 1993)

New features:

- User-written analysis functions.
- SAVE PIC button to save window image to a file.
- Perspective viewing mode.
- New contour line options to draw dashed negative lines and restrict contouring to a specific range of values.
- Data Probe mode.
- Topography color editing.
- Grid compression done layer-by-layer.

3.2 (August 1993)

New features and changes:

- Volumetric rendering on SGI systems with VGX, VGXT, VTX, RE, or RE2 graphics hardware.
- User-contributed software directory.
- 2-D contour function rewritten in C.

3.3 (January 1994)

New features:

- VIS-5D ported to HP, DEC, Sun, and Kubota (DEC Alpha) workstations. The most important part of this work was the enhancement and integration of the VOGL library. This work was done by Simon Baas and Hans de Jong

for the Dutch Meteorological Institute, KNMI. Porting to the Kubota Denali graphics system was done by Pratish Shah of Kubota Inc. Thanks guys!

-wdpy option now creates a window on the widget display which can be used to move and interact with the 3-D view using the widget display's mouse.

SAVEPIC button let's you save the window image in PostScript or color PostScript formats (SGI only).

-wind2 option added to specify a second set of U,V,W variables for the second set of wind vector slices.

-texture option added for a texture mapping an image onto the topography (SGI only).

user functions are computed faster on SGI multi-processor systems by computing time steps in parallel.

9. VIS-5D AS A SUBSYSTEM OF McIDAS

VIS-5D is being distributed as a stand alone system free of charge. However, it is also a subsystem of a much larger system named McIDAS (Man-computer Interactive Data Access System). McIDAS is a proprietary system of the Space Science and Engineering Center which we have been developing for over twenty years and which now includes about two million lines of FORTRAN. McIDAS also includes lots of SSEC designed hardware for satellite data acquisition, communications, and workstations. McIDAS can ingest data from almost all weather satellites and many other sources in real time, manage enormous data bases (our GOES archive contains 100 trillion bytes), display that data, and apply lots of source specific analyses to the data. As a subsystem of McIDAS, VIS-5D needs to be consistent with its structures. This may make VIS-5D seem a little odd as a stand alone system. McIDAS is evolving to be sure, but a system so large changes slowly. As recently as 1980 the source code for McIDAS resided on several tons of punch cards, and it has come a long way since then.