N94- 35046

RAVE: Rapid Visualization Environment

D. M. Klumpar
Lockheed Space Sciences Laboratory
3251 Hanover Street
Palo Alto, CA 94304
klump@agena.space.lockheed.com

Kevin Anderson and Evangelos Simoudis
Lockheed AI Center
3251 Hanover Street
Palo Alto, CA 94304
{kevin, simoudis}@aic.lockheed.com

Abstract

Visualization is used in the process of analyzing large, multidimensional data sets. However, the selection and creation of visualizations that are appropriate for the characteristics of a particular data set and the satisfaction of the analyst's goals is difficult. This process consists of three tasks: generate, test, and refine, that are performed iteratively. The performance of these tasks requires the utilization of several types of domain knowledge that data analysts do not often have. Existing visualization systems and frameworks do not adequately support the performance of these tasks. In this paper we present the RApid Visualization Environment (*RAVE*), a knowledge-based system that interfaces with commercial visualization frameworks and assists a data analyst in quickly and easily generating, testing, and refining visualizations. *RAVE* has been used for the visualization of *in situ* measurement data captured by spacecraft.

1. Introduction

Large volumes of multidimensional data are routinely collected in fields that range from space physics to retail marketing. Information is extracted from the collected data through visualization techniques, as well as a variety of analysis methods, e.g., statistical, whose results are comprehended also using visualizations. The creation of useful visualizations is a knowledge intensive task that is usually performed as a **generate and test** process. The analyst must know (1) how to visualize the data set he is analyzing, (2) what visualization package to use to realize the selected visualization(s), and (3) whether the data set will need to be transformed before the selected package can generate the chosen visualization. Very few analysts possess all the necessary types of knowledge. As a result, the generate and test process takes a long time to perform. Existing data visualization systems and frameworks suffer from two limitations. Either they are not easily extensible implying that the generate and test process cannot be performed, or they are hard to use. In this paper we present the RApid Visualization Environment (*RAVE*), a knowledge-based system that interfaces with commercial visualization frameworks and assists a data analyst in quickly and easily generating, testing, and refining visualizations. *RAVE* generates visualizations that satisfy a set of analysis and display goals stated by its user. The system has been used by space scientists for the visualization of *in situ* measurement data captured by spacecraft. It currently interfaces with the PV-Wave and AVS visualization frameworks.

Once the user selects a goal, *RAVE* automatically identifies a set of visualizations that can be used to achieve the goal. The user selects one or more visualizations from this set, and *RAVE* automatically creates and executes the appropriate program to generate each of the selected visualizations. The program is implemented in the language of the visualization frameworks that are interfaced to *RAVE*. The user can compare the effectiveness of the created visualizations with regards to the amount of information that can be extracted and the way the information is presented. In this way, the analyst can quickly and easily select a set of target visualizations which can then be further refined.

2. Data Visualization

Data is collected and analyzed to create models that predict the future behavior of a system, or explain an observed event. For example, a space scientist may try to explain whether the earth's magnetosheath contain solar wind plasma or just noise. Data analysis gives rise to goals that must be achieved. Visualization can be used to achieve these goals. For example, an analyst may create a scatter plot of temperature versus density for protons in an attempt to explain the existence of solar wind in earth's magnetosheath.

Visualization of a data set to achieve a goal implies that the analyst must be able to (1) decide how to visualize a data set, (2) create the decided upon visualization, and (3) assess the created visualization's effectiveness in presenting the information contained in the data set and thus achieving the stated goal. The selection and creation operations can be viewed together as a generate operation. Therefore, visualization in this class of domains can be framed as a **generate and test** process.

Deciding how to visualize the data implies that the analyst understands the benefits of using each visualization, and knows how to map from a space of analysis goals to a space of visualizations. Oftentimes these mappings are one-to-many further complicating the visualization-selection task. Creating a visualization, and later refining it through the addition of features such as color, implies that the analyst must have knowledge about computer graphics and programming. For example, the analyst must know how to write a program to generate a scatter plot, and then how to use color to display the data points whose values are greater than some threshold. Existing visualization languages that are included in frameworks such as AVS, PV-WAVE facilitate the programming task but have steep learning curves. The knowledge included in the *RAVE* system (1) supports the data analyst during the visualization selection operation, and (2) enables the automatic creation of the corresponding programs in the visualization languages of the frameworks to which *RAVE* is interfaced.

3. Application Domain

RAVE has been used for the visualization of space sciences data from *in situ* measurements of plasmas, fields, and corpuscular radiation. For example a set of instruments for *in situ* measurements may capture the DC magnetic field (a three-component field vector), the DC electric field (also a three component field vector), the flux of particles as a function of their energy of arrival, charge, and mass composition, and the AC electric and magnetic wave field spectra (energy density vs. frequency). We have experimented with a subset of observations from the Hot Plasma Composition Experiment (HPCE) on the AMPTE/CCE spacecraft.

The Charge Composition Explorer (CCE) satellite of the AMPTE program was launched in August, 1984 into a near equatorial orbit (inclination 4.8° with apogee of 8.8 R_e and perigee 1108 km). The CCE was spin-stabilized at 10 rpm with its spin axis pointing approximately toward the sun. The spacecraft carried instrumentation to measure composition and charge state of ions over a very broad energy range, electrons, plasma waves, and the geomagnetic field. The data used here from the HPCE was taken by a set of eight magnetic electron spectrometers that measure the flux of electrons from 50 eV to 25 keV [Shelley et al., 1985]. Each spectrometer is operated at a fixed energy with an energy resolution of 50%. The eight instruments are co-aligned with their fields-of-view perpendicular to the spacecraft spin axis and are operated simultaneously, making measurements in unison every 155 msec. Thus an eight point electron energy spectrum is obtained every 9.5° of satellite rotation. The spectrometers are collimated with a 5° full width conical field-of-view giving an effective full width field-of-view of 5° x 14.5° during each measurement period. As the spacecraft rotates the view directions sweep through a range of pitch angles the amplitude of which depends on the direction of **B** with respect to the spacecraft spin axis. The full pitch angle scan (0°-180°) is achieved when **B** is perpendicular to the spin axis.

The captured measurements may be viewed as a succession of time slices through energy space. Each time slice contains eight measures of the instantaneous electron flux (at the eight energies) and each successive slice is displaced by 9.5° of rotation in the spin plane of the satellite. All captured parameters vary in time and space with rates-of-change that are highly variable as (1) the environment responds to externally applied forces, and (2) the spacecraft moves between different plasma regimes. In addition to the data captured from the HPCE sensors, our data set also contained data that was derived by combining data from multiple sensors. In this way we were able to represent global characteristics of the plasma, e.g., plasma beta, ratio of plasma frequency to gyro frequency, etc.

4. *RAVE*: System Description

RAVE is a knowledge-based system that supports a data analyst in generating, testing, and refining visualizations of a set of relational data. It is layered on top of existing visualization frameworks, e.g., AVS, PV-WAVE, etc., capitalizing on their data representation and rendering strengths, while improving their ease of use. *RAVE* consists of: a graphical user interface, a knowledge base, an inference engine, a facts database, a visualization program generator, and interfaces to the visualization frameworks to which it is connected. The system's architecture is show in Figure 1. *RAVE*'s graphical user interface is implemented using DevGuide under Open Look, the interfaces to the visualization frameworks are implemented in C, whereas the rest of the system is implemented in Common Lisp using the Common Lisp Object System.

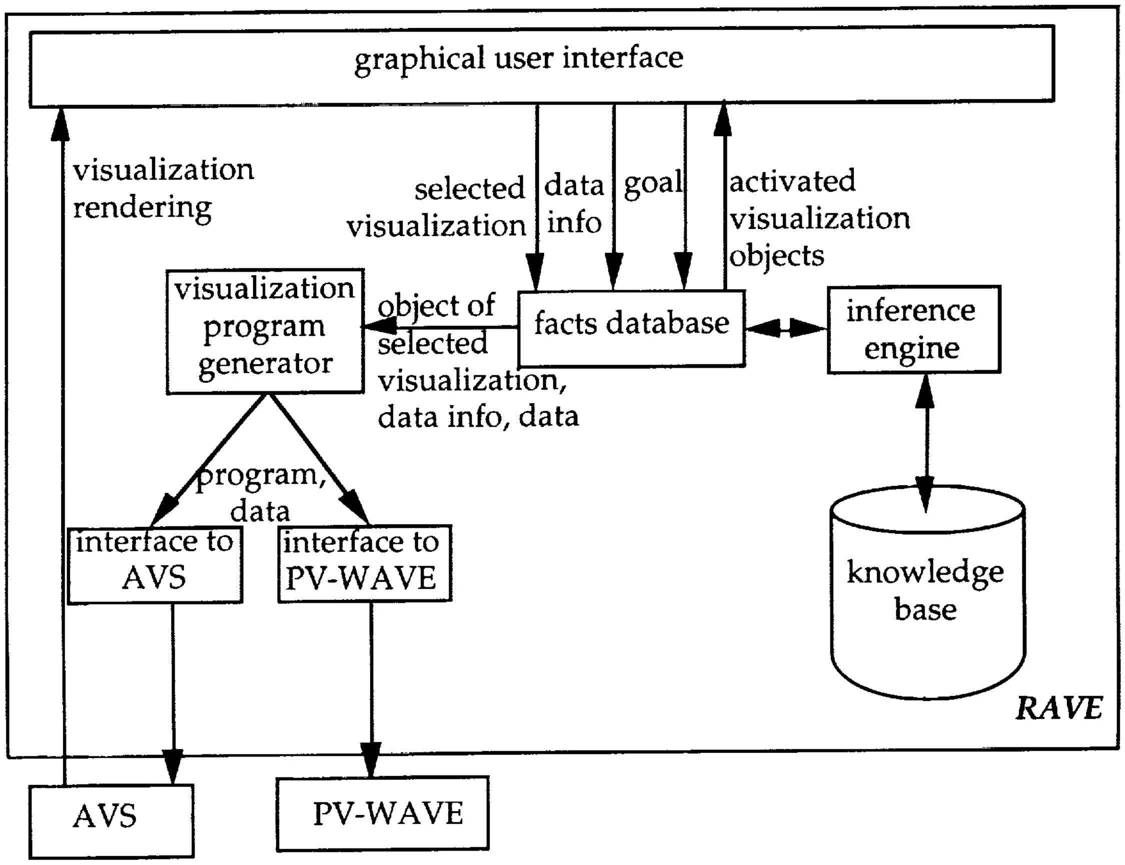


Figure 1: The architecture of the *RAVE* system

RAVE's knowledge base contains (1) a set of visualization objects, and (2) a set of rules that relate visualization goals to visualization objects. Each method for visualizing a data set is represented by a separate object. An object has slots for: (1) the name of the visualization, (2) the goals the corresponding visualization can satisfy, (3) the refinements the visualization can accept, (4) the domain(s) in which it can be used, and (5) the program that implements it in the language of one of the visualization frameworks with which *RAVE* interacts. For example, the visualization object that corresponds to the two-dimensional scatter plot can satisfy the goal "attribute x is related to attribute y," it can accept zooming and color as refinements, and can be applied in any domain where numeric-valued attributes are compared.

Occasionally, a data set may need to be transformed, e.g., to a different coordinate system, before the visualization can be rendered in a framework. The visualization-generation program included in an object includes the necessary transformations that must performed, and executes them automatically as part of the program-creation process. If a particular visualization can be rendered by more than one of the frameworks with which *RAVE* is interfaced, then a visualization object can include a separate visualization-generation program, along with the necessary transformations, for each of the appropriate frameworks. The user can select the framework(s) where the data will be displayed. We plan to expand the types of knowledge that can be represented in a visualization object by making explicit visualization-selection criteria such as cost of displaying a visualization, type of display device needed for displaying effectively a selected visualization, etc. In this way, not only we will be able to further assist the analyst's selections, but we will also be able to capture the rationale behind each selection. Finally, this knowledge will also be accessible by the rules and will enhance the *RAVE*'s mapping capabilities.

A set of visualization objects that are appropriate for the data of a particular application domain can be organized into a collection. For example, objects that correspond to visualizations used with financial data are organized into a collection. Such collections can be organized hierarchically with the top-level collection (node) containing the set of general visualizations, e.g., scatter plots, line plots, etc.

The antecedents of each rule represent (1) the statement of a goal, and (2) the constraints that need be satisfied before the goal can be achieved. The rule's consequents select the visualizations, i.e., instantiate the visualization objects, that can satisfy the goal. The rules make explicit: (1) what constitutes an appropriate visualization for a data set and a goal, (2) how to develop knowledge that can be shared across several domains, and (3) how to choose a particular visualization. A set of rules can be associated with a collection of visualization objects, e.g., the objects that correspond to visualizations that pertain to analysis of financial data. Two example rules are shown in Figure 2. Variables in these rules are preceded by question marks. Each variable, e.g., ?x, is bound to an object that contains information about each attribute in the data set to be visualized. The information includes: the type of the attribute's values, e.g., integer, real, nominal, etc., the number of distinct values included in the set to be visualized, minimum and maximum values (for numeric-valued attributes), the type of data, e.g., time-series, a pointer to the actual values, etc.

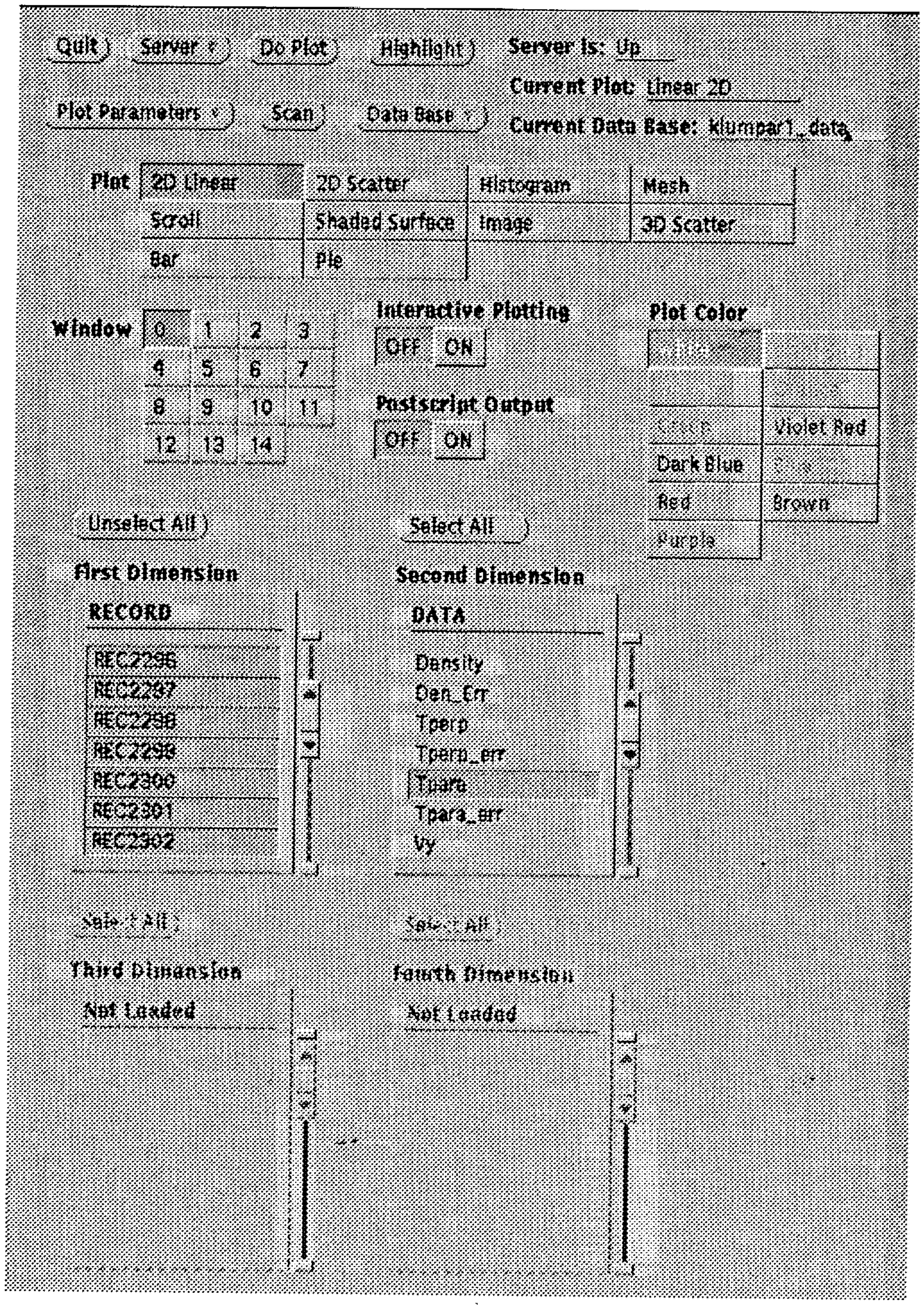
```
(if (and (goal (related-to ?x ?y))
         (equal (value-type ?x) 'numeric)
         (equal (value-type ?y) 'numeric))
 then (activate 2-d-scatter-plot ?x ?y))

(if (and (goal (value-distribution-of ?x))
          (<= (number-of-distinct-values ?x) 5))
 then ((activate 2-d-bar-graph ?x) (activate 2-d-pie-chart ?x) (activate 3-d-pie-chart ?x)))
```

Figure 2: Examples of rules used by *RAVE*

The first rule states that the visualization object corresponding to the two-dimensional scatter plot is enabled for selection by the user if the stated goal seeks to establish whether attribute x is related to attribute y, and the values of both attributes x and y are numeric. The second rule states that the visualization objects corresponding to the two-dimensional bar graph, the two-dimensional pie chart, and the three-dimensional pie chart are enabled for selection by the analyst if the stated goal seeks to identify the distribution of the values of a particular attribute and the number of distinct values this attribute takes in the selected data set is not greater than five.

The user interface initially allows the analyst to: (1) select a data set to be analyzed, (2) specify any special characteristics of the selected data, e.g., time-series data, and (3) choose a visualization goal from a menu of pre-specified goals. The selected data, i.e., attributes and values, is preprocessed so that information such as minimum and maximum values can be established. The data and the resulting metadata are organized into an object that is asserted in *RAVE*'s facts database. The selected goal is asserted as a separate fact in the same database. The inference engine executes the rules whose antecedents match facts in the database. As a result of executing the matching rules, one or more visualizations may be activated. The names of the activated visualizations are displayed to the analyst through *RAVE*'s interface. The analyst can select one or more of the activated visualizations. A portion of *RAVE*'s user interface is shown in Figure 3.

As was stated above, if a selected visualization can be displayed by more than one of the frameworks that are connected to *RAVE*, the analyst must select the frameworks that will be used. The program(s) included with the object that corresponds to each selected visualization is subsequently sent to *RAVE*'s visualization program generator which instantiates it for the specified data set and attributes. The instantiated program is then sent, through the appropriate interface, to the corresponding visualization framework for rendering. Each selected visualization is rendered in a different window. The analyst is given control, through the user interface, of simultaneously displaying all the created visualizations. In this way, the analyst is able to compare the information-extraction and presentation effectiveness of each selected visualization, and either choose the best one(s), refine one or more of the displayed visualizations, or discard them and modify his initial selections so that new visualization objects may be enabled.



Server is: Up
Current Plot: Linear 2D
Current Data Base: klumpar1_data

Figure 3: A portion of *RAVE*'s user interface

Should, after seeing the results of a selected visualization, the analyst decide to refine it, he can select one or more of the refinements supported by the particular visualization. The selected refinements are reflected on the instance of the visualization object that is asserted on *RAVE*'s facts database. The updated object is communicated to the program generator which creates an instance of the updated program(s) corresponding to the visualization. This new instance is similarly sent to the appropriate visualization framework for rendering. Figure 4 shows a two-dimensional scatter plot that has been refined by enabling the zooming and coloring capabilities.

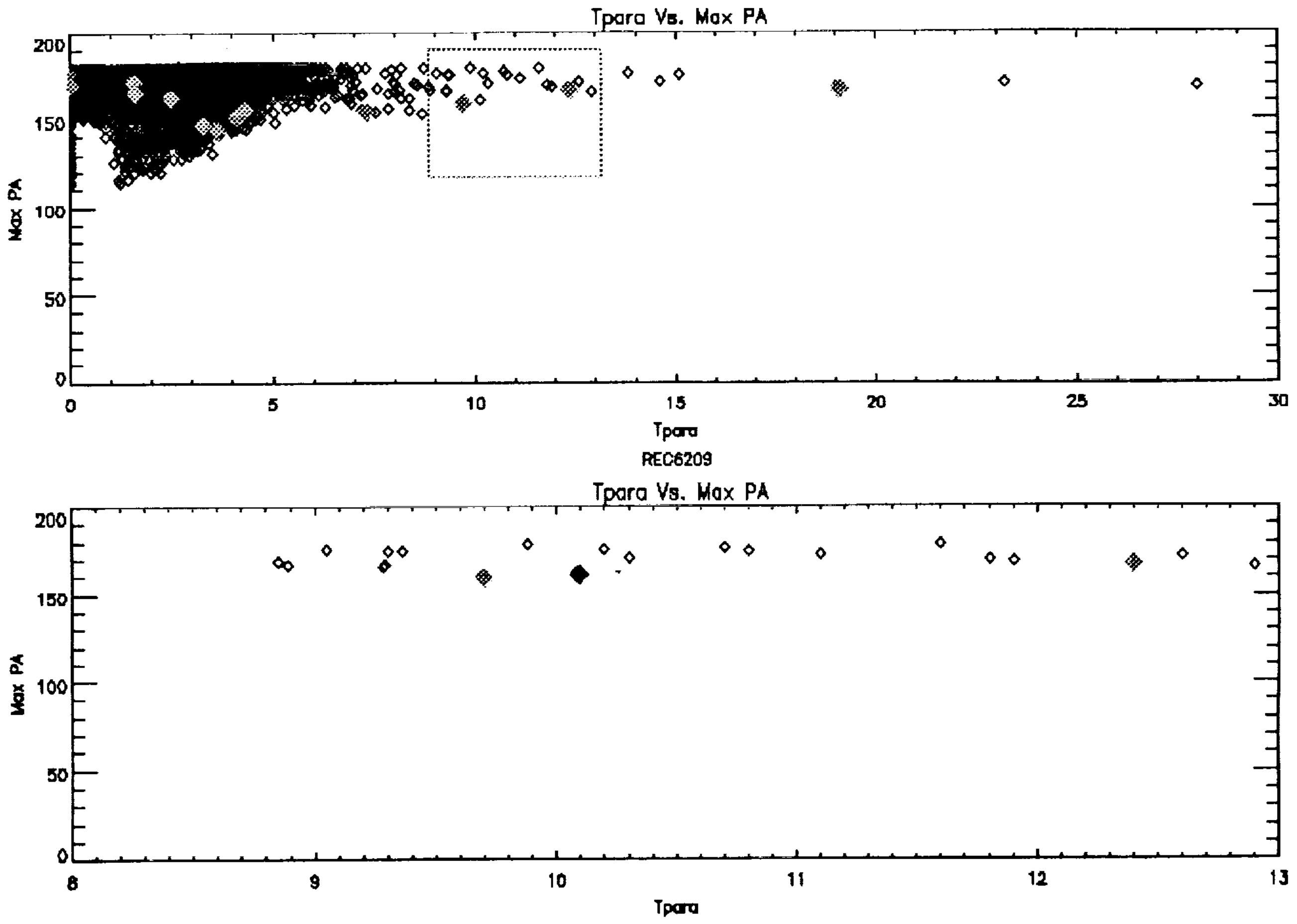


Figure 4: A two-dimensional scatter plot produced by *RAVE*

6. Related Work

We compare *RAVE* to two sets of systems. The first set consists of systems that use artificial intelligence techniques to facilitate the creation of complex user interface. In particular, we examine the Integrated Interfaces system [Arens *et al.* 91], and the APT system [Mackinlay 91]. The second set consists of two visualization frameworks that have been developed for space sciences data: LinkWinds [Jacobson & Berkin 93], and SAVS [Szuszczewicz *et al.* 92].

Integrated Interfaces is a knowledge-based system whose goal is to alleviate the work of user interface designers and developers. For this reason the system uses knowledge that allows it to automatically select, at run time, the most appropriate way for presenting information. Integrated Interfaces supports output in natural language, text, maps and other graphics, tables, and forms. Its knowledge base represents domain knowledge, and knowledge about user interfaces. In contrast, *RAVE* concentrates on output that can be presented through graphics. Its knowledge base represents deeper and more detailed knowledge both about graphics and the particular domains where the system is applied. Furthermore, through its ability to display several different visualizations that can satisfy a particular goal, *RAVE* provides its user with the ability to perform guided exploration of the visualization space by comparing the effectiveness of several visualizations in extracting the most possible information from the contents of the data set.

The goal of the APT system is to dynamically create effective visualizations of relational information by searching a space of possible designs. In this respect, it is a tool for performing open-ended exploration of the visualization space. It concentrates on bar charts, scatter plots and connected graphs. APT has the ability to decompose a data set into components that can be visualized individually, selecting a visualization appropriate for each component and then composing from the individual pieces the visualization for the overall data set. As was mentioned above, *RAVE* provides less support for a guided rather than an open-ended exploration of the visualization space. It assumes that the user will partition a data set, should the need arise. However, it supports more complex domain-specific and domain-independent visualizations of relational information than the APT system. Finally, *RAVE* is able to interface with existing visualization frameworks making it an easily extensible system able to render complex domain-specific visualizations, in addition to the generic simple ones.

The LinkWinds system provides an environment for rapidly prototyping and executing visualization applications on planetary data. It allows its user to analyze data and creation visualizations through a spreadsheet interface. *RAVE*, in contrast, provides knowledge-based support for selecting appropriate visualizations and automates the creation of each selected visualization template.

The SAVS tool provides data acquisition, manipulation, and visualization capabilities. The system is being applied on solar-terrestrial and planetary data. It is implemented on top of the AVS visualization environment. SAVS does not provide knowledge-based support during the visualization-creation process. In addition, the visualizations it supports are those that can be supported by AVS. In addition to the knowledge-based support it provides, *RAVE* can be interfaced with a variety of visualization languages, including AVS. As such it can support a wider range of visualizations than SAVS.

7. Conclusions

We continue developing the *RAVE* system expanding its knowledge base with more visualizations that are appropriate for space sciences data, and interfacing it to other visualizations frameworks, such as SGI Explorer, that have desired features. In addition, we are in the process of interfacing *RAVE* to the Recon data mining system [Simoudis *et al* 93] that provides sophisticated data management and analysis capabilities.

The process of generating, testing, and refining visualizations is particularly well-suited for the space sciences domain, as well as for other domains where graphics are used for extracting information from data but where the identification of the appropriate visualizations is not easy. *RAVE* provides knowledge-based selection of visualizations that are appropriate for achieving analysis and data display goals. Once visualizations are selected, *RAVE* can automatically create and execute programs that generate the visualizations. In the process, it automatically transforms the target data set to satisfy constraints imposed by the visualization framework in which the data will be displayed. These capabilities allow the analyst to perform a guided exploration of domain-specific and generic visualization spaces, while concentrating on information extraction.

References

Arens, Y., Miller, L, Sondheimer, N., "Presentation Design Using an Integrated Knowledge Base," Intelligent User Interfaces, Sullivan, J.W., Tyler, S.W. eds., pp. 241-258, ACM Press, 1991.

Jacobson, A. S. and A. L. Berkin, LINKWINDS, A system for Interactive Scientific Data Analysis and Visualization, High Performance Computing 1993, Adrian Tentner ed., The Society for Computer Simulation, 1993.

Mackinlay, J.D., "Search Architectures for the Automatic Design of Graphical Presentations," Intelligent User Interfaces, Sullivan, J.W., Tyler, S.W. eds., pp. 281-292, ACM Press, 1991.

Shelley, E. G., A. Ghielmetti, E. Hertzberg, S. J. Battel, K. Altwegg-Von Burg, and H. Balsiger, The AMPTE CCE Hot Plasma Composition Experiment (HPCE), IEEE Trans on Geoscience and Remote Sensing, GE-23, 241, 1985.

Simoudis, E., Kerber, R., Livezey, B., "Recon: A database mining framework," submitted to the International Journal of Intelligent Information Systems.

Szuszczewicz, E. P., Mankofsky, Alan, and Charles C, Goodrich, SAVS: A Space Analysis and Visualization System, Proceedings of the 2nd NASA AISRP Workshop, 1992.

Planning, Scheduling, and Robotics