An Environment for Telecollaborative Data Exploration

Gudrun J. Klinker

Digital Equipment Corporation, Cambridge Research Lab
One Kendall Square, Bldg. 700, Cambridge, MA 02139

Abstract

This paper presents an environment for telecollaborative data exploration. It provides the following capabilities essential to data exploration: (1) Users can probe the data, defining regions of interest with arbitrary shapes. (2) The selected data can be transformed and displayed in many different ways. (3) Linked cursors can be established between several windows showing data sets with arbitrary relationships. (4) Data can be displayed on any screen across a computer network, allowing for telecollaboration arrangements with linked cursors around the world. (5) Our system is user-extensible, allowing programmers to change any component of it while keeping the remaining functionality. We demonstrate how the system can be used in several applications, such as biomedical imaging, robotics, and wood classification.

1 Introduction

Several systems for visualizing data have recently been developed with the intent to free users from the burden of graphics programming [7, 8, 13, 15, 16, 17, 18, 19, 21, 22, 23, 24, 27, 28]. Why do many application programmers still prefer writing their own visualization code over using such visualization systems? In part, this may be the case because applications such as biomedical imaging, robotics, seismic data analysis, and wood classification require highly interactive user-customizable tools to explore and interpret their data [12].

Stand-alone systems, such as EXVIS [7], KBVision [28] and PVWave Point & Click [15], have provided various data exploration tools. Yet, such systems become hard to adapt to new aspects of visualization research, due to their monolithic structure. To provide flexibility, other systems, like AVS [27], offer visual programming interfaces with which users can assemble their own networks of communicating programs (*modules*). Despite such flexibility, these new systems have until recently not been well suited for data exploration because they have lacked essential interactive capabilities. For example, the X-based [25] AVS module to display images used to encapsulate all user interaction within the module, providing no feedback to other modules. Application programmers could not exploit the full interactive power of X in their own AVS modules[1]. This missing feedback channel was crucial because users need to probe the data and define regions of interest. They also need to see the selected areas in relationship with other data, such as in a collection of CT scans shown side-by-side on a screen. Interactivity involving several windows could not be achieved easily using the programming style of AVS.

Current systems also dilute the distinction between two concepts, data representation (storage) and data presentation (display). Users often need to view the same data in many different presentations. It is important that users be able to define mappings between selected dimensions of a data set and a set of available display capabilities [2, 9, 13, 15]. Systems must also be able to mix several data types in a single view, such as geometric data and array data.

[1] Improved interactive capabilities in the newest release of AVS (Version 5) are based on the concepts presented in this paper.

This paper describes a system designed to help users analyze data interactively. It uses the network editor and the data flow model of AVS-V4. But it changes AVS-V4 in most other respects. Our system provides tools to merge data of different data types into a single view, overlaying geometric data on images. Its new X-based *display data* module also provides a feedback channel to make user interaction available through an output port to other modules. All our modules send a *log record* along with the data in which they record coordinate transformations, as well as other descriptive information about the data. Other modules can then use the *log record* to relate a selected window position to the pixel position in the original data set. With this mechanism, we can attach arbitrary data exploration and data interpretation routines to windows and establish linked cursors between any number of windows. We can establish very flexible telecollaboration arrangements by sending windows to other displays. Colleagues can then annotate the same data at both displays.

The next three chapters present three different aspects of our approach: a) the distinction between data representation and data presentation, b) data exploration tools, and c) telecollaboration capabilities.

2 Data representation versus data presentation

There are many ways to view a data set. Typically, images are shown as images, histograms as graphs, etc. This practice dilutes the distinction between two concepts, data representation (storage) and data presentation (display). Data exploration applications often need to see the same data in different presentation styles, depending on the exploration focus. Users need to define mappings between specified dimensions of data sets and sets of available display capabilities [2, 9, 13, 15]. Stand-alone systems, such as EXVIS [7], have explored various presentation styles like icons or sound generation. But the systems are hard to adapt to new aspects of visualization research, due to their monolithic structure. In contrast, our data presentation tools are available as separate modules which can be easily exchanged. Geometric and intensity-based presentations can be combined into a single view.

2.1 Presentation styles

We provide the following presentation styles:

- Intensity- and color-based displays: As an example, Figure 1 shows a color image of a scene with several plastic objects [11].
- Graphs and surface plots: Figure 2 shows surface plots of the yellow and blue rings from Figure 1 (white rectangle).
- Icons such as vectors, face drawings[6], and stick figures[7] have been used successfully in several applications.
- Printed numbers: The individual pixel values of an image (or subimage) are shown as a matrix of numbers.
- Overlays: Coherence between several presentation styles can be established by overlaying them in a single window. Figure 3 shows normalized color vector icons and printed numbers overlaid on the image data from the yellow and blue rings in Figure 1. Figure 4 shows the surface plots of Figure 2 in combination with a slanted view of the image data.
- Interleaving: Several images can be overlaid in a single window by interleaving the data pixel-by-pixel.
- Time: Movie loops and blink comparators [5] exploit motion parallax in the time dimension to establish visual coherence between related pixels in several images.
- Audio: Acoustic data presentation has found its way into visualization environments [3, 10, 26]. Current studies towards including such capabilities into our system are in progress[20].

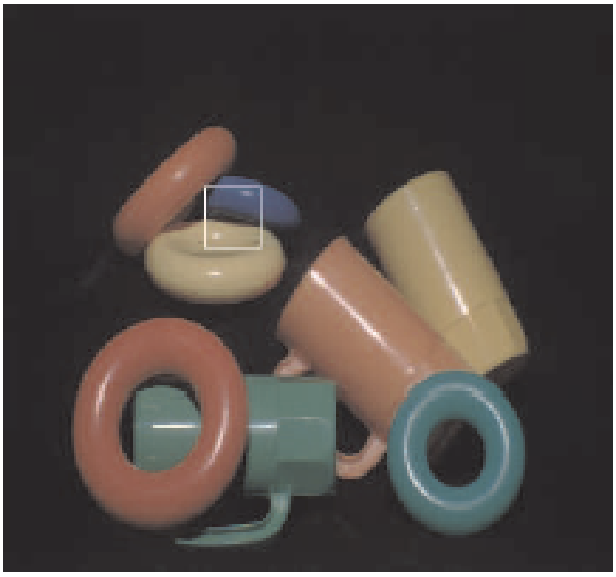

Figure 1: Color image of scene with plastic objects.

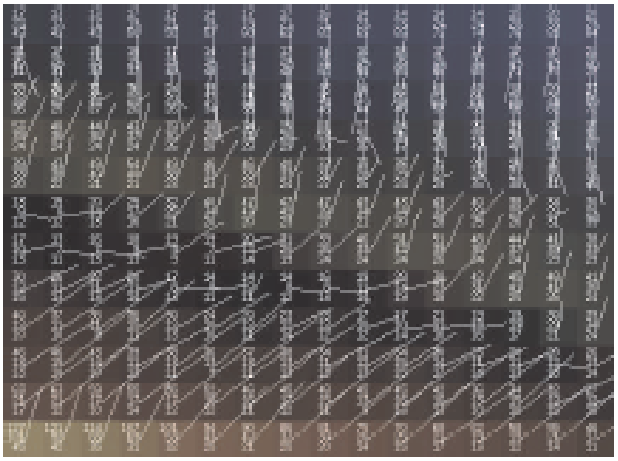

Figure 3: Numbers and vectors overlaid on the image data.

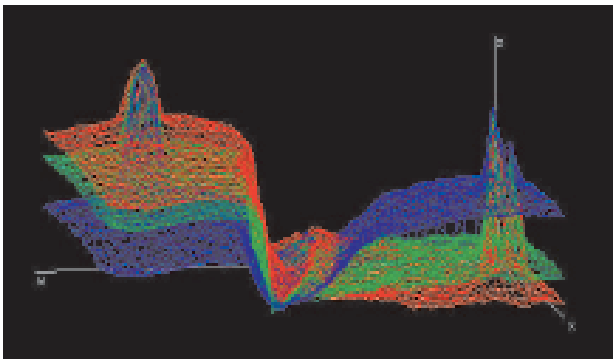

Figure 2: Surface plot of color variation in plastic scene.

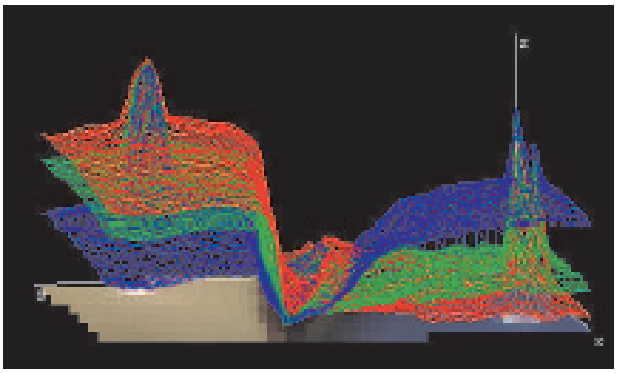

Figure 4: Surface plot overlaid on a slanted view of the image data.

2.2 Data mapping

When the data is presented in a particular style, it typically needs to be adapted to the limited capabilities of the hardware. In particular, unbounded or high-precision data values need to be adjusted to the specified range of the display device. Such mapping can take various forms: linear or non-linear scaling, subsampling or interpolation, cropping or thresholding with a specified maximum and minimum, wrapping modulo a specified value, substitutions according to a user-defined lookup table, random substitutions, permutations, and combinations of all of these techniques. Many such concepts are well-established colormapping techniques. Yet, the concepts can apply equally well to other dimensions of a data set, providing concepts, such as thresholding (cropping) and non-linear scaling (warping) to geometric dimensions or time.

3 Data exploration

Many applications, such as biomedical imaging, need to explore their data, not merely view it. They need interactive data probing tools and mechanisms to establish visual relationships between different parts of the data.

3.1 Interactive data probing

We have extended AVS by providing a *feedback channel* through which users can send arbitrary cursor and keyboard commands from the display windows back into the AVS network – where the commands are then available to any suitable module. With this extension, users can interactively set up many different data probing arrangements. We will now present several example data exploration applications which use the feedback channel. It is important to note that – although the examples do not provide new capabilities but that we provide an improved infrastructure in which it is easy to interactively arrange for such data exploration functionality and to customize it.

Figure 5 shows a very basic data probing arrangement on a CT image of a human chest. The setup

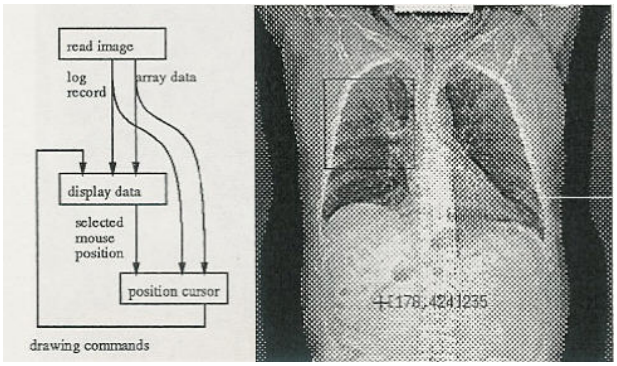

Figure 5: Basic example of a data exploration network.

consists of three modules which read the chest image, display it, and overlay a software-cursor. When a user selects a pixel in the image, the *display data* module sends the position as part of a *log record* through the feedback channel to the *position cursor* module. Since any number of modules could, in principle, have preceded the *position cursor* module and altered the geometric arrangement of pixels in the window (e.g., via zooming, warping, scrolling, projections, etc),the selected window position is not necessarily identical to the array index of the pixel in the image. The *position cursor* module uses the log record to transform the window position into the correct array index. It then sends geometric drawing commands back to the *display data* module to overlay a cursor on the image data.

3.2 Regions of interest

The data probing arrangement can be extended to define a region of interest, as shown in Figure 6. The *crop image* module selects a subarea from the chest image, using the cursor position and the *log record* to determine the subarea (the black rectangle in Figure 5). A second *display data* module shows the cropped area in a second window. Interactive window resizing allows users to zoom the view. Statistical data analysis, such as the computation of mean and variance or other moments, can also be performed on the selected region of interest.

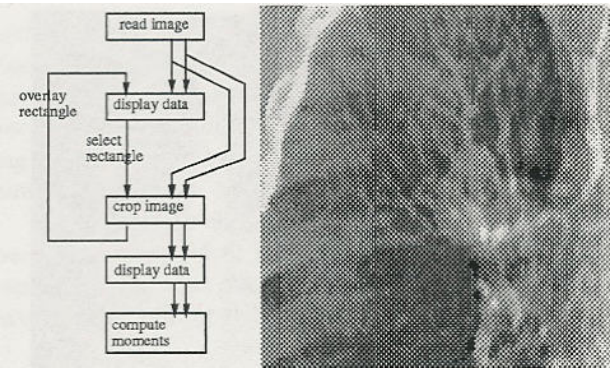

Figure 6: Network to select an enlarged region of interest from the chest image and compute its mean and variance.

3.3 Cursor linking

We provide cursor linking capabilities to help users explore the relationship between several data sets. In a typical cursor-linking arrangement, several windows each have their own software cursor to indicate the current cursor position. The cursor can be repositioned in any window, affecting the cursors in all windows. *Live links* for visualizing the relationship between several data sets are available in some current visualization systems [15, 28]. Yet, those are closed systems without a visual programming interface. They provide only limited capabilities which cannot be extended easily by the user.

In our system, users can configure arbitrary cursor-linking arrangements of varying complexity. In a simple case, linked cursors show corresponding pixels in images of identical dimensions and size, such as two CT images. In a slightly more complicated case, the cursors are linked between images of different sizes, such as between different levels of an image pyramid or between an image and a cropped and zoomed subimage. The cursor-linking mechanism then has to multiply the pixel index with the appropriate scale factor or add the cropping offset. In yet more complicated cases, the linking mechanism can be used to help visualize the relationship between data sets of different dimensions, such as an image and its histogram.

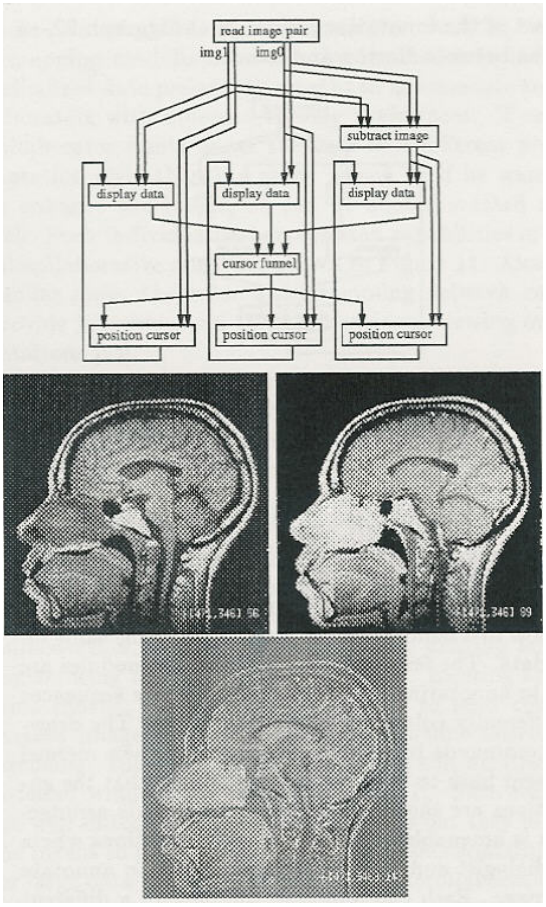

Figure 7: Network to view pre- and post-contrast MR-images and their difference image with linked cursors.

Such cursor linking arrangements will now be presented in detail. The demonstrations are examples of a very general concept. Any two data sets with a known semantic relationship can be linked by inserting the appropriate transformation algorithm into the cursor linking arrangement. Further examples could include cursor transformation mechanisms for non-rigid image warping, fish-eye views, the histogrammed analysis of some derived image properties (such as edges, texture, and flow vectors), or the joint visualization of CT and MR data.

3.3.1 Cursor linking between several images

Figure 7 shows two registered pre- and post-contrast MR images of a human head. A third image shows the pixel-wise differences. The bright pixels indicate areas which the solution has penetrated. On the screen, the images are shown in three windows with linked cursors: When a user identifies an interesting position in one of the windows, the corresponding pixel positions are marked in all three windows. The network in Figure 7 shows that this functionality is achieved by multiplexing the cursor positioning output from all three *display data* modules in the network through a *cursor funnel* and then sending it on to three *position cursor* modules. Each *position cursor* module creates geometric drawing commands which are then sent back to the respective *display data* modules. The *log* mechanism ensures correct cursor positioning even if different windows show the data at different scale factors.

The same cursor-linking mechanism can be used to explore corresponding pixels on several levels of an image pyramid in computer vision applications. The lowest level of the pyramid shows the image at full resolution, higher levels of the pyramid show the data at increasingly lower sampling rates.

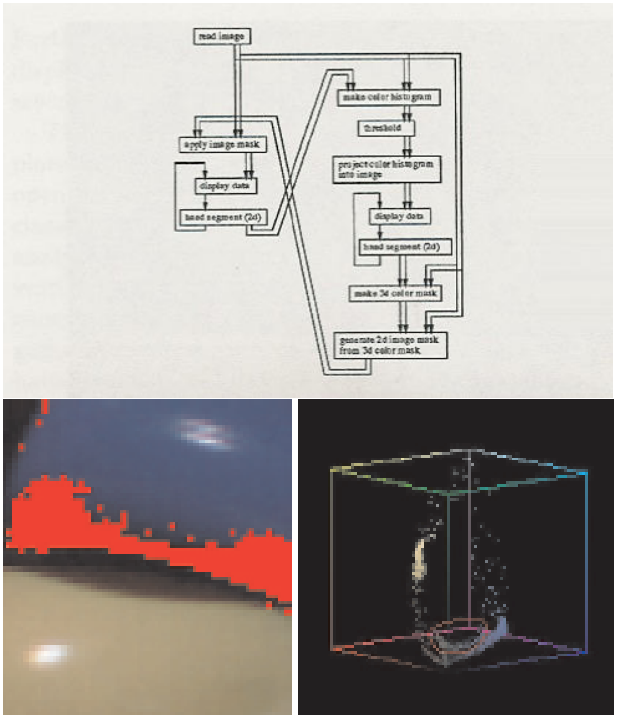

Figure 8: Linked cursors for the plastic scene and its color histogram.

3.3.2 Cursor linking between images and histograms

The cursor-linking mechanism can also be used to establish relationships between windows with very different types of dimensions, such as an image and its histogram. In this case, selected cursor positions flow through special transformation modules which establish the correct relationships between image pixels and histogram counts, i.e, they use the data value of a selected image pixel as the index into the histogram.

We use the cursor linking mechanism to explore the relationship between color images and color histograms (three-dimensional scatter plots). A color histogram consists of four dimensions, $(r, g, b, count)$. It is stored as a volume, with each voxel (r, g, b) indicating how many pixels in the original color image have this particular color value. The color histogram is presented, using a z-buffering algorithm and ignoring all voxels with *counts* below a threshold. The voxels are displayed either as intensity data according to their *count*-value, or as color triples (r, g, b) according to their position in the histogram.

In Figure 8, the blue and yellow rings from Figure 1 and its color-encoded histogram are shown side-by-side. The left branch of the network above shows the modules which process and display the color image. The modules in the right branch operate on the histogram. Users can outline an area in the color image with the module *handsegment*. The system then generates a new color histogram, using only the pixels in the outlined area. Conversely, users can also outline an area of the color histogram. In that case, the system marks all image pixels which have colors in the selected part of the histogram. In Figure 8, a small area, outlined in red, has been selected in the color histogram. The resulting image pixels are marked in red, indicating an interreflection area between the

two rings.

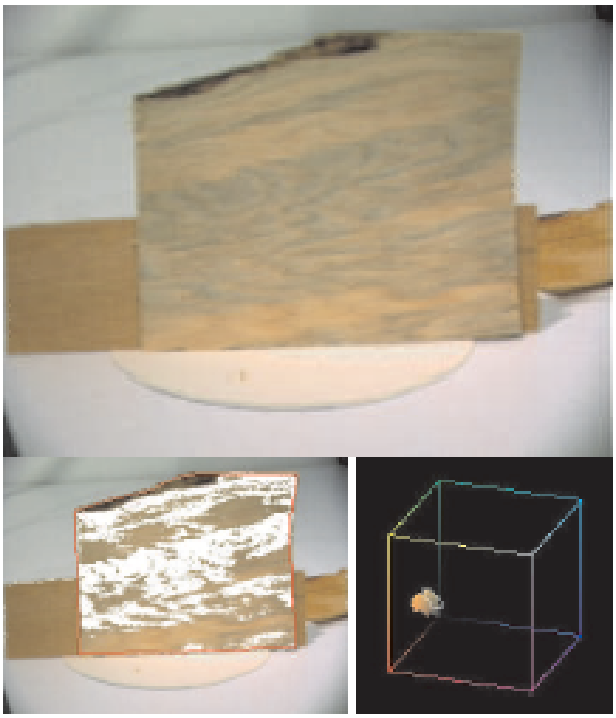

Figure 9: Color histogram analysis for wood samples.

Color histograms can be used in a similar way to develop automatic wood inspection algorithms [4]. In such research, various impurities, such as knots and stains, need to be detected and classified. They typically cause color shifts in the wood, yet they are hard to recognize automatically. Figure 9 shows a wood sample with a bluish wood stain. The bluish colors are hand-selected in the color histogram. The corresponding pixels in the lower left image identify a significant portion of the stained area. With this interactive viewing arrangement, researchers can investigate the characteristic color clustering properties for different wood samples before they design appropriate automatic wood inspection algorithms.

The semantics of any relationship between data sets can be encoded in transformation modules. Extensions to higher-dimensional data sets, such as data bases of census data or financial data, can be created.

4 Telecollaboration

Telecollaboration is integrated in our data exploration environment. All capabilities available to one user are also available to two or more users working on the same problem. This approach stands in contrast to current practice where users have to switch back and forth between exploring data in one system and sharing the results in a different system.

We can provide integrated telecollaborative capabilities because we combine cursor linking with a window migration capability. Users can send any window to any screen simply by specifying a *display name* parameter. We have successfully demonstrated telecollaborative arrangements between Chicago and Boston as part of the Innovation Showcase at Siggraph 92, as well as between Boston and Sweden.

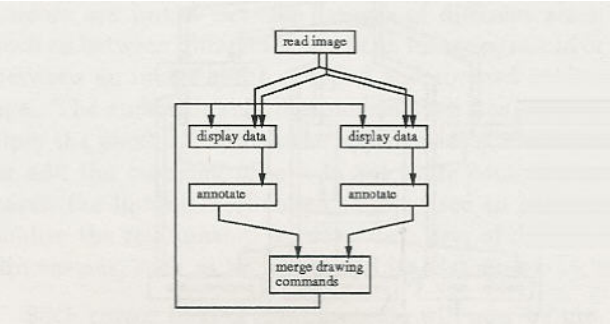

Figure 10: Basic tele-collaboration network.

Our system gives users the flexibility to set up collaboration environments with various degrees of complexity. In a simple case, collaborators share one window, with a mechanism to allow each person to independently annotate the data, as shown in Figure 10: Two *display data* modules – one of which may send the window to a remote display – each receive the same input data. The feedback channels of both modules are sent to annotating modules which generate sequences of differently colored drawing commands. The drawing commands from both windows are then merged and sent back to both display modules so that the annotations are shown in both windows. This arrangement is amenable to teleradiology applications where a radiologist and a surgeon jointly

want to annotate an image. Each can annotate the data in a different color, both see the results.

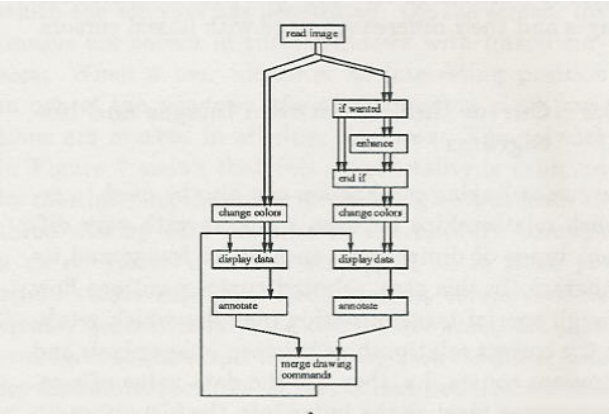

Figure 11: Tele-collaboration network with individually customizable views.

Users can also share more complicated exploration arrangements, such as the joint analysis of the pre- and post-contrast MR images of Figure 7. In the process, collaboration partners can individually customize their views, e.g, by resizing or repositioning their own windows. Specific modules, such as colormapping modules, can also be inserted into individualized data presentation paths to accomodate collaborators with different viewing preferences. If one collaborator wants to see the data in a different presentation style than the other person or if he wants to enhance the image, he can be accommodated as well. Such individualized exploration capabilities in a telecollaborative setup are shown in Figure 11. Along similar lines, the color histogramming network can provide individualized three-dimensional viewing orientations [14].

We expect these telecollaborative capabilities to be an important extension to data exploration. Many data exploration tasks in real applications are group efforts rather than problems solved by an individual. There will be an increasing demand for systems that allow researchers to share their views and truly explore them with their colleagues. The current situation where researchers explore data in one system and share it in another one is an obstacle to remote collaboration. Our system can enhance existing teleconferencing technology in this respect. Existing systems provide sophisticated techniques to send compressed image data and sound for real-time teleconferencing. Tools also exist to share and edit arbitrary X-based windows on several displays [1]. Such tools are well suited for sharing text windows but they lack the means to provide in-depth exploration capabilities for large data sets. We expect future telecollaboration systems to combine traditional window sharing and teleconferencing arrangements with data exploration systems to analyze large data sets.

5 Conclusions

Data exploration imposes special requirements on visualization environments. It requires a high degree of interactivity which has been ignored by current visualization systems. Our system is designed to provide interactive data exploration capabilities. Built on top of AVS, it benefits from the visual programming paradigm that AVS provides. By making feedback from the user directly available to all modules and by introducing a *log* scheme between modules, we have been able to change AVS from a pure visualization system into a data exploration system. In this new system, users are able to interactively select pixels in windows, perform arbitrary operations on them, and redisplay them in many different ways. Selected pixels can be cross-linked between several windows with different data sets and sent around the world for telecollaboration arrangements between remote colleagues. Furthermore, we provide many different techniques for displaying data, and for mixing and merging data from several data sets into a single view.

These capabilities are essential to many data exploration applications. We thus expect our system to open up scientific visualization environments to new classes of applications. It has already been used to analyze color variation on wood samples, and to investigate reflection properties on plastic objects for computer vision research. It is an integral part of ongoing biomedical research at our research lab, and we have demonstrated its telecollaboration capabilities on several occasions.

Acknowledgements

The development of this system has been influenced by many people. I am especially grateful to Ingrid Carlbom and the Visualization Group at CRL for their help and many discussions on visualization environments. The data shown in this paper has been collected from various sources. The plastic scene comes from the Calibrated Imaging Lab of the Robotics Institute at Carnegie Mellon University. The CT scan of a human chest and the pre- and post-contrast MRI scans of a head are from the Mallinckrodt Institute of Radiology. The wood sample is from Department of Forest Products at Oregon State University.

References

[1] Michael Altenhofen, Burkhard Neidecker-Lutz, and Paul Tallett. Upgrading a window system for tutoring functions. In *European X Window System Conference and Exhibition (EX'90)*, November 1990.

[2] R.A. Becker and W.S. Cleveland. Viewing multivariate scattered data. *PIXEL*, 2(2):36–41, 1991.

[3] S. Bly. *Sound and Computer Information Presentation.* PhD thesis, Computing Science Group, University of California, Davis, Lawrence Livermore National Laboratory, March 1982.

[4] C.C. Brunner, G.B. Shaw, D.A. Butler, and J.W. Funck. Using color in machine vision systems for wood processing. *Wood and Fiber Science*, 22(4), 1990.

[5] I. Carlbom, D. Terzopoulos, and K.M. Harris. Reconstructing and visualizing models of neuronal dendrites. In *Proc. of CG International '91: Visualization of Physical Phenomena, Boston, MA*, Tokyo, Japan, June 24-28 1991. Springer-Verlag.

[6] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68(342):361, 1973.

[7] G. Grinstein, R.M. Pickett, and M.G.Williams. EXVIS: An explanatory visualization environment. In *Graphics Interface*, pages 254–261, London, Ontario, Canada, June 1989.

[8] A.J. Hanson and L.H. Quam. Overview of the SRI Cartographic Modeling Environment. In L.S. Bauman, editor, *DARPA-Image Understanding (IUS) workshop*, pages 576–582. Morgan Kaufmann, April 1988.

[9] W. Hibbard, C.R. Dyer, and B. Paul. Display of scientific data structures for algorithm visualization. In *Proc. of Visualization '92*, pages 139–146, Boston, MA, October 1992. IEEE Computer Society Press.

[10] B. Kaplan. Sonification in AVS. In *AVS '93*, Walt Disney World, Lake Buena Vista, FL, May 24-26 1993.

[11] G.J. Klinker. *A Physical Approach to Color Image Understanding.* PhD thesis, Computer Science Department, Carnegie-Mellon University, May 1988. Available as technical report CMU-CS-88-161.

[12] G.J. Klinker. We need interactive data interpretation rather than interactive data visualization. Workshop on Scientific Visualization, Visualization '91, San Diego, October, 1991.

[13] G.J. Klinker. VDI – a Visual Debugging Interface for image interpretation and other applications. In F. H. Post and A.J.S. Hin, editors, *Advances in Scientific Visualization.* Springer Verlag, Berlin, Heidelberg, New York, 1992.

[14] G.J. Klinker. Interactive data exploration and telecollaboration in biomedicine using AVS. In *Proc. of the 2nd Int. AVS User Group Conference*, Walt Disney World Dolphin, FL, May 24-26 1993.

[15] R.D. Kriz. PV-Wave point and click. *PIXEL*, 2(2):28–30, 1991.

[16] B. Lucas, G.D. Abram, D.A. Epstein, D.L Gresh, and K.P. McAuliffe. An architecture for a scientific visualization system. In *Proc. of Visualization '92*, pages 107–114, Boston, MA, October 1992. IEEE Computer Society Press.

[17] C.C. McConnell and D.T. Lawton. IU software environments. In L.S. Bauman, editor, *DARPA-Image Understanding (IUS) workshop*, pages 666–677. Morgan Kaufmann, April 1988.

[18] P.J. Mercurio. The data visualizer. *PIXEL*, 2(2):31–35, 1991.

[19] P.J. Mercurio. Khoros. *PIXEL*, 3(1):28–33, 1992.

[20] J. Morse. Using a audio module within the EDI framework. Private communication, 1992.

[21] J. Mundy, T. Binford, T. Boult, A. Hanson, R. Beveridge, R. Haralick, V. Ramesh, C. Kohl, D. Lawton, D. Morgan, K. Price, and T. Strat. The image understanding environment program. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'92)*, pages 406–416, 1992.

[22] Department of Computer and Electrical Engineering of the University of New Mexico. Xvision 3.0. Also known as the Visualization Workbench from Paragon Imaging, 1989.

[23] L. Quam. The Image Calc vision system. Technical report, Stanford Research Institute, Menlo Park, CA, 1984.

[24] K. Riley and C. McConnell. Powervision. Technical report, Advanced Decision Systems, Mountain View, CA, March 1988.

[25] R.W. Scheifler and J. Gettys. The X window system. *ACM Trans. Graphics*, 5(2):79–109, April 1986.

[26] S. Smith and M.G. Williams. The use of sound in an exploratory visualization environment. Technical Report R-89-002, Department of Computer Science, University of Lowell, Lowell, MA 01854, May 1989.

[27] C. Upson, T. Faulhaber Jr., D. Kamins, D. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, and A. van Dam. The Application Visualization System: A computational environment for scientific visualization. *IEEE Computer Graphics and Applications*, 9(4):30–42, July 1989.

[28] T.D. Williams. Image understanding tools. In *IEEE 10th International Conference on Pattern Recognition (ICPR'90)*, pages 606–610, Atlantic City, NJ, June 1990. IEEE.