Annual Report

Visualization Techniques to Aid in the Analysis of Multi-Spectral Astrophysical Data Sets[1]

Gitta Domik[2], Salim Alam and Paul Pinkney
University of Colorado
15 November 1992

1. Introduction

This report describes our project activities between September 1991 and October 1992. Our activities included stabilizing the software system STAR, porting STAR to IDL/widgets (improved user interface), targeting new visualization techniques for multi-dimensional data visualization (emphasizing 3-d visualization), and exploring leading-edge 3-d interface devices.

During the past project year we emphasized high-end visualization techniques, by exploring new tools offered by state-of-the-art visualization software (such as AVS[3] and IDL[4]/widgets), by experimenting with tools still under research at the Department of Computer Science (e.g. use of glyphs for multidimensional data visualization), and by researching current 3-d input/output devices as they could be used to explore 3-d astrophysical data (as explained in section 2.1). As always, any project activity is driven by the need to interpret astrophysical data more effectivly.

1. This work is supported by the National Aeronautics and Space Administration under a grant for Astrophysics Software and Research Aids, NAGW-1902. Investigators: E.W. Brugel, G.O. Domik and T.R. Ayres.

2. Address: Gitta Domik, Dept. of Computer Science, University of Colorado, CB 430, Boulder, CO. 80309-0430. Tel. (303)492-4062. Email: domik@cs.colorado.edu

3. Application Visualization System by AVS, Inc.

4. Interactive Data Language by Research Systems, Inc.

2. High-End Visualization Techniques for Astrophysical Data

Prof. John Bally joined CASA[5] in January of 1992. Collaboration with John Bally gave us an opportunity to present new visualization techniques as recently developed (or still under development) by the computer graphics community. His research involves scientific interpretation of data cubes as described in section 2.1; he is familiar with the representation of spatial-spectral data (data containing spatial as well as spectral dimensions) and was interested in the development of tools to interact with his data. In order to present the scientist with expressive and effective visual representations, we studied the nature of his data as well as his scientific interpretation intents.

The role of data visualization is to stimulate mental processes different from quantitative data analysis. Visual data analysis offers an overview of data characteristics through browsing, often leading to an intuitive understanding of data characteristics and their relationships by sacrificing accuracy in interpreting the data values. Because the human visual system emphasizes spatial relationships, up to three data characteristics can be represented in a natural, intuitive way in form of spatial dimensions. Data visualization is an indirect way of interpreting data: instead of being interpreted from their natural, usually quantitative characteristics, data are first encoded into a pictorial representation. The encoding process bears the danger of creating artifacts and therefore missing the correct interpretation: e.g. abrupt color changes may mislead by pointing to discontinuities in a data set or subjective assessments of patterns may lead to misinterpretations.

A visual representation of data values should take into account the data characteristics as well as the interpretation intent, as suggested by (Mackinlay, 1986; Wehrend and Lewis, 1990; Robertson, 1990). De Ferrari (1991) adds the influence of other visualization specifications, such as user imposed restrictions, besides the interpretation intent.

In the case of visualizing the astrophysical data cubes, we used a simple mapping model as shown in Figure 1 to map numbers into pictures and map the pictures into a valid scientific interpretation (Domik, 1991).

5. Center for Astrophysics and Space Astronomy at the University of Colorado at Boulder

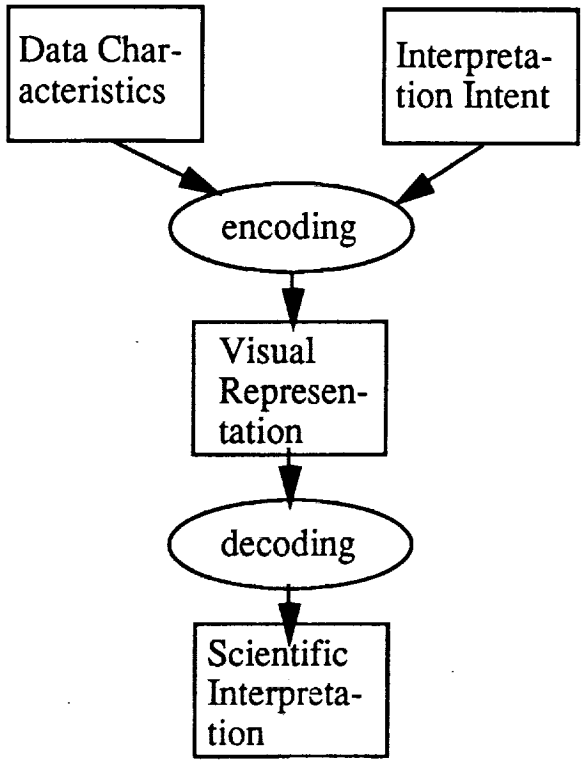


Figure 1: Going from numbers to pictures to a valid scientific interpretation of the numbers.

2.1 Data Characteristics

Data is collected by a 7 m telescope dish owned by ATT Bell Labs in New Jersey. It operates at a frequency between 20 to 40 Ghz, corresponding to a wavelength of 1.3 cm to .7 cm. The collected data is in form of 2-d image tiles for each measured frequency. Processing of the collected raw data values from the heterodyne receiver results in even gridded data values defined in three dimensions (spatial, spatial, frequency). The data values correspond to a count of carbon monoxide molecules at that specific spatial location and frequency. Data values range between -32000 and +32000.

2.2 Interpretation Intent

Carbon monoxide is used to trace molecular clouds. It is important to understand changes of the molecular cloud in space as well as in frequency. Therefore, scientific tasks in interpreting data values are: observe, if the cloud is expanding; look for indications that the cloud is collapsing; in what direction is it moving; identify dense matter at each stage of frequency.

2.3 Encoding numbers into pictures

It is important to express essential data characteristics in the resulting visual representations. In the case of the astrophysical data cubes, such essential characteristics are spatial location as well as frequency and the data value itself. Leaving both spatial dimensions in their natural form and mapping frequency into a third spatial dimension created an even gridded cube with the data values expressed as voxels. This geometric representation will be referred to as "data cube" throughout this paper.

However, the various slices of spatial data values could also collapse into one single slice, where spatial dimensions are represented in their natural form, but various data values along one frequency dimension appear clustered together. This geometric representation will be called "collapsed slices".

It is also important to represent the data is an effective way, so that the decoding process from pictures to scientific interpretation is quick and accurate. The following visual representations were chosen and discussed with John Bally:

a) Iso surfaces:

Data values of a certain threshold were connected to create iso surfaces. This is a well known rendering technique of the data cube representation. In this representation, the overall shape of the data can be observed as well as isolated volumes (see Figure 2). Understanding the overall distribution of the carbon monoxide in the given spatial-spectral dimensions is important in order to understand the detailed quantitative information. The iso surface representation can be enhanced by adding individual slices through the data cube (see Figure 3) or by using several transparent iso surfaces (see Figure 4).

b) Translucent representation:

Rays penetrate the data cube from a chosen point-of-view and accumulate values

of opacity assigned to the data values. This representation inflicts a translucent characteristics on the molecular clouds, very much like the visual form of real clouds. It allows to look into the cloud as opposed to observe the surface only. Because the scientist felt a natural understanding of this representation, it was favored as compared to any other representation.

Figure 5 shows a translucent rendering of the cube by looking at the data from the side: one spatial dimension increases to the right, the frequency increases from bottom up. The rapid changes of the data values in the mid-frequencies show special characteristics of carbon monoxide at these frequencies. Figure 6 shows the same data cube using the same representation looking from top down onto the cube.

c) Data slicer:

To monitor the change of one data value in relation to its neighbor values, a data slicer was used. Even though a data slicer can only monitor the neighbors surrounding a certain data value inside a plane, flexibility in placing the slices inside the cube can monitor various changes. Figure 7 shows four slices cutting through the cube parallel to the x/y plane, enhancing the understanding of the movement of the cloud through frequency. Figure 8 shows three orthogonal slices through the data, intersecting in the center of the cube

d) Collapsed data slices:

To collapse all (or a subset of) data slices along the frequency dimension into one single two dimensional image, one must be careful to maintain all information of all three dimensions. By using principles of Gestalt theory (Gordon, 1989), all data values along one frequency dimension are mapped into one complex "glyph" or "icon". In order to be seen as belonging to the same Gestalt, the set of data values along one frequency dimension is mapped into one figure that can be distinguished from its neighbors. The difference of one figure from its neighbor relates to the spectral characteristics of carbon monoxide and can be interpreted accordingly.

The resulting image can also be seen as one entity, therefore allowing interpretation of the overall distribution and change of carbon monoxide in the data cube. Visual representations of collapsed data slices leave it up to the human visual system to decide if the focus is on large-scale or small-scale structures.

Figure 9 shows a representation using color to indicate the various carbon monoxide counts of nine consecutive slices of a subset of the original data values; the spatial location inside each red square is used to indicate the various spectral responses. Figure 10 encodes five slices into five characteristics of a cube: width, height, depth, color and view point.

Figure 2: Isosurface of data

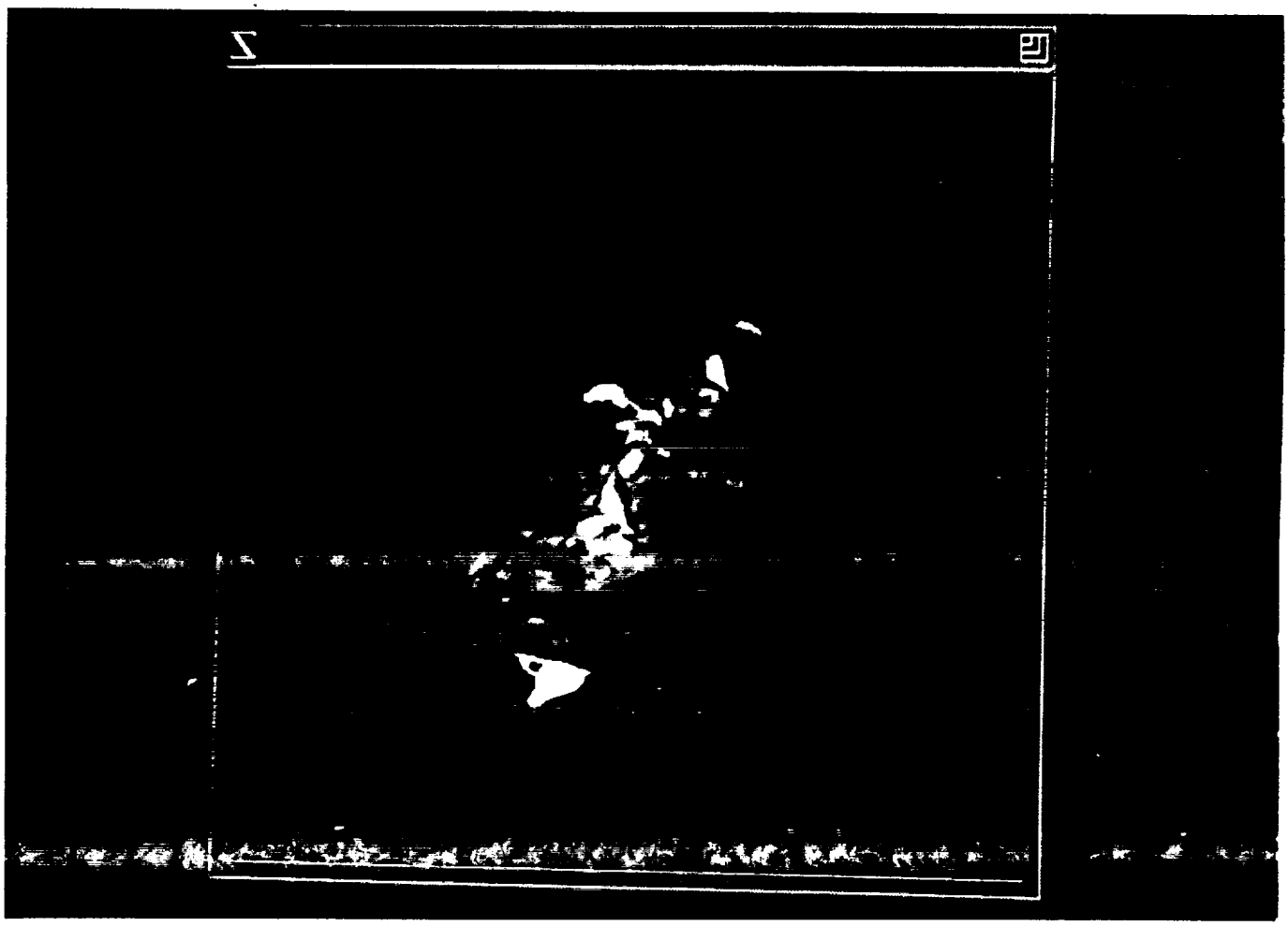


Figure 3: Isosurface with plane through data cube

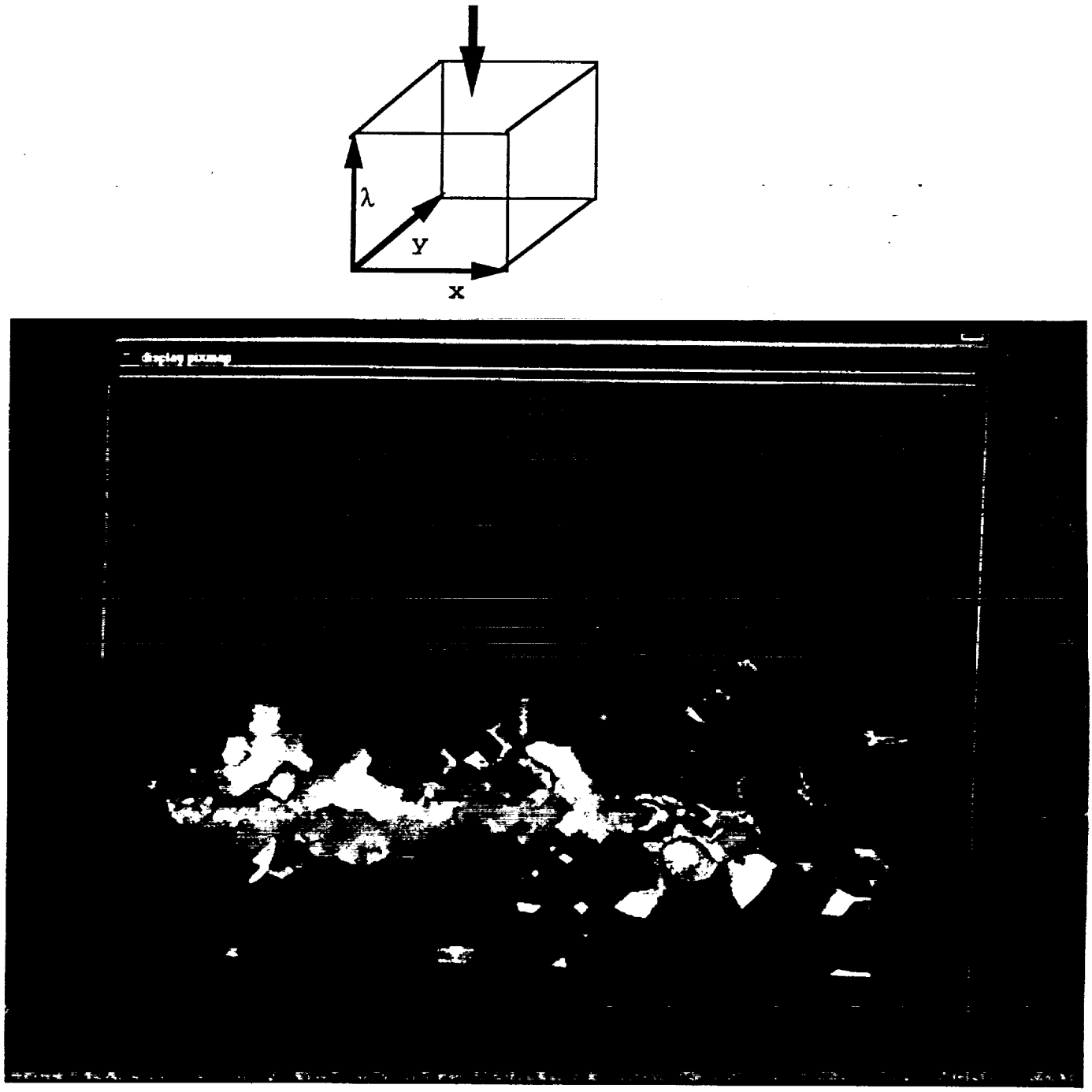
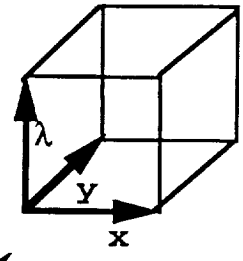


Figure 4: Two transparen isosurfaces of the same data

Figure 5: Translucency



viewing from side

dark blue: low molecule count
cyan: medium molecule count
green: high molecule count
yellow: highest molecule count

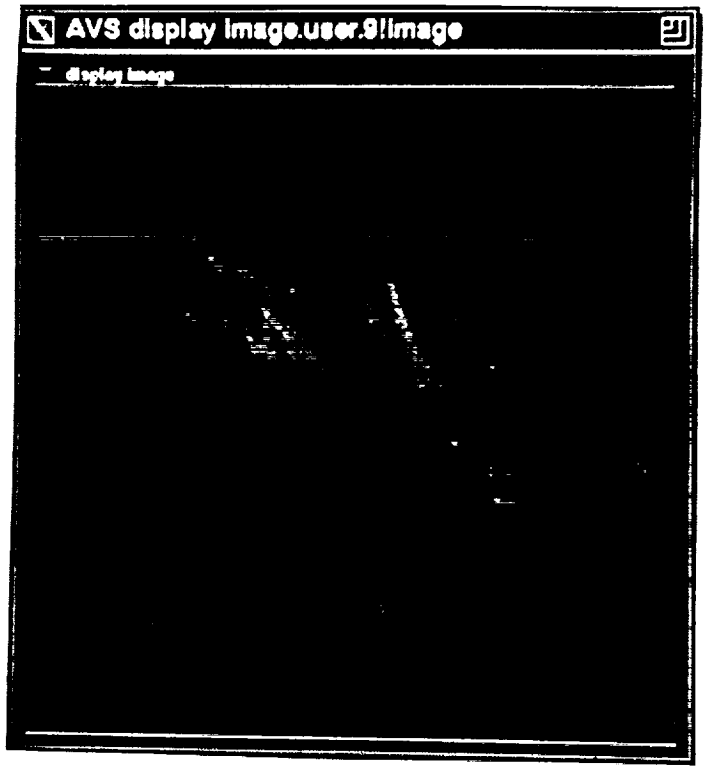
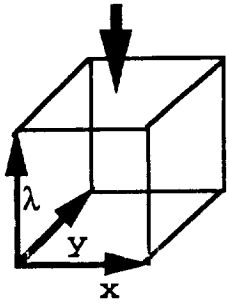
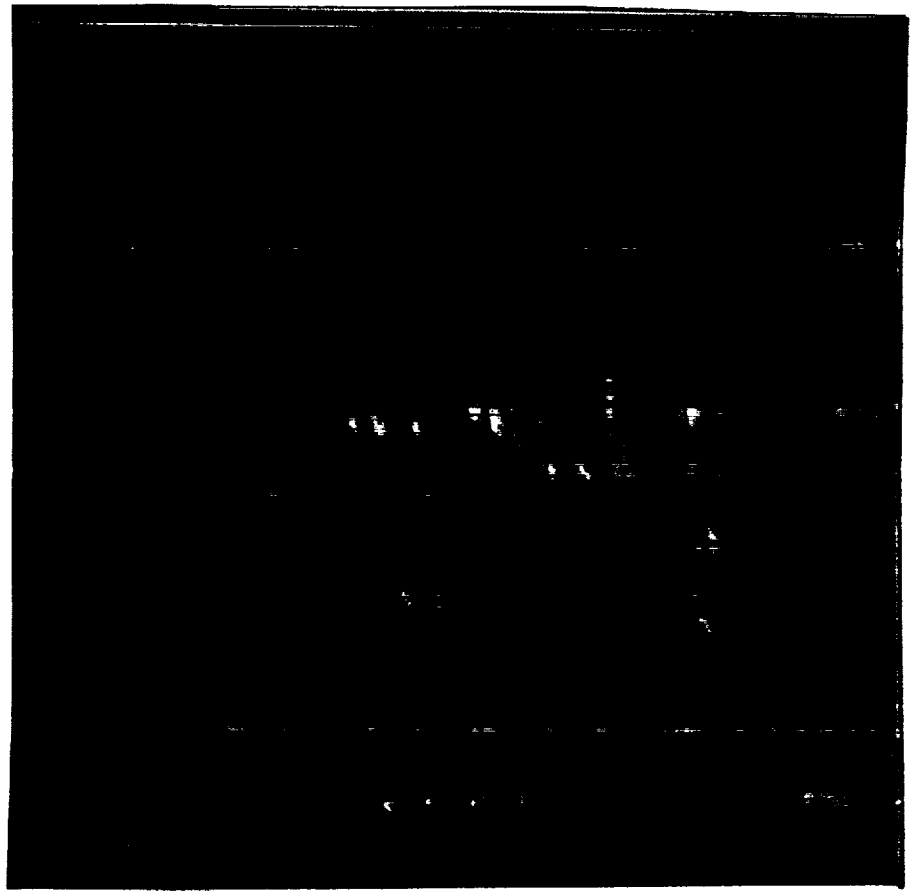


Figure 6: Translucency
viewing from top



dark blue: low molecule count
cyan: medium molecule count
green: high molecule count
yellow: highest molecule count



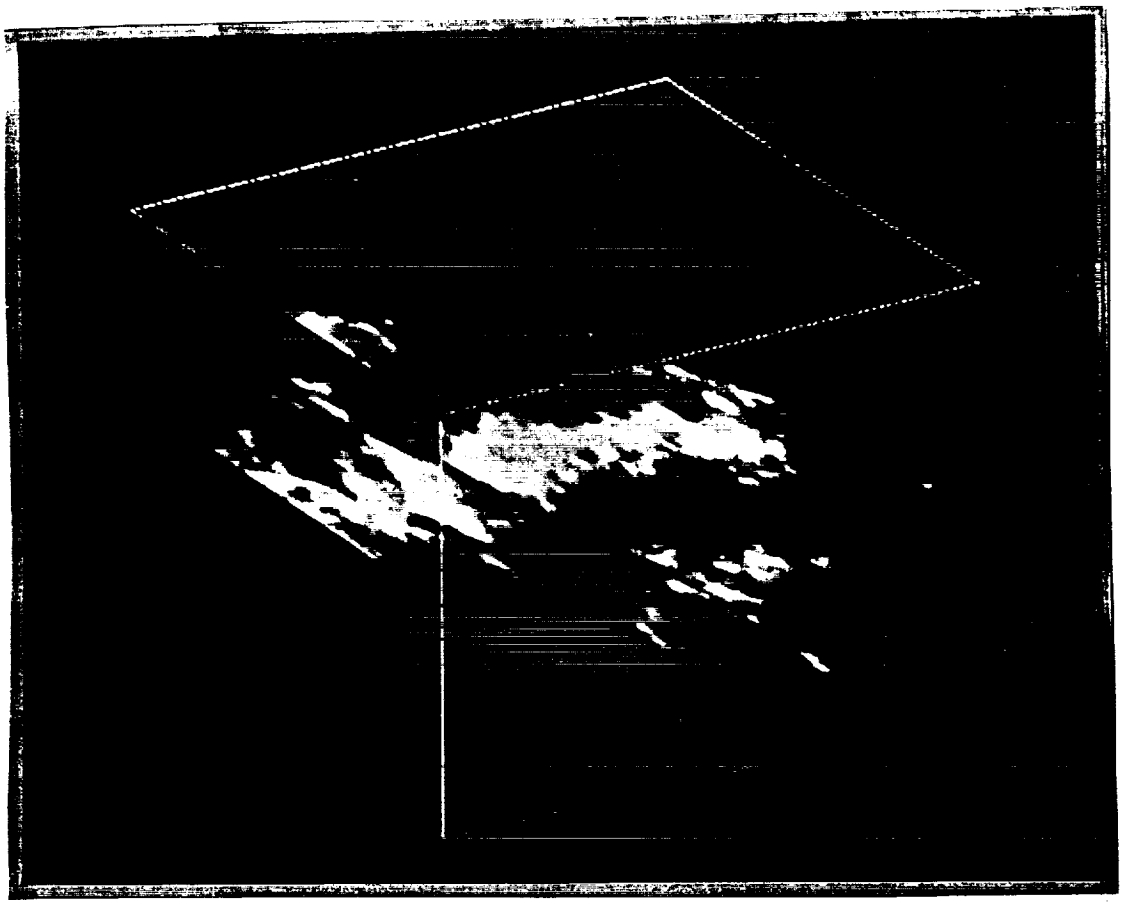


Figure 7: Data slicer (viewing four slices parallel to x/y plane)

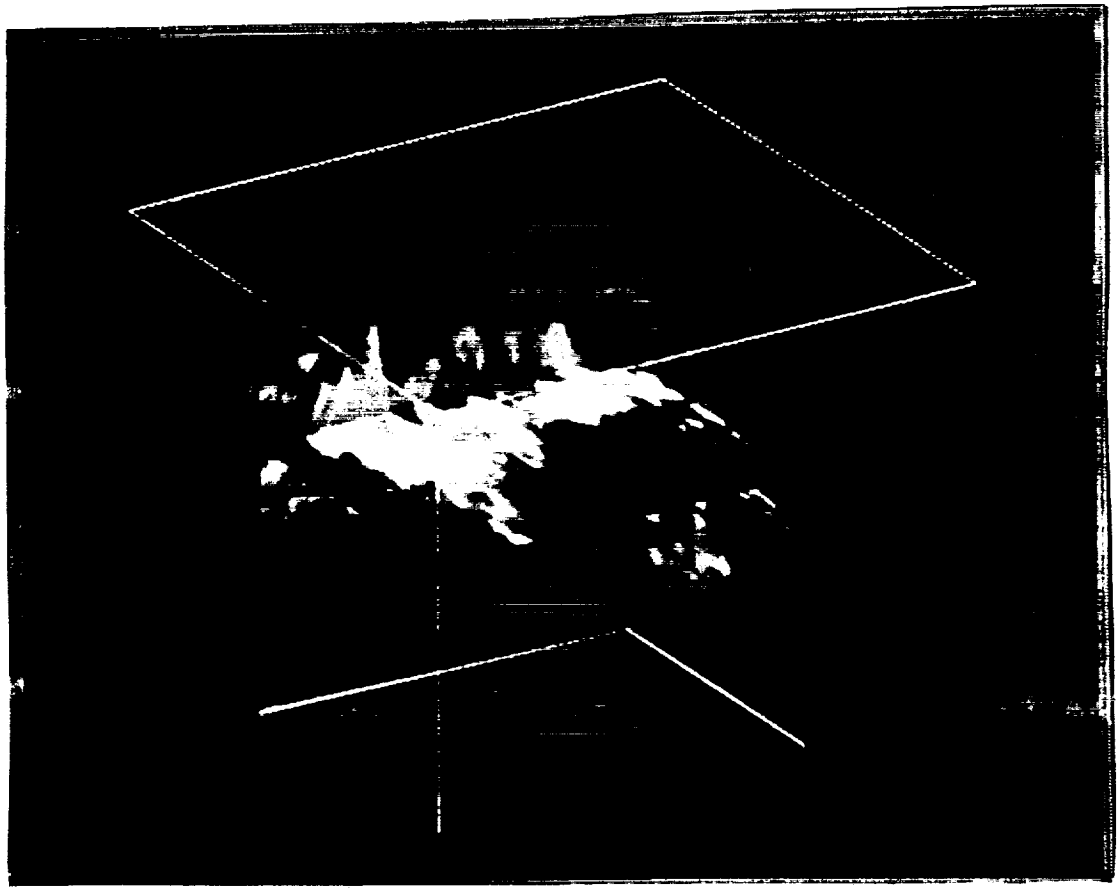


Figure 8: Data slicer (viewing three slices orthogonal to each other)

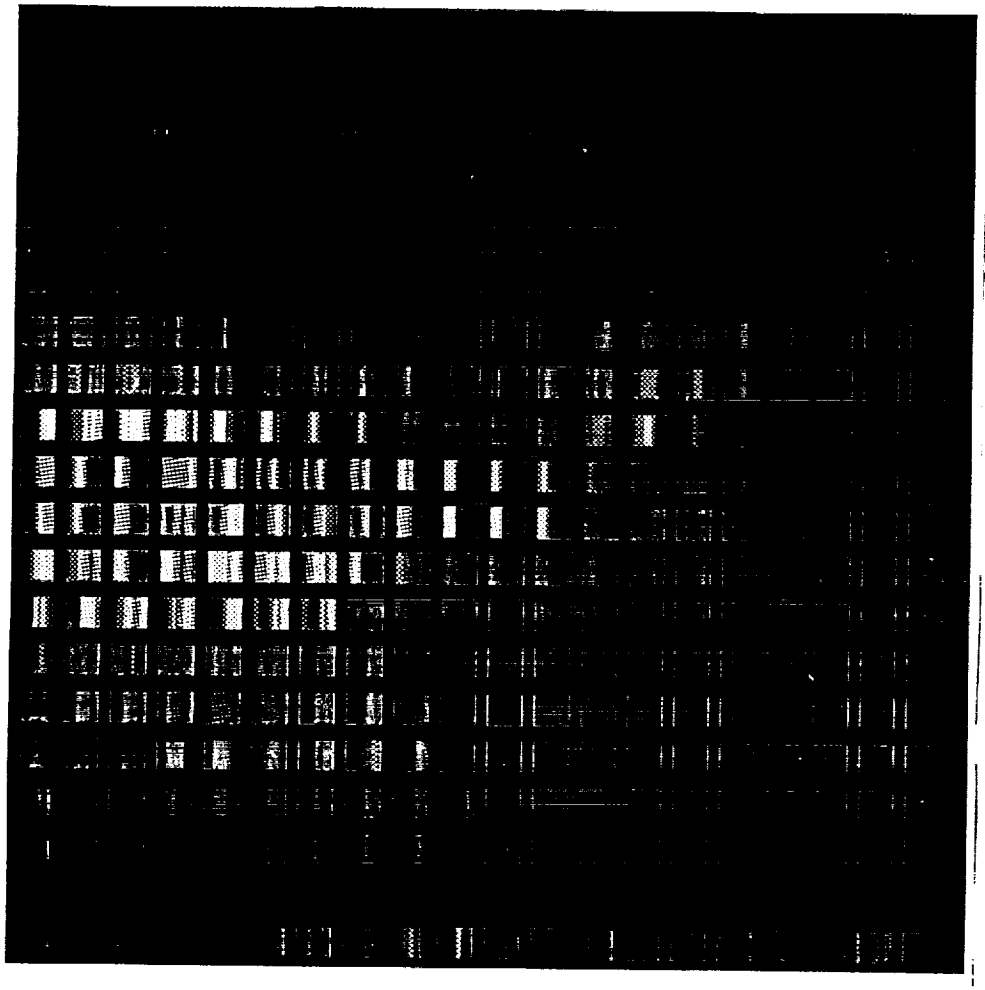


Figure 9: Nine slices of a subset of the data cube (20 x 20 x 9 voxels) are visually correlated. Each red square contains 9 colored slices relating (from right to left) to the spectral responses at that spatial location. Low values are blue and cyan, medium values are green and yellow, and higher values are red and magenta.

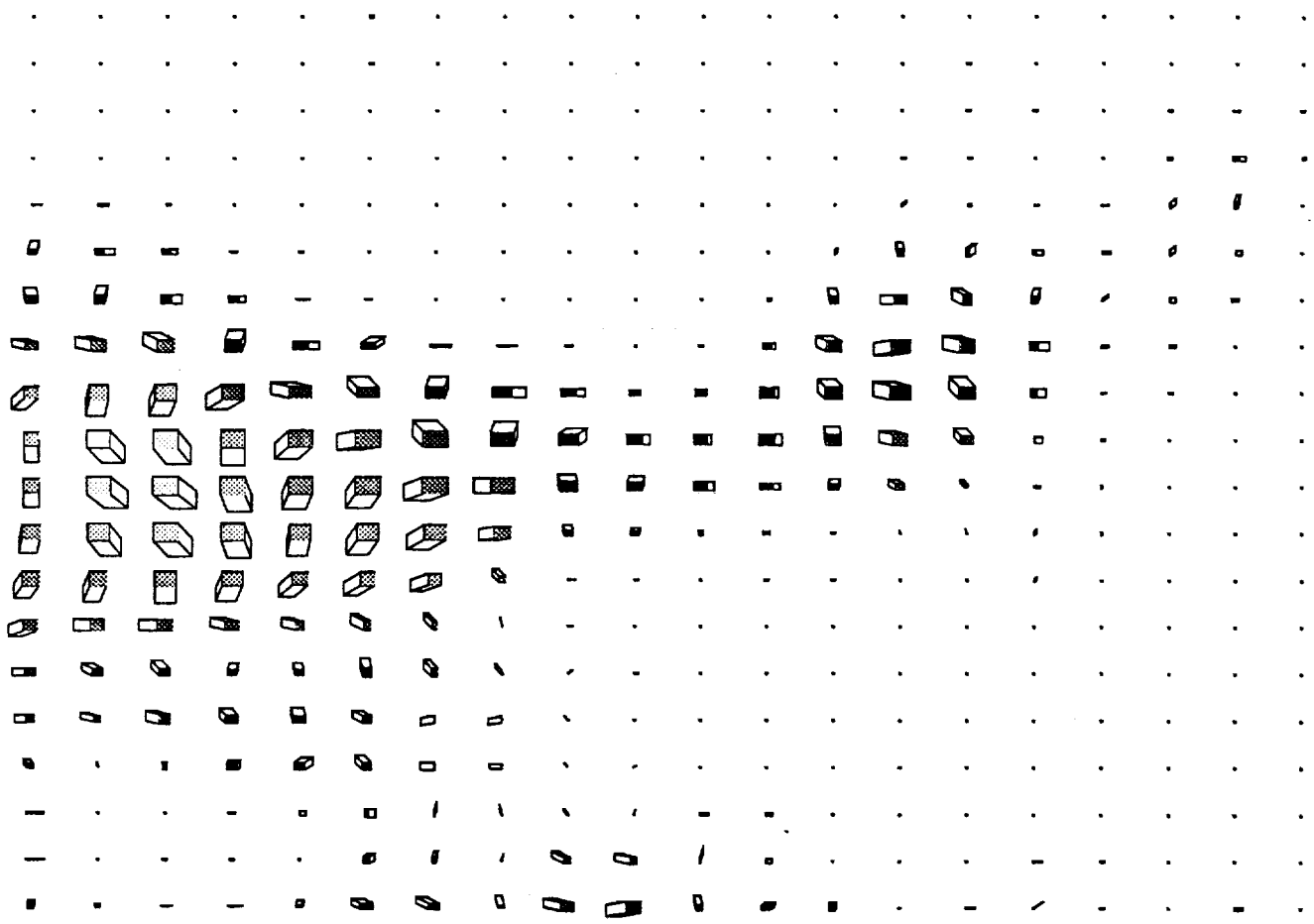


Figure 10: Five slices of a subset of the data cube (20 x 20 x 5voxels) are encoded into five characteristics of a cube: width, height, depth, color and point-of-view. Data of the first three slices was scaled down to a range beween 0 and 20 to indicate width, height and depth. The fourth slice is mapped into a range between 0 and 255 to show color; the last slice contains a range of values between 0 and 360 and indicates the point-of-view. It is interesting to observe an animated version of this representation by visually correlating each five consecutive slices in the data set of 100 slices: growth and rotation relate to changes in the spectral response from slice to slice.

3. Make new visualization tools available to astrophysicists

Many of the tools described in section 2 where available through AVS. AVS is not available at CASA due to its cost, but was made available by the Department of Computer Science for a feasibility study for astrophysical data cubes. Other visualization tools shown in section 2 are part of IDL/widgets, a software package that is available at CASA, however, needs modification to make the tool useful for astrophysical data analysis (e.g. connection to FITS reading programs, data analysis combined with visualization etc.). Before the summer of 1992, Prof. J. Bally and Prof. G. Domik identified a list of visualization tools which were developed/expanded over the summer by Paul Pinkney, a graduate student at the Department of Computer Science.

The additional modules included qualitative as well as quantitative data analysis tools:

- a data slicer capable of slicing 3-d data by an arbitrary plane as seen in Figure 11 (note the arbitrary position of the slice inside the data volume as compared to Figure 7 and 8);
- an isosurface viewer to view a selected plane (represented by a grid) and isosurface in a combined mode (therefore enhancing spatial positioning on the isosurface) as seen in Figure 12;
- an enhanced extraction tool for subcubes;
- an integration tool for isosurfaces, summing up values inside each individual object defined by an isosurface as seen in Figure 7 and 8 in the appendix.

The appendix contains a user's guide to the new visualization tools, including examples.

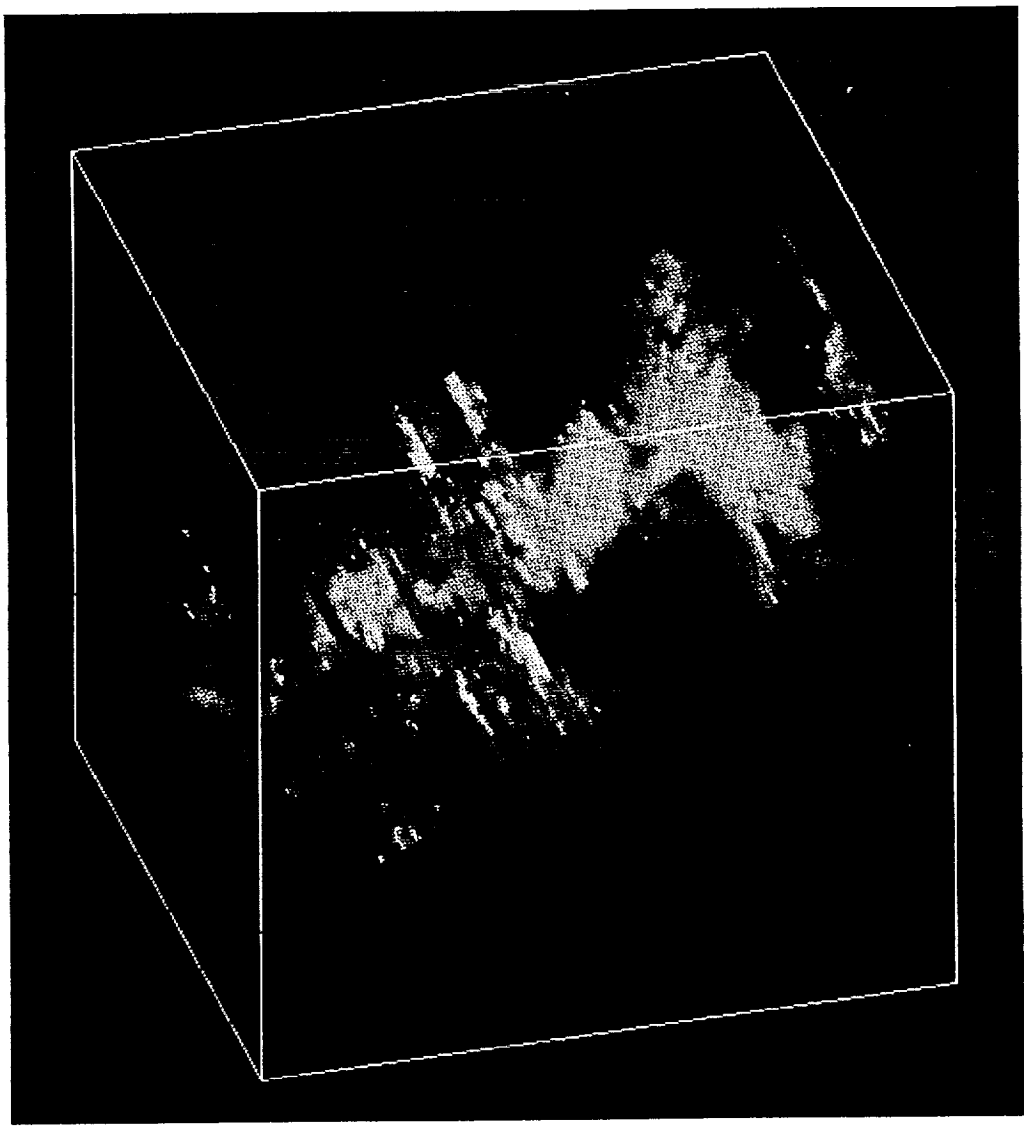


Figure 11: Arbitrary slice inside volume

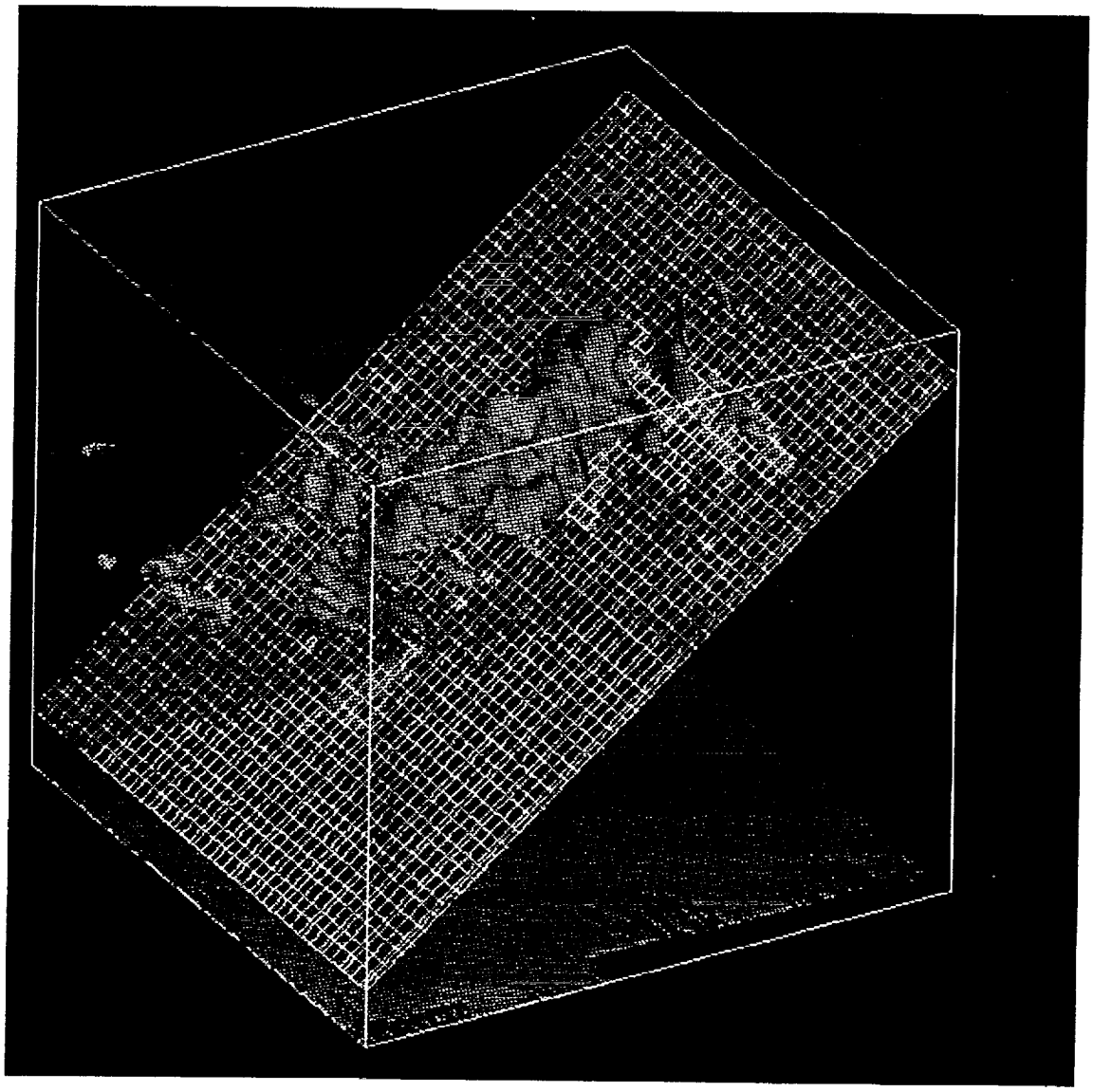


Figure 12: Grid plane and isosurface

4. Leading-edge three-dimensional user interfaces

With the exploration of volumetric data, the problem of how to interact with 3-dimensional data on a 2-d screen became more obvious. Both input devices (e.g. mouse) and output devices (e.g. current 8-bit color workstation displays) as currently available at reasonable prices are limiting the scientist.

An example might illustrate the problem in a better way: A scientist looks at an isosurface display of an astrophysical data cube, as pictured in Figure 2. By rotating the cube and its content, the 2-d screen clearly conveys the effect of a 3-d object. In order to analyze part of the cube in more detail, the scientist wishes to extract a subcube. At this moment several problems hamper the interaction:

- In order to "reach into" the cube (spatial positioning), the rotating movement has to be stopped. This results in collapsing the 3-d information into a static 2-d perspective projection. An output device such as a stereo-screen coupled with stereo glasses or a virtual environment would solve this problem.

- A mouse is moved over a 2-d surface to simulate 3-d movement in various ways: e.g. any 2-d coordinate plane in 3-d space can be selected consecutively to reach any spatial position in 3-d space. While 3-d spatial feedback is already poor on the 2-d screen, the separate movement is also very un-intuitive to a natural gesture of grasping an object.

There is a lack of 3-d input as well as output devices that are effective as well as cheap. Furthermore, we need devices that are similarly standardized as the trackball or a mouse currently are, to get such devices widely distributed in the scientific community.

Salim Alam, a graduate student at the Department of Computer Science, performed a summer study on the availability, effectiveness and costs of such devices. Salim's report is included with this document. We are hoping for Salim to design over the next years a low-cost 3-d interaction device (consisting of hard-and software) that is highly effective for scientific 3-d data interpretation. All future efforts on this topic need to be funded independently to this NASA grant.

5. Dissemination of sponsored activities

Publications during current project phase:

Domik, G., 1992, A Case Study in Astrophysical Data Visualization, Technical report CU-CS-622-92, University of Colorado, Department of Computer Science, Boulder, CO. 80309-0430.

Domik, G. O. and Mickus-Miceli, K.D. 1992, Design and Development of a Data Visualization System in a Workstation Environment, J. Microcomputer Applications, Vol. 15, pp. 81-88.

Domik, G.O. and Mickus-Miceli, K.D., 1992, Software Design and Development in a Scientific Environment: Lessons Learned During the Development of STAR, an Astrophysical Analysis and Visualization Package, Astronomical Society of the Pacific Conference Series, Volume 25, Astronomical Data Analysis Software and Systems I, ed. by D.M. Worrall, Ch. Biemesderfer, and J. Barnes; pp. 95-99.

Domik, G., 1992, Designing User-Centered Interfaces for Astrophysical Software, Workshop Proceedings on "User Interfaces for Astrophysical Software", April 14-15, 1992, Goddard Space Flight Center, Greenbelt, MD, sponsored by NASA Headquarters, Astrophysics Division Workshop.

Domik, G. , (to appear in proceedings of conference), Visualization of Multi-Dimensional Arrays in Astronomy, Conference on "Astronomy from Large Databases II", Haguenau, France, September 14-16, 1992.

Participation at other conferences and workshops by G. Domik during this project phase:

1. Visualization '91, San Diego, October 22-25, 1991
2. Visualization '92 Boston, October 19-23, 1992
3. Going from the pictures to the numbers, a workshop sponsored by the Communications and Information Division Systems of NASA.

Acknowledgement

Many thanks to John Wilson, Walter Fortson and Chris Joslyn for providing many of the color pictures in this report.

References

De Ferrari, L., 1991, *New Approaches in Scientific Visualization,* CSIRO Division of Information Technology, GPO Box 664, Canberra ACT 2601, Australia, Technical Report CSIRO-DIT TR-HJ-91-06.

Domik, G.O., 1991, *The Role of Visualization in Understanding Data,* Lecture notes on Computer Science 555, Springer Verlag, "New Trends and Results in Computer Science", pp 91-107.

Gordon, I.E., 1989, *Theories of Visual Perception,* John Wiley & Sons, ISBN 0 471 92196 3.

Mackinlay, J., 1986, *Automating the Design of Graphical Presentations of Relational Information*, ACM Trans. Graphics, Vol. 5, No. 2, April 1986, pp. 110-141.

Robertson, P.K., 1990, *A Methodology for Scientific Data Visualization: Choosing Representations Based on a Natural Scene Paradigm*, Proceedings of the Conference on Visualization '90, October 23-26, San Francisco, pp. 114-123.

Wehrend, S., and Lewis, C., 1990, *A Problem-Oriented Classification of Visualization Techniques*, Proceedings of the Conference on Visualization '90, October 23-26, San Francisco, pp. 139-143.

Appendix

Contains:

IDL Tools for Volumetric Data Analysis by Paul Pinkney

Potential Research Aspects of 3-D User Interfaces by Salim Alam

IDL Tools for Volumetric Data Analysis*

by
Paul Pinkney
Department of Computer Science
University of Colorado

* Supported by NASA/Astrophysics Division, NAGW-1902

1. Introduction

The Interactive Data Language (IDL) developed by Research Systems, Inc. is a powerful tool for displaying scientific data in a variety of formats. As a part of this software package, the IDL slicer was developed. Using the slicer, a user can view three-dimensional data using a subset of the IDL graphical rendering routines with the benefits of a Graphical User Interface (GUI). Some of the data analysis tools include those to render isosurfaces, data blocks, and orthogonal data slices.

In addition to IDL's data slicer, a series of IDL routines was developed to handle astrophysical data stored in the Flexible Image Transport System (FITS) format. These routines include those that allow the user to view FITS data in the IDL slicer, filter FITS data, and quantify structures within FITS data. All of these IDL routines will be described in detail in the following sections.

2. Getting Started

To use the IDL extensions, the user must first enter the IDL software package. For information on obtaining or using IDL, see your systems manager.

To enter IDL, simply enter **idl** in the login shell, at which time the user should see something like the following.

```
IDL. Version 2.2.2
Copyright 1989-1992, Research Systems, Inc.
All rights reserved. Unauthorized reproduction prohibited.

IDL>
```

As a convention throughout this document, words in a bold character format will represent user entries.

3. Viewing Data Stored in FITS Format

Many FITS data sets are available as a series of two-dimensional data slabs. To produce a three-dimensional volume of data which may be used in the IDL slicer, they must be loaded sequentially and put into a single three-dimensional array. The slicer can then render isosurfaces, slices, or data blocks within this data volume. To read in this type of FITS data set and enter the IDL slicer, the routine *fits_slicer* was developed. This routine allows the user to select which FITS data set is to be used in the extended IDL slicer.

3.1. Command Line Interface

fits_slicer, data_set, start, finish, x = [x1, x2], y = [y1, y2]

Positional Arguments

data_set	Name of FITS data set; character string
start	Extension of first FITS data file to be read; integer
finish	Extension of last FITS data file to be read; integer

Optional Keyword Arguments

x	Computational window for data in first dimension, integer vector
y	Computational window for data in second dimension, integer vector

This routine reads FITS data into a single three-dimensional array. The first argument, *data_set*, is a character string that defines which data set is to be read. If this data set is not in the user's current working directory, either a relative or absolute path to the data set must be given. For example, if the data set *12co* resides in the directory */images/L1551*, then *data_set* must be **'/images/L1551/12co'**.

The arguments *start* and *finish* are the first and last extensions of the two-dimensional slabs of FITS data to be read. Continuing with the above example, if the files *12co.1*, *12co.2*, ..., *12co.25* are in the directory */images/L1551*, and the user wished to read the entire sequence of files, then *start* and *finish* would be **1** and **25**, respectively. The two-dimensional slabs of data would be placed sequentially in a three-dimensional array as shown in Figure 1. For many data sets stored in FITS format, the x- and y-axes are spatial dimensions, but the z-axis, in general, will not be a spatial dimension. Instead, it may imply a direction of increasing velocity, for example.

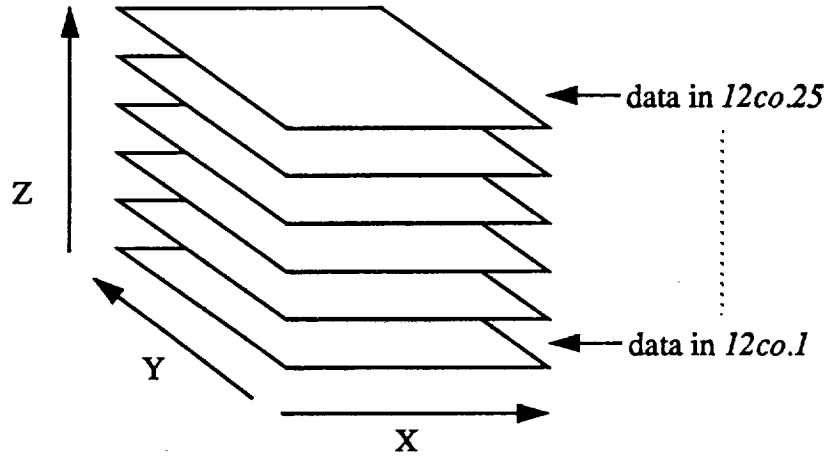


Figure 1. Organization of FITS Slabs in Data Volume

Optionally, a subset of the horizontal domain may be viewed with the slicer instead of the full domain. Specifying a windowed domain is done through the use of the keyword arguments *x* and *y*. These arguments are IDL vectors which define the domain window in horizontal computational space; that is, they specify beginning and ending indices in the first and second dimensions. For example, consider a data set in which the two-dimensional slabs of data are 200 by 400. If a 100 by 100 portion at the center of the data set was to be viewed with the slicer, then the user could enter **[50, 150]** for *x* and **[150, 250]** for *y*.

Completing this example, the full command entered at the IDL prompt would be

IDL> **fits_slicer, '/images/L1551/12co', 1, 25, x = [50, 150], y = [150, 250]**

3.2. Graphical User Interface

The original IDL slicer is described in *IDL Basics* by Research Systems, Inc. This slicer was modified for both the general user and for those who analyze FITS data. These enhancements are discussed in the following sections.

3.2.1. Rendering Data Slices

The original slicer was limited to displaying orthogonal data slices, that is, cross sections that were perpendicular to one of the coordinate axes. A new interface was developed to allow arbitrary rectangular slices to be rendered.

The cross sections of data that may be viewed with the extended slicer are based on a rotational axis. This axis is a single line within the slice to be displayed. Instead of the entire slice being perpendicular to a coordinate axis, the rotational axis is parallel to an axis. Three parameters for a slice of data may be specified:

- The orientation of the rotational axis
- The position of the rotational axis
- The angle of the slice about the rotational axis

Finally, a button is pressed to signal the program to display the slice. The entire interface is explained in the following sections.

3.2.1.1. Orientation

The orientation of the rotational axis is specified by pressing the right mouse button until the desired orientation appears on the interface. The three possible orientations are illustrated in Figure 2, the rotational axis being in the center of the plane that identifies the orientation.

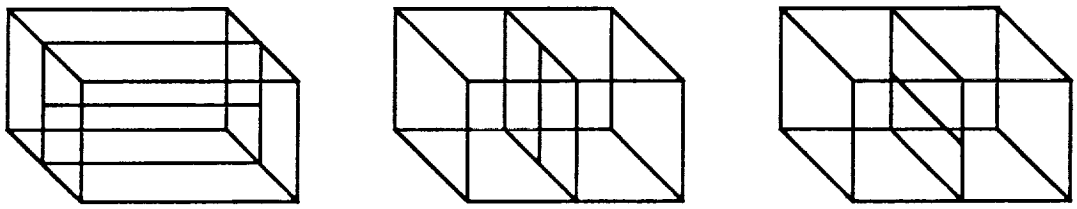


Figure 2. Extended Slicer Interface Orientations

Figures 2(a), (b), and (c) represent the rotational axis being parallel to the x-, y-, an z-axis, respectively. In the actual interface, the orientation plane is gridded, but these grid divisions are not represented to simplify the diagrams.

3.2.1.2. Position

The position of the rotational axis may be moved anywhere within the domain of the other two coordinates. For example, if the user chooses to have the rotational axis parallel to the x-axis, it may then be given any (y, z) location within the domain of the three-dimensional volume. Figure 3 shows how the rotational axis may be positioned within the data volume.

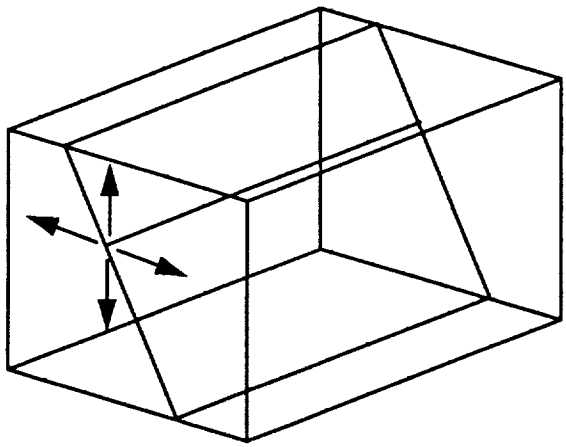


Figure 3. Selecting a Data Slice Angle Using the Extended Slicer Interface

3.2.1.3. Angle

The last parameter that may be specified when selecting a slice is an angle. This angle is that of the data slice rotated around the rotational axis. Figure 4 shows how this angle is specified.

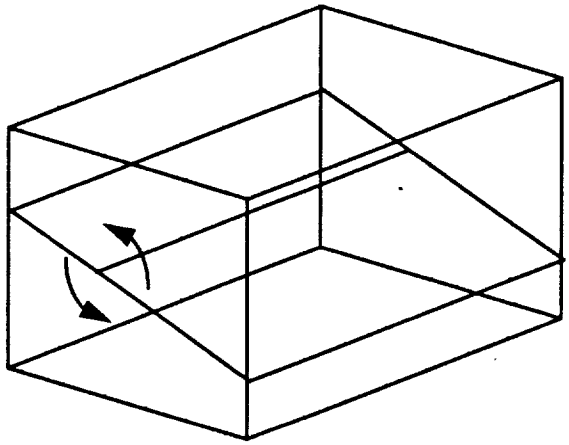


Figure 4. Selecting a Data Slice Angle Using the Extended Slicer Interface

As is done when selecting the rotational axis position, the user clicks in the foward-facing plane that is perpendicular to the rotational axis, but in this case, with the middle mouse button. The wireframe plane that represents the selected slice of data may then be rotated around the rotational axis. Once the desired angle is chosen, the mouse button is released.

3.2.1.4. Rendering the Slice

The orientation, position, and angle parameters have been specified to the user's liking, the button labelled *Render Slice* is then pressed to render the data slice. The slice will then be rendered as a shaded cross section in the large graphical window.

3.2.2. Saving Data Slices

Once a data slice is rendered, it may be saved to a file as a two-dimensional FITS data slab. Saving the slice is done by using the interface shown in Figure 5.

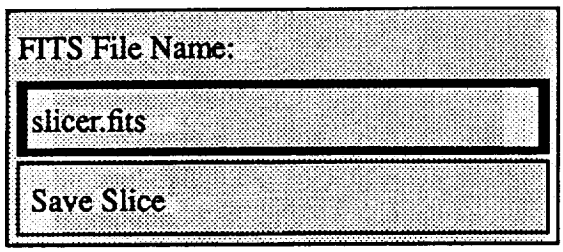


Figure 5. Interface for Saving Data Slices

Using the IDL text bar, enter the name of the file in which the data is to be saved. The defaults file name is *slicer.fits*. Once the file name is entered, press the button labelled *Save Slice*.

3.2.3. Rendering Isosurfaces

By selecting *Isosurface* on the main slicer interface, the user may view isosurfaces within the data volume. The interface that allows the user to enter the isosurface value is shown in Figure 6. Note that this interface differs from that in the original IDL slicer.

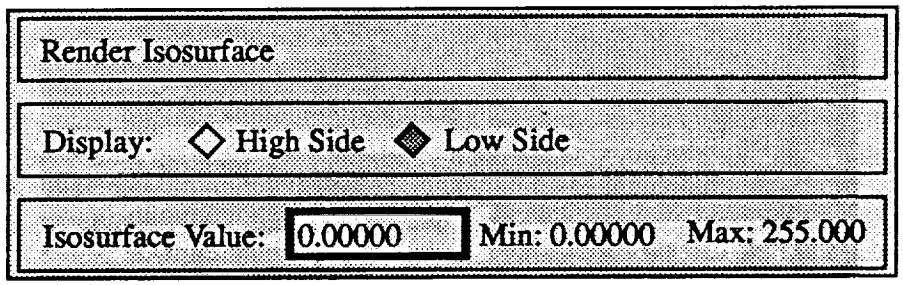


Figure 6. Interface for Rendering Isosurfaces

Again, a text bar is used. A value between the displayed minimum and maximum values may be entered. The user also may select whether the isosurface is to enclose values higher or lower than the threshold. If the isosurface is to enclose values lower than the threshold, press the button labelled *High Side*; otherwise, press the button labelled *Low Side*. Finally, pressing *Render Isosurface* displays the isosurface in the large graphical window.

3.3. Related Routines

The routine *fits_slicer* uses two other routines that may be of interest. The first routine is that which reads in the sequence of FITS slabs. Second, the extended IDL slicer is the GUI that allows the user to render the data volume in various formats. The following sections describe these routines.

3.3.1. Reading FITS Data

The function *read_seq* reads in a sequence of FITS data slabs and puts them into a common three-dimensional volume.

volume = read_seq (data_set, start, finish)

Arguments

data_set	Name of data set; character string
start	Extension of first FITS data file to be read; integer
finish	Extension of last FITS data file to be read; integer

The arguments for this routine are the same as *data_set*, *start*, and *finish* for *fits_slicer*. The data slabs are placed in the three-dimensional array in the manner illustrated in Figure 1. The resulting data volume is returned in *volume*.

3.3.2. Rendering Data

The procedure *arb_slicer* is an extension of the original IDL slicer.

arb_slicer, volume

Arguments

volume	Volume to be used in the slicer; three-dimensional array

If the user has a three-dimensional volume of data in *volume*, it may be fed directly to the extended IDL slicer using this interface.

4. Filtering FITS Data

To filter values out of a sequence of FITS data slabs, use the routine *fits_filter*. This routine loads a sequence of two-dimensional FITS slabs into a single three-dimensional data array in the same manner as *fits_slicer*. Data values either above or below a given threshold are filtered out of the data volume and written to disk as a three-dimensional FITS data array.

4.1. Interface

fits_filter, data_set, start, finish, thres, new_file, high = high, low = low

Positional Arguments

data_set	Name of data set; character string
start	Extension of first FITS data file to be read; integer
finish	Extension of last FITS data file to be read; integer
thres	Threshold value for data filtering; same type as data volume
new_file	Name of file for filtered data; character string

Mutually Exclusive Keyword Arguments

high	Specifies values higher than *thres* to be filtered out of data; flag
low	Specifies values lower than *thres* to be filtered out of data; flag

4.2. Description

The argument *data_set*, *start*, and *finish* are identical to those in the routine *fits_slicer*.

Arguments *high* and *low* are mutually exclusive; that is, either one or the other of them is specified, but not both. These flags determine how the data is filtered. If the user specifies **/high**, then values higher than the given threshold value are filtered from the data. Similarly, if **/low** is specified, values lower than the threshold are filtered. The argument *thres* is the threshold value. All values filtered out of the volume are replaced by *thres* in the resulting data array.

Once the data is loaded into a single three-dimensional array and is filtered, the resulting data volume is written to the file specified by *new_file*.

Using the sample data set from Section 3.1, consider the following example. If values above 2.1 were to be filtered out of the data and the resulting data volume were to be written to the file *myfile.dat*, then the user would enter

IDL> **fits_filter, '/images/L1551/12co', 1, 25, 2.1, 'myfile.dat', /high**

5. Quantifying FITS Objects

Displaying an isosurface from a three-dimensional data volume can provide information about the behavior or structure of a system, but often a more quantified approach is necessary. For this purpose, the routine *fits_objects* was created. This routine sums the values inside each individual object defined by an isosurface.

5.1. Interface

fits_objects, data_set, start, finish, thres, x = [x1, x2], y = [y1, y2], low = low, high = high

Positional Arguments

data_set	Name of data set; character string
start	Extension of first FITS data file to be read; integer
finish	Extension of last FITS data file to be read; integer
thres	Bounding value for data objects; same type as data volume

Required Mutually Exclusive Keyword Arguments

low	Specifies that objects are defined as those structures bounded by and enclosing values lower than *thres*; flag
high	Specifies that objects are defined as those structures bounded by and enclosing values higher than *thres*; flag

Optional Keyword Arguments

x	Computational window for data in first dimension; integer vector
y	Computational window for data in second dimension; integer vector

5.2. Description

When an isosurface is rendered, the structures that are displayed are those bounded by a threshold value and containing values either higher or lower than that value. The routine *fits_objects* is used to add up the values inside each of these structures.

The arguments *data_set*, *start*, and *finish* are the same as those in the routines *fits_slicer* and *fits_filter*. The argument *thres* is of the same data type as that of the three-dimensional data volume and specifies the bounding value for the data objects in the same way a threshold value is given for isosurfaces. Either values above or below this value will be contained in the data objects, depending on whether **/high** or **/low** is specified.

Domain windowing may be done using the arguments *x* and *y*. These arguments are identical to those in *fits_slicer*.

As an example, suppose the following was entered:

IDL> **fits_slicer, '/images/L1551/12co', 1, 25**

Notice that variables *x* and *y* were not specified, so the entire data volume will be used in the extended slicer.

Using the *Isosurface* option in the slicer, the user then obtained the isosurface shown in Figure 7. This isosurface encloses values higher than 2.0. Notice that two distinct objects are revealed: a large structure and a much smaller one. If the user entered

IDL> **fits_objects, '/images/L1551/12co', 1, 25, 2.0, /high**

the output shown in Figure 8 would be displayed. In this figure, three objects were identified, but one of these contained a single value within the data volume. The other two objects are those displayed in Figure 7, but revealed in a quantified format.

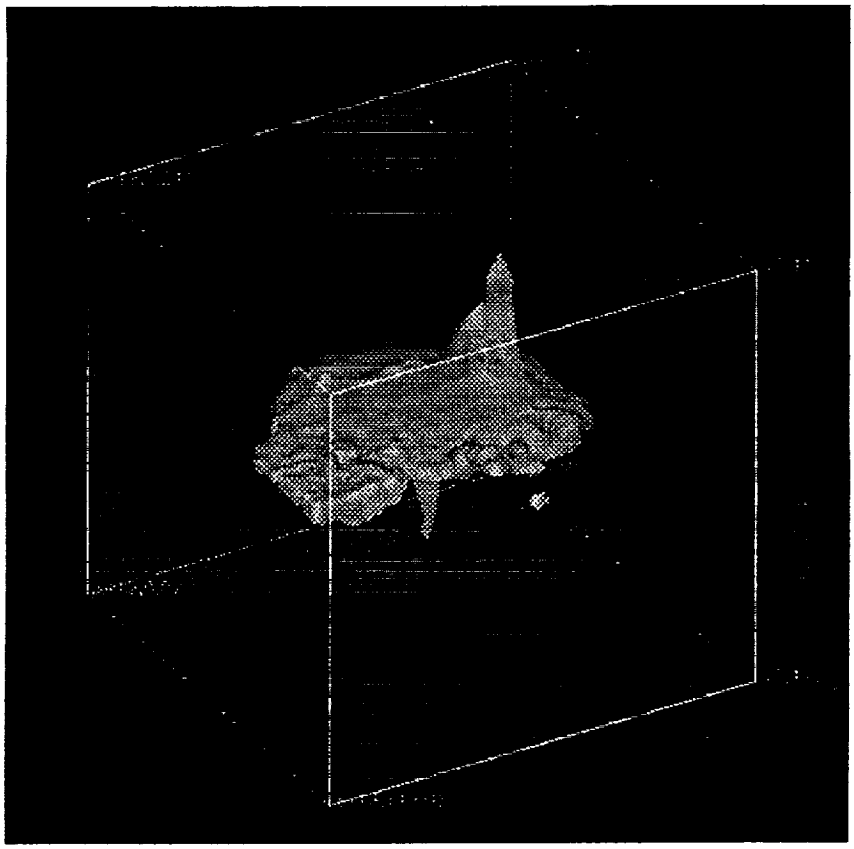


Figure 7. Graphical Representation of L1551 Isosurface Enclosing Values Greater Than 2.0

Objects enclosing values higher than 2.00000

Object 1: 45004.1
Object 2: 2.00966
Object 3: 46.4203

Total number of objects: 3
Total sum for all objects: 45052.5

Figure 8. Quantified Representation of L1551 Isosurface Enclosing Values Greater Than 2.0

6. Conclusion

In conclusion, it is the hope of everyone involved in the development of these tools that they become quite useful in the analysis of astrophysical data sets as well as other types of three-dimensional data. Experience has shown that experimentation with these tools is the best way to reveal their usefulness. Play a little while and see what is discovered.

Potential Research Aspects of 3D User-Interfaces*

Salim Alam

November 12, 1992

Contents

*Supported by NASA/Astrophysics Division, NAGW-1902

1 Introduction

This work has been motivated by a lack of commonly available, low cost and effective user-interfaces for three-dimensional interaction, especially the manipulation of large volumetric data-sets. Workstations which allow fast access to three dimensional graphics have become increasingly common and are being used for a wide variety of tasks, from solid modeling to medical imaging to animation [1]. However, while the hardware has become increasingly sophisticated, the user-interface has generally retained its two-dimensional heritage — a number of commercial programs are available which provide very advanced methods of visualizing data but restrict the user to interacting with the data in only certain fixed ways, which are ultimately unintuitive.

Although high-end research institutions have been working for a long time on problems of user-interface design for three-dimensions, the state-of-the-art interfaces have been custom-designed for certain applications and are also very expensive. However, this is starting to change — sophisticated three-dimensional user-interface hardware is coming into a price range that is affordable by the average user.

With this work we hope to highlight some of the problems with current implementations, look at a number of state-of-the-art hardware and software solutions and attempt to point out some of the promising new tools and paradigms. This may then be resolved into a system that provides a better cost/performance ratio. Of course, the measurement of "performance" needs to be defined in terms of some explicit criteria.

2 Defining the Problem

Experience with some of the commonly available visualization tools shows areas of deficiency that could be improved. Below, we describe some of these deficiencies.

IDL Slicer. IDL is a popular interactive language for visual data analysis. The IDL system also provides some built-in tools for three-dimensional visualization. One of these tools that we frequently use is the *data slicer*.

The data slicer allows one to visually examine volumetric data in a variety of ways [2]. The data can be examined in any orthogonal plane. Furthermore, a subsection or "sub-cube" can be selected and examined. A simple three-button mouse is used to interact with the slicer.

The interface for selecting the sub-cube, while technically competent, is cumbersome to use because it is difficult for the user to manipulate the cursor in three dimensions – the visual cues are not very strong. Furthermore, it is difficult to select and pinpoint the exact area that one wants to examine because of the un-intuitive selection method.

AVS Geometry Viewer. Another very powerful tool that we have been using is the Application Visualization System (AVS). The Geometry Viewer allows rotation and translation of objects in an arbitrary manner, using a combination of mouse and keyboard keys. While this is an excellent tool for the most part, we have found that when precise positioning of complicated three-dimensional objects is needed, the rotation and translation methods become very difficult to use. For example, on one occasion it took upwards of twenty minutes to precisely position two six-atom molecules using the mouse controls.

While it is possible to use a Spaceball three-dimensional input device to interact with AVS [3], there is anecdotal evidence [1] that suggests that the precise positioning and rotation still remains a cumbersome process. Part of the problem stems from the fact that the display is monoscopic.

It is clear that these user-interface deficiencies are directly related to the use of two-dimensional input and output devices, and can be grouped into three different areas:

- **Positioning.** This means that it is difficult to precisely position an object in 3D space. The main reason is the difficulty of accurately perceiving where in space the cursor actually is.
- **Rotation.** Using mouse and keyboard areas to interact with an object is cumbersome and not intuitive. Furthermore, it forces the user to remember complex mouse and keyboard sequences.

[1] Replies received from queries posted on Usenet *alt.3d* newsgroup.

- **Selection.** Selecting an area of the 3D space to examine further is probably the most difficult of all, since it requires both accurate positioning and complicated selection commands to remember. Moreover, the depth cues are not sophisticated enough to sustain accurate positioning.

We need to look at ways to eliminate or minimize the deficiencies. New user-interface hardware that is effective and affordable is becoming increasingly common, and there is also continuing research in new software paradigms for user-interfaces.

3 State-of-the-Art Interfaces

Cutting-edge research in user-interface design seems to have a number of things in common. Firstly, the user-interface is usually custom-designed for a specific application. Secondly the main component is designed to take advantage of a very specific technological break-through — for example, holography. And thirdly, the cost of such a system is quite high since most of the technology is not off-the-shelf.

The following subsections describe some important research contributions in three separate areas of user-interface design.

3.1 Input Devices

A three-dimensional input device is a six-degree-of-freedom device that can sense position as well as orientation. Usually one needs special hardware to track the position of the user's physical cursor (such as the hand or a stylus). The most commonly used tracking device is manufactured by Polhemus and uses a magnetic field to locate the sensor.

The Bat. The "Bat" is a tool designed by Colin Ware and Danny Jessome at the University of New Brunswick [4]. It is basically a three-dimensional mouse, constructed using a Polhemus device. The mouse is actually a sensor that senses a signal output from a fixed source. A controller measures the position of the sensor and communicates with a host computer.

The effectiveness of the bat as an input device was tested using 3D placement as the primary operation. Two important factors that play a role in accurate operation were found to be kinetic depth (ie, a feeling of 3D when object is moved) and a simple protocol.

3-Draw. 3-Draw is a tool for designing three-dimensional figures, specifically for CAD. It was constructed as part of a research project at MIT [5]. It uses two six-degree-of-freedom sensors, both of which are part of a Polhemus 3Space Tracker. One sensor is attached to a plate and is held in one hand, while the other is attached to a stylus and is held in the other hand. Using two sensors allows the user to use one of them to function as a plane in the virtual world, while the other is used to draw three-dimensional points relative to the plane. Software was written that interactively reflects the position of both the sensors as well as the plotted points.

The effectiveness of the tool was measured by making CAD drawings. It was found that working directly in three dimensions was much more faster and natural.

3.2 Output Devices

Some of the problems with having a 2D input device can be alleviated by using a three-dimensional output device. Research has indicated that for simple tasks such as positioning an object with a mouse, a stereoscopic display cuts down on the time needed for accurate positioning of the object [6]. It is not unreasonable to extend this thesis by claiming that in general for these kinds of tasks any 3D output device can make it easier to interact with a data-set. More research is needed, however, to ascertain exactly what the benefits are for various devices.

We discuss two state-of-the-art output methods below.

Stereoscopic Monitors. Stereo monitors allow the user to observe a scene in three dimensions. They work by transmitting two different images, one to each eye. This parallax leads to a 3D effect.

There are many different ways that monitors send different images to each eye. Some monitors use special shuttered glasses which automatically keep in sync with the picture being transmitted — while the scene for one eye is being displayed the shutters for the other eye are closed, and vice versa. Monitors made by Stereographics use this type of technology [7]. Other monitors place the shuttering mechanism directly on the monitor, and require the user to wear simple passive polarized glasses — monitors made by Tektronix fall into this category. [2]

[2] One testament of the usability of stereoscopic displays is that they are starting to be used in commercial ventures. For example, Vexcel Corp. in Boulder currently uses the Textronix stereographic system, along with a mouse and a dialbox for input, to create

The most recent research has resulted in auto-stereoscopic displays that do not require the user to wear any kind of glasses. Three such displays are currently available, with more models being in the research stages. The most promising of these is a model developed by Dimension Technologies, Inc. which uses a vibrating mirror to project various cross-sections of the two-dimensional image onto 3D space. Both monochrome and color versions are available, with a resolution of 320x480 pixels in 16 gray-shades or 32 colors.

Holography. While still in mostly the research state, holography offers another method of three-dimensional vision, including a "walk-around" effect through a number of degrees. Holograms are based on use of laser light, but there are a number of ways to reproduce a holographic image [8]. The most common method is film based [9], but it has serious limitations. Another method, which is being researched by engineers at Texas Instruments, is based on the projection of laser light onto a spinning helical surface which is enclosed in a dome-like structure. Finally, researchers at MIT have been working on generating holographic images using computer-controlled generation of light-patterns which are focused by optics to generate real-time holographic images — this process is called *holovideo.* These holograms however require enormous computing power and storage, and are still limited by a viewing angle of only 15 degrees. However, recently there have been some breakthroughs which have decreased the storage and computation needs while increasing the viewing angle [10].

3.3 Virtual Reality

Virtual Reality combines both three-dimensional input and output devices to provide a complete virtual environment for the user to manipulate. The main advantage of virtual reality is that the user is completely "enclosed" by the virtual environment and can interact with it using normal body movements. The main disadvantage of virtual reality is the difficulty of completely simulating the environment — there are many restrictions. However, as we shall see, virtual reality interfaces are being used and offer much potential for solving many of the problems associated with positioning, rotation and selection of data.

three-dimensional surface maps from satellite data. The quality of the three-dimensional images is very good – when correctly calibrated the images appear solid and flicker-free.

Two different approaches to a virtual reality environment are described below.

Virtual Wind Tunnel. In [11] researchers describe how they have developed a virtual reality environment to visualize and interact with unsteady flow simulations. The system consists of a stereo head-tracked display which is worn on the user's head using a "boom" device. The boom is manufactured by Fake Space Labs, and allows for the mounting of "two CRTs on a counter-weighed yoke attached through six joints to a base". The large degrees-of-freedom allow for comfortable movement for the users (within a limited spatial boundary) and the exact position and orientation of the user's head is transmitted by the boom directly to the host computer using a serial port.

In addition, the user interacts directly with the environment by using a VPL dataglove, which uses the Polhemus 3Space Tracker to determine the postion and orientation of the hand, and fiberoptic sensors to determine the flexing of the finger joints.

The entire hardware is attached to a Silicon Graphics Iris 380 VGX workstation with eight processors and a rating of 200 VAX MIPS. All the computation and rendering is done in real-time using this machine.

The user can interact with the simulation in a number of ways. The co-ordinate system for the data can be transformed by the user by simply making a fist and gesturing. Also, "rakes" and "seed points" can be placed by simply picking them up and dragging them as one would normally do with a real object.

Using this setup it was found to be very easy to experiment with various factors in the simulation — the user obtained an intuitive understanding of the flow field. The researchers found the interaction method to be very satisfactory.

VR on Five Dollars a Day. Randy Pausch at the University of Virginia has developed an experimental virtual-reality setup for research in devices to help severely disabled children [12]. His system consists of a 386-based IBM PC, a Polhemus 3Space Isotrak, two LCD displays, and a Mattel Powerglove. The software, including rendering libraries, has been custom-developed. This entire system costs around $5000.

Use of this system has outlined several criteria for virtual-reality user interfaces. Firstly, the time-lag for the head-tracker is very important and

should be minimized. Secondly, while a stereoscopic display is not essential, a reference plane is. Also, some kind of method is needed to allow the user to escape the spatial constraints — for example a "vehicle" which can be moved rather than the user.

4 Software Research

The software interface also plays a large role in dealing with three-dimensional data. An interface might have the best hardware, but if the software is not friendly and intuitive the overall functionality will be diminished and the system will be difficult to use.

Researchers have been approaching the problem of creating a better software interface from various directions. Some of these approaches are described below.

2D Direct Manipulation. The Human Interface Group at Apple Computer, Inc. have designed and experimented with a system for moving solid-modeled three- dimensional objects using a monoscopic display and a single-button mouse [13]. From the very start, consideration was given to using the correct metaphors – for example, since people move things using hands, an icon of a hand was used as a cursor. Furthermore, while performing different tasks, the icon was changed to reflect the physical aspect of the task, for example, pushing, lifting or rotating. However it was found that users were still confused about actual directions and ranges of movement, and also about the position of the "hot-spots". To resolve these problems, bounding boxes and "narrative" handles were used to further clarify the interface.

VPL System Interface. An experimental system by VPL was evaluated for the task of visualizing and interacting with data consisting of the neuroanatomical structure of the human brain [14]. [3] While the researchers found this VR system to be "by far the best and easiest" method of interacting with the data, they did acknowledge some problems with the interface. Some of the problems were associated with the nature of the hardware, for example, limited scope of movement, problem with tripping over cords, slow update rates, etc. In terms of the software interface, the "flying" function

[3] VPL also sells a similar commercial system, and has since announced a much lower-cost product with even better performance. However, at even the amazing price of $58,000, only large R&D institutions can afford to purchase it.

provided by the software was found to be difficult to use due to the inability to control many aspects of the speed of movement. Also, it did provide the actual experience that one would associate with "flying". [4] The researchers suggest that a better metaphor for a virtual reality interface might be a "push/pull" interface where users could in-effect directly interact with the data. Another problem they found was the inability to "fly" in one direction while looking in another direction. This ability could provide a very unique method of visualization if it was available. Finally, the use of visual cues to warn the user when attempting to move out of the range of the tracking devices was also suggested.

Other Research. Simply modifying the software interface to minimize the kinds of problems discussed above could have a significant impact on its usability. For example, a technique for rapid controlled movement is suggested in [15], which could solve the problem with "flying" that was encountered with the VPL interface. Use of force-sensing and force-feedback devices could provide a better interface for tasks such as the handling of virtual objects [16]. And use of neural-nets could allow for a large library of gestures that could be customized for individual users [17],

5 Criteria for a New Interface

It is clear that the use of three-dimensional input and output devices, used both separately and together, provide a distinct advantage over two-dimensional interfaces. It should be possible to preserve this advantage while developing a three-dimensional interface that would be low-cost and easily available.

By studying the research results obtained we get an idea of what an interface for three-dimensional manipulation should be. We have seen a number of interfaces that have been optimized for a particular task. In our own problem definition, we are interested in the use of a three-dimensional cursor which can be used to place, rotate and select the data-set. There does not seem to be any reason why it would not be possible to develop a general interface that would be usable for a variety of tasks while providing excellent performance for our primary problem.

[4] Although Randy Pausch's system uses a similar metaphor, he does not specifically mention these problems. However, he uses the "vehicle" metaphor to circumvent the limited range of physical movement available.

Is virtual reality the ultimate interface? Perhaps, although a totally usable virtual reality interface cannot be expected in the near future. It seems that we might be better off identifying the basic advantages of a virtual reality interface and using low-cost and easily available hardware to simulate those advantages. For example, use of a glove to "grab" and interact with data on a normal monoscopic display would be much preferable to simply using the mouse — it would not matter too much that the user was not actually "enclosed" by a virtual reality.

In order to test the effectiveness of the interface we would like to use criteria similar to that used by McWhorter, et al. The primary quantitative criteria would be the time and accuracy in positioning, rotating and selecting an area of the data-set. A secondary qualitative criteria would be the elegance, intuitiveness and usability of the software paradigm.

6 Commonly Available Hardware

User-interface hardware is now becoming available at a much lower cost than the high-end hardware being used in state-of-the-art research projects. Off-the-shelf components that can be used without a lot of custom design work are described below. We will not describe any high-end components, for example those that have already been described in the previous sections — these are too expensive for normal use. For example, the Polhemus Tracker costs about $12,000, the BOOM device costs about $27,000 and the Tektronix stereo display costs about $8000. The devices we select below are available at a cost of $1500 or less.

6.1 Input Devices

- **SpaceBall 2003.** The SpaceBall is a "three-dimensional" input device. It is similar to a trackball, but does not move — the device senses the actual force exerted by the user and also the direction of the force, which allows the software to move or rotate objects in three dimensions. It is available from Spaceball Technologies and works with a large number of platforms, including Silicon Graphics and Dec. The approximate cost is $1500.00

- **Logitech 3D Mouse.** This is a six-degree-of-freedom sensor that is available for a number of platforms including the IBM PC and Silicon Graphics. Since it is serial-port based, it is easy to interface it to

other platforms, although the driver software needs to be written. It can be used as a normal mouse or as a six-degree-of-freedom three-dimensional sensor. The cost is $1000.

- **Gyropoint 6D Mouse.** The GyroPoint is a three-dimensional mouse that uses gyroscopes and an optical-sensing interface to provide six-degrees-of-freedom. It has been announced with a projected release date of first quarter of 1993, and a price of $1000 for the developer version. Versions will be provided for both IBM PCs and Apple Macintosh.

6.2 Output Devices

- **3DTV Stereoscope.** The Stereoscope hardware consists of a hardware card and a pair of LCD shutter glasses for about $450. A software development kit is available for $350. For use in a virtual-reality environment these are also available with Headband, Eyeglass or Wireless models.

- **Haitex X-Specs.** These are manufactured by Haitex Resources, and sell for about $80. Using these LCD glasses and the proper software gives a stereoscopic display of computer-generated graphics. A generic interface is provided which can be used with a variety of platforms like the IBM PC or the Macintosh.

- **Sega 3D Glasses.** This is a discontinued but widely available item, available for less than $100. These provide simple stereoscopic display using any monitor. They are based on the shutter concept. Public domain serial-port interface circuits are available, as well as driver and user-interface software for the IBM PC, although they can be connected to pretty much any platform using the serial port.

6.3 Virtual Reality Hardware

- **Mattel Powerglove.** The Powerglove was originally manufactured for the Nintendo and although discontinued, is still widely available. It uses audio sensors to determine the position of the hand and the flexing of the finger joints. Interfaces are available for the IBM PC and the Macintosh computers. The cost for the glove is typically less than $50, and the interfaces range from public-domain circuits to an interface for

the Apple Macintosh from TransInfinity (for $160). Abrams/Gentile Entertainment (AGE) also sell a serial interface that can be used on any platform with a serial port (for $200).

- **The Private Eye.** Reflection Technologies manufactures a head-mounted display device that can be used for virtual reality. The display is monoscopic and uses a spinning mirror to provide a resolution of 720x280. Cost is about $400.

6.4 Miscellaneous

- **Force Sensors.** Interlink sells force sensors called Force Sensing Resistors which transduce force into resistance. The force sensing pads are available for a few dollars for small ones, and about $50 for the large ones.

- **Stereo Sound.** Focal Point is a system for the Macintosh that can produce directional stereo (ie, binaural) sound. The complete package, with all the software and hardware needed costs about $1290.

- **Programmable Tactile Array.** The TiNi Alloy Company develops these for use in machine-human interfaces. Feedback is provided using mechanical simulation. A fully working demonstration package is available from Mondo-tronics for $149, and includes a single tactor (which can be mounted in a glove, for example), control box and software for the IBM PC. The interface is through a serial port.

7 Proposed Research

We would like to experiment with both hardware and software paradigms. In the best situation, we would develop a hardware user-interface using off-the-shelf components and experiment with various interaction methods for that specific interface. The quality of the interface would be judged using our criteria above.

7.1 Accessible Hardware

Access to the following three-dimensional hardware is available:

- **Mattel Powerglove.** The TransInfinity glove interface is used to connect it to a Macintosh. Demo software provided with the interface allows a user to move a virtual hand on the screen.

- **Logitech 3D mouse.** This is interfaced using the serial port. Demo software and drivers are available for the IBM PC and the Silicon Graphics workstations.

- **Citizen M329 LCD Monitors.** Two of these are available. Normal video and graphics can be output to these using the RasterOps graphics card for the Macintosh. These could be used to provide a stereoscopic display.

The total cost of the hardware is well under $3000.

7.2 Work in Progress

It was found that the resolution of the Powerglove was not sufficient to provide a good interface for fine interaction with a data set.

Therefore as part of this research effort the Logitech mouse was interfaced to an available Decstation and the software drivers were ported to it. Then some basic three-dimensional library routines were written. Finally, using the drivers and the three-dimensional library a rudimentary program was written that allows use of the mouse to position an object in space. The resolution was found to be sufficient, albeit a little low. However, the ease of positioning the dataset is unrivaled.

The next stage of work would involve writing a better interface, using some of the techniques that have already been found to be effective by other researchers. Since we also have access to a Silicon Graphics Indigo, it might be possible to use the GL library to provide the 3D graphics, which would provide much better performance.

References

[1] Grady, Sue. Here Comes the Visualization Workstation. *Computer Graphics Review.* Volume 4, No 3. March 1989.

[2] *IDL Reference Manual.* Research Systems Inc. Boulder, CO.

[3] *Dec AVS User's Guide.*

[4] Ware, C. and Jessome, D.R. Using the Bat: A Six-Dimensional Mouse for Object Placement. *IEEE Computer Graphics and Applications.* Volume 8, No 6. November 1988. pp. 65-70.

[5] Sachs, E., Roberts, A., Stoops, D. 3-Draw: A Tool for Designing 3D Shapes. *IEEE Computer Graphics and Applications.* Volume 11, No. 6. November 1991. pp. 18-26

[6] McWhorter, S., Hodges, L., Rodriguez, W. Evaluation of Display Parameters Affecting User Performance of an Interactive Task in a Virtual Environment. *Tech Report.* Graphics, Visualization and Usability Center. Georgia Institute of Technology.

[7] Eichenlaub, J. and Martens, A. 3D without glasses. *Information Display.* Vol. 8, No. 3, March 1992. pp. 9-12.

[8] Arielle Emmett. In Search of the Miracle Hologram. *Computer Graphics World.* Vol. 15, No. 7, February 1991. pp. 44-52.

[9] Kenneth and Debby Haines. Computer Graphics for Holography. *IEEE Computer Graphics and Applications.* Volume 12, No. 1. January 1992. pp. 37-46.

[10] Wall Street Journal item on holography.

[11] Bryson, S. and Levit, C. The Virtual Wind Tunnel. *IEEE Computer Graphics and Applications.* July 1992, pp. 25-34.

[12] Randy Pausch. Virtual Reality of Five Dollars a Day. *Proceedings of the ACM SIGCHI Human Factors in Computer Systems Conference.* April 1991.

[13] Stephanie Houde. Iterative Design of an Interface for Easy 3-D Direct Manipulation. *Proc. of the ACM SIGCHI Human Factors in Computer Systems Conference.* May 1992. pp. 135-142.

[14] Mercurio, P.J. and Erickson, T.D. Interactive Scientific Visualization: An Assessment of a Virtual Reality System. *Proc. of the IFIP, Third International Conference on Human-Computer Interfaces.* August 1990. pp. 741-745.

[15] Mackinlay, J.D., Card, S.K., Robertson, George G. Rapid Controlled Movement Through a Virtual 3D Workspace. *Computer Graphics.* Volume 24, No. 4, August 1990. pp. 171-176.

[16] Hiroo Iwata. Artificial Reality with Force-feedback: Development of Desktop Virtual Space with Compact Master Manipulator. *Computer Graphics.* Volume 24, No. 4. August 1990. pp. 165-170.

[17] S.Sydney Fels and Geoffrey E. Hinton. Building Adaptive Interfaces with Neural Networks: The Glove-Talk Pilot Study. *Proc. of the IFIP Third International Conference on Human-Computer Interaction.* August 1990, pp. 683-688.

8 APPENDIX A: Hardware Manufacturers

3DTV Corp.
P.o. Box Q
San Rafael, CA 94913-4316
Voice: (415) 479-3516
FAX: (415) 479-3316

Abrams/Gentile Entertainment, Inc.
244 West 54th Street, 9th Floor
New York, NY, 10019
Voice: (212) 757-0700

Dimension Technologies, Inc.
Rochester, New York.

Fake Space Labs
935 Hamilton Avenue
Menlo Park, CA 94025
Voice: (415) 688-1940
FAX: (415) 688-1949

Focal Point
318 W. Galer Street
Seattle, WA 98119
Voice: (206) 286-8922

Haitex Resources
P.O. Box20609
Charleston, SC 29413-0609
Voice: (803) 881-7518

Interlink Electronics
P.O. Box 40760
Santa Barbara, CA 93103
Voice: (805) 684-2100
FAX: (805) 684-8282

Mondo-Tronics
2476 Verna Court
San Leandro, CA 94577
Voice: (510) 351-5930
FAX: (510) 351-6955

Polhemus, Inc.
P.O. Box 560
1 Hercules Drive
Colchester, VT 05546
Voice: (802) 655-3159
FAX: (802) 655-1439

Reflection Technology
240 Bear Hill Road
Waltham, MA 02154
Voice: (617) 890-5905
FAX: (617) 890-5918

Sega America
3375 Arden Rd.
Hayward, CA 94545
Voice: (800) USA-SEGA

Spaceball Technologies
600 Suffolk Street
Lowell, MA 01854
Voice: (508)970-0330
FAX: (508)970-0199

TiNi Alloy Company
1621 Neptune Drive
San Leandro, CA 94577

Transfinite Systems Company, Inc.
Post Office Box N
MIT Branch Post Office
Cambridge, Massachusetts 02139-0903
Voice: (617) 969-9570

VPL Research Inc.
950 Tower Lane
14th Floor
Foster City, CA 94404
Voice: (415) 312-0200
FAX: (415) 361-1845 (?)