Integrated Visualization of Realtime Environmental Data

Elijah Saxon, Zoe Wood, Michael O'Neil, Chris Oates,
Jeremy Story, Suzana Djurcilov, and Alex Pang
University of California, Santa Cruz

Abstract

This paper presents the visualization effort of the REINAS (Realtime Environmental Information Network Analysis System) project. REINAS is a continuing engineering research and development system with the goal of designing, developing and testing an operational prototype system for data acquisition, data management, and visualization. It includes a growing web of networked remote and in-situ instruments, a federated geographical database, and an extensible visualization interface. REINAS focuses on the needs of both oceanographers and meteorologists who monitor realtime data or analyze retrospective data.

The visualization system of REINAS provides an integrated access via a distributed database to a host of different realtime data sources, numerical model forecasts, as well as more static data sets such as coastline and terrain information. It is designed as a modular tool-based system where users activate the appropriate visualization tools to generate different types of visualization products. It is also designed to be an extensible systems where tool designers can create new tools by programming and linking simpler elements together. Finally, it also supports visualization-on-demand *through the WWW where visualization products using the latest available data are generated on the fly based on information obtained from a web request.*

Key words and phrases: HDF, database, environmental visualization, web interface, visualization system, data fusion, data assimilation.

1 Introduction

The highlights of the visualization system that we describe in this paper are: its integration with a database front end that enables data to be queried and retrieved using a graphical interface and displayed using a variety of methods; the ability to query and display realtime data as soon as they become available in the database; simultaneous support for data set access through HDF files – thereby allowing both realtime monitoring/visualization as well as retrospective analysis of data sets; incorporation of uncertainty visualization techniques; support for geographically distributed collaborative visualization; and visualization product on demand through the web.

In the next sections, we discuss the background work and our motivation for creating a new visualization system, followed by an overview of the REINAS project in order to provide a context for the design considerations that were made; next we give a quick summary of the Spray Rendering system [1, 2] and the major changes that were made; this is followed by the architecture and major components of the new visualization system; some results obtained using the new system; and a status and summary section.

2 Background and Motivation

Through the years, universities and research labs have created a number of tools to address the visualization needs of different scientific community, some more specialized than others. For example Vis5D [3] and GEMPAK [4] are geared towards 3D meteorological data and provide almost all standard imaging options, while VTK [5] is an all-around visualization toolkit. With the growth of Internet we are starting to see more applications interactively available over the Web, such as WXP [6] - "The Weather Processor". In addition, powerful general-purpose commercial visualization packages such as Data Explorer(DX) and AVS can certainly serve the purpose of making visualizations out of scientific data. While these systems focus on or can be tailored to the environmental visualization arena, they do not address all of our project requirements, which are:

- Integrate data from geographically dispersed sensors showing both observed values and interpolated fields. These include sparsely distributed in-situ sensors, remote sensors, as well as numerical model simulations.
- Support sensors that collect different types of data, stored in different formats, and of different

temporal density and spatial dimensionality.

- Interactively select the region, time range, type of data of interest, and more importantly, keep track of time and space registration.
- Access to data in realtime as they are collected, as well as access through files for backward compatibility.

In addition, we had commitments to our users to provide a visual environment that can monitor current conditions, but also provide a retrospective analysis of past data; tools that can display forecast products and experimental results; and last, but not least, we wanted to service the general public by providing web products on demand. To accomplish this we had to resort to a new paradigm: a data-independent system which would allow file-loading, but primarily rely on interactive requests for specific data from a vast database. This system will be described in more detail in the context of the REINAS project.

3 Overview of REINAS

REINAS is a multi-year effort of the Baskin Center for Computer Engineering and Computer Science of the University of California, Santa Cruz (UCSC), in cooperation with environmental scientists from the Naval Postgraduate School (NPS), and Monterey Bay Aquarium Research Institute (MBARI). It is currently in its fifth and final year. We have gone through the steps of problem and concept identification, requirement specification, system design and implementation, and are currently in the midst of experimentation and evaluation as well as transitioning to an operational version. Online information regarding REINAS including technical reports are available from `http://www.cse.ucsc.edu/research/reinas`.

The goal of this project is to bring modern technologies to bear upon the problem of realtime environmental (oceanographic and meteorological) data acquisition, management, and visualization/analysis. As a focus area, we are looking at the phenomena within the regional scale of Monterey Bay, California. Since physical changes happen at much smaller time scales within our region of interest (compared to global or climate studies), the main challenge of REINAS is to provide realtime environmental information of the physical parameters. The tools that are being used to help realize this goal are a combination of wireless, networked instruments, land and water based stationary and mobile platforms, remote instrument steering and control, spatial/temporal database management, artificial intelligence, flexible tool-based visualization, multimedia, collaboration software, and data assimilation of field measurements into numerical models.

While the main driving requirement for REINAS is the realtime measurement and access of field data, it also has to support the needs of three classes of users. These are: operational users, scientific users, and developers/instrumentation engineers. Operational users include forecasters who need the most recent measurements to produce short-range forecasting or nowcasting, policy makers or planners who need a bird's eye view as well as the ability to zoom in on areas of interest, disaster control planners who need current measurements as well as models to compare the impact of various remedial alternatives, and finally a growing list of recreational users who are interested in the current surf, sail [7], or ski conditions. Scientific users include retrospective researchers who need synoptic views for historical analyses, experimental researchers who need to combine current measurements into model runs, and sensor scientists who need to control instruments as the need arises – e.g. adjust sampling rate or point instrument at certain directions as a weather front approaches. The third class of users are system developers and instrumentation engineers who need to add, recalibrate, or remove instruments from the field, add or remove instruments from the network of instruments without bringing down the entire system, modify or add features to a running system, upload data from other sources into the database, etc.

From the list above, it is clear that the REINAS design must support both realtime and retrospective analyses. It also has to transition smoothly from a development system to an operational system. Finally, the effort here must be easily replicated to other regional areas of interest. Hence, the system must also be portable.

The REINAS system architecture is broken down into three main components: data acquisition, data management, and data visualization. Figure 1 shows a high-level view of the system. Lines indicate physical or wireless network connections. Lines are bidirectional unless indicated otherwise. Data flows generally from left to right, and feedback and control generally flow from right to left.

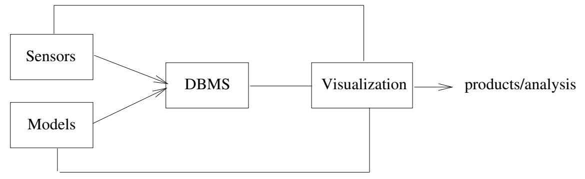


Figure 1: Major REINAS components.

4 Overview of SPRAY

Spray Rendering was the original design of the REINAS visualization system. It was divided into three modes to address the needs of the three target user groups: *monitor* mode where users can watch the most current state of the environment (e.g. top left of Figure 2 shows interpolated winds from multiple sensors distributed over the Monterey Bay, while bottom left image shows derived ocean currents from multiple radial measurements); *forecast* mode allowed operational forecasters to generate standard and customized weather products with a click of a few buttons (e.g. middle column of Figure 2); and *analysis* mode where scientific users can perform retrospective analyses of their data using a variety of visualization methods.

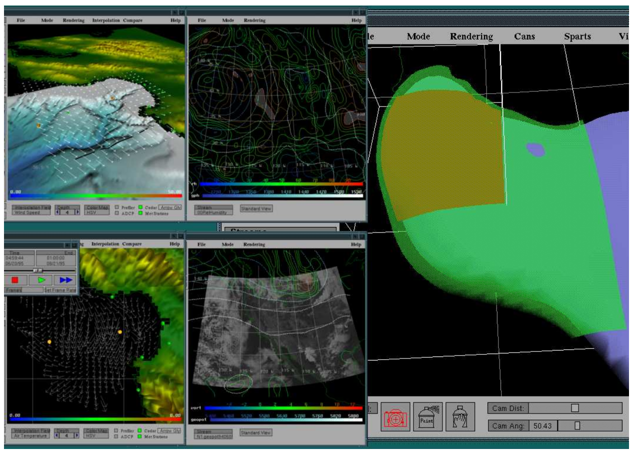


Figure 2: Spray Rendering system.

Spray rendering provides the users with the metaphor of spray painting their data sets as a means of visualizing them. In its simplest form, data are rendered by the color of the paint particles. By using different types of paint particles, data can be visualized in different ways. The key component of spray rendering is how the paint particles are defined. They are essentially smart particles (or sparts) which are sent into the data space to seek out features of interest and highlight them. Among the advantages of this visualization framework are: grid independence (sparts operate in a local subset of the data space and do not care whether data is regularly or irregularly gridded), ability to handle large data sets (sparts can be "large" and provide a lower resolution view of the data set or they can be "small" and provide a detailed view of an area of interest), extensible (it is easy to design new sparts). Sparts can also travel through time-dependent data sets. Existing view-independent visualization methods such as isosurfaces, streamlines, etc. can be encapsulated using this metaphor.

At the onset of the project, it was clearly impossible to identify and anticipate all the needs and requests of the visualization users. Thus, it was a major design goal to make the system flexible and extensible allowing it to grow with the needs of the users as they arise. The result is a visual programming extension to spray rendering that allows users to graphically create new sparts (and hence, new visualization methods) using a mix and match paradigm [8]. Sparts are created by hooking up different components. These components are organized into four categories: *Target* components are functions that detect features in the data set; *Visual* components are functions that describe the graphical representation of the feature that was detected; *Position* functions update the current position of the spart. These can be absolute movements or dependent on the data, as in vector fields; and *Death* functions determine when the spart should die. There is also a birth function in this category that spawns new sparts. A spart composition is the specification of the components that make up the spart and the connections between them. Users can select components from a browser, drop it onto a canvas and graphically connect them. This composition defines how the spart behaves at the current location. The compositions and the components are usually quite simple. However, complex visualizations can be obtained by multiple applications of several sparts.

While Spray Rendering provided a powerful visualization paradigm and a flexible and extensible interface, it also has some limitations that were discovered during the evaluation phase of the project. It was implemented in C and IRIX GL thereby limiting its portability to a limited number of platforms. The separation of its functionality into different modes was perceived as a hindrance to some users who wanted to compare observed or measured data against model forecast data. Finally, the graphical interface that made for flexibility of creating new visualization methods contributed to the difficulty in the learning curve for new users. That is, a sacrifice was made to trade off ease of use against flexibility of the system. These shortcomings are addressed in the redesign of Spray.

5 System Architecture and Components of PET SLUG

The new REINAS visualization system is called PET SLUG. The banana slug is the mascot of our university, while PET is an acronym for Products, Elements, and Tools which reflect the architecture of the new system.

The new system is being developed using OpenGL, C++, and xforms [9] for improved portability. It uses dynamic shared object libraries to facilitate phased incremental release of the system. It is also a tool-based system as opposed to a mode-based system allowing users to activate different visualization tools as needed. Finally, in preliminary tests, the system is perceived as being easier to use while the extensibility is hidden from the user. We now describe the architecture of PET SLUG and its major components.

PET SLUG is an interactive visualization system for space and time synchronized visualization of realtime environmental data. It uses a tool-based approach where users activate one or more visualization tools to visualize their data. Figure 3 shows the graphical interface of PET SLUG.

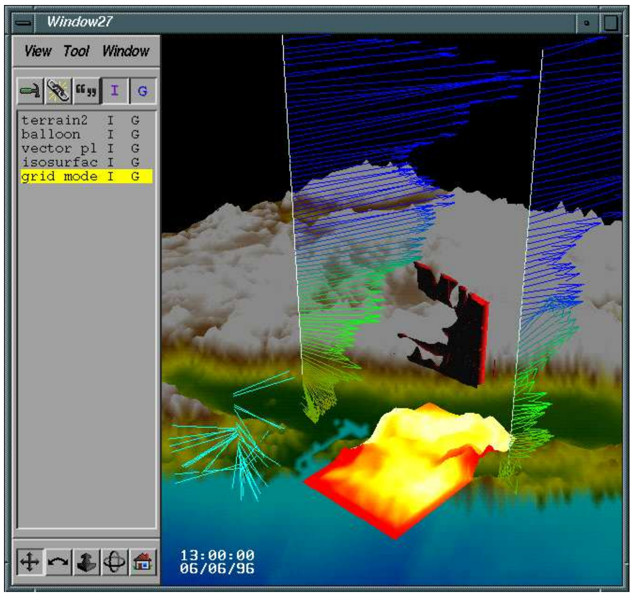


Figure 3: Graphical interface of PET SLUG showing visualizations a variety of realtime environmental data.

PET SLUG is designed to be readily extended by users with varying requirements and levels of expertise. As such, it is organized into a three level hierarchy: products, tools, and elements. Products are collections of tools with specific preset parameters and data (see Figure 4). In this way, a first time or casual user can simply open a product file and see a meaningful visualization. An example might be a temperature isosurface that is pseudo-colored with pressure values over the Monterey Bay. The frequent user, on the other hand, is more likely to use tools to build their visualization. Typically, a tool is used to implement a common visualization method such as isosurfaces, contour lines, streamlines, etc. Tools can be composed from other tools and even smaller building blocks called elements (see Figure 5). Elements are small C++ programs with a common API that are linked into PET SLUG at run time. The expert user or programmer can write elements in order to expand the capabilities of slug to fit their specific needs. Examples of elements include different types of spatial and temporal interpolators, glyph makers, data transformation modules, etc.

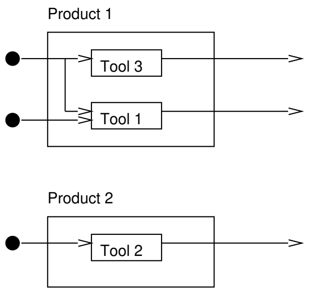


Figure 4: Visualization products are composed from one or more tools with inputs bound to specific data sets.

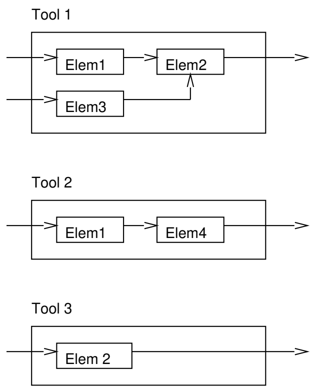
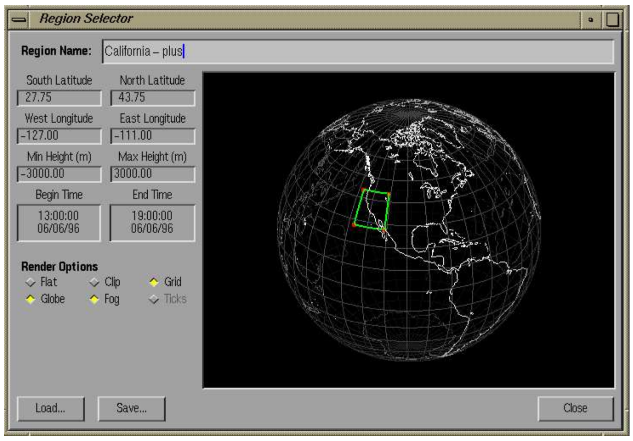


Figure 5: Visualization tools are made up of one or more elements.

The major components of PET SLUG are: database query interface, dataset manager, region selector, animation controller, geometry library, and tools. In addition, there is support for multiple graphics window (to be used for collaborative visualization later on), customization of visualization session to user preference, and visualization products on demand through the web. These are described next.

5.1 Database query interface

A graphical user interface is provided for the user to query the database via an application programmer's interface [10] that isolates the visualization application from the database development efforts. Queries are specified by three general characteristics: region of interest, time period of interest, and environmental parameter (e.g. pressure, humidity, etc). The first two are specified via the *region selector* (see Figure 6) or via user customization (see Section 5.6). Different map projections are also supported here. The latter is specified through a list of commonly used environmental parameters, as well as special types, such as data collected from a moving platform (see Figure 11). We have support for three general types of database queries: (a) query by station (e.g. give me information from the meteorological station at UCSC), (b) query by environmental parameter (e.g. give me all the temperature data from the region/time of interest), and (c) query by instrument (e.g. give me all the wind profiler data for the region/time of interest). The *dataset manager* then keeps track of the internal data structure used to hold the query results (see Figure 7). In addition, the dataset manager also allows static data stored in HDF [11] files to be loaded. One of the advantages of having a separate dataset manager is the reduction in data replication when datasets are processed elements within a tool.



Region Name: California – plus

South Latitude	North Latitude
27.75	43.75

West Longitude	East Longitude
-127.00	-111.00

Min Height (m)	Max Height (m)
-3000.00	3000.00

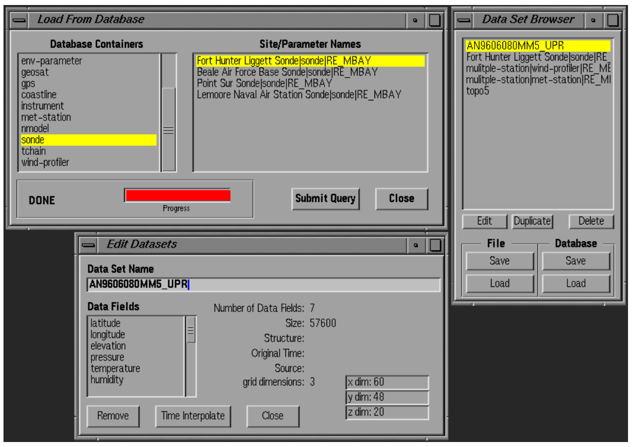
Begin Time	End Time
13:00:00 06/06/96	19:00:00 06/06/96

Render Options: Flat [✓ Globe] Clip [✓ Fog] [✓ Grid] Ticks

Figure 6: Region selector for specifying region and time period of interest.

5.2 Animation controller

Most data queried through the database have an associated time stamp. Some do not e.g. coastline, terrain, and bathymetry data are static. The animation controller (see Figure 8) provides time synchronized playback of data from multiple sources (e.g. met stations, GOES satellite data, vertical wind profilers, etc.). Time sensitive data are filtered through time interpolator elements by the different tools and displayed



Database Containers:
env-parameter
geosat
gps
coastline
instrument
met-station
nmodel
sonde
tchain
wind-profiler

Site/Parameter Names:
Fort Hunter Liggett Sonde|sonde|RE_MBAY
Beale Air Force Base Sonde|sonde|RE_MBAY
Point Sur Sonde|sonde|RE_MBAY
Lemoore Naval Air Station Sonde|sonde|RE_MBAY

DONE

Data Set Browser:
AN9606080MM5_UPR
Fort Hunter Liggett Sonde|sonde|RE_
mulitple-station|wind-profiler|RE_ME
mulitple-station|met-station|RE_MI
topo5

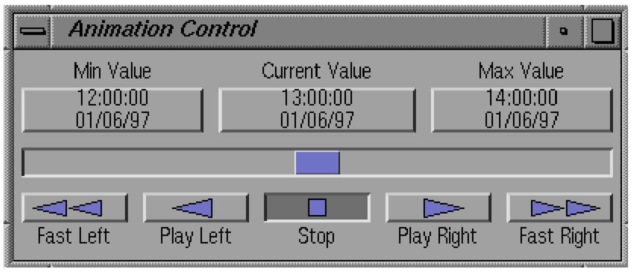
Data Set Name: AN9606080MM5_UPR

Data Fields:
latitude
longitude
elevation
pressure
temperature
humidity

Number of Data Fields: 7
Size: 57600
Structure:
Original Time:
Source:
grid dimensions: 3
x dim: 60
y dim: 48
z dim: 20

Figure 7: Dataset manager for queried data sets.

at user specified time resolutions (e.g. hourly, or every minute, etc.) In addition, a playback facility of historical data is also provided.



Min Value	Current Value	Max Value
12:00:00 01/06/97	13:00:00 01/06/97	14:00:00 01/06/97

Figure 8: Animation controller for time synchronized display of data.

5.3 Geometry library

The geometry library operates behind the scene to provide support for tools requesting different types of geometry to be mapped to data values and sent to the renderer. Among the items supported in the library are: 3D text, color tables, various types of glyphs, polygon base class, lines, and points. These items are used by individual tools to create a hierarchical display list that contains geometry nodes to represent a particular visualization. Figure 9 shows the geometry tree for a tool displaying arrows and a polygonal surface, both using a common color-map. The display list is then sent to the renderer for display in a specified window.

5.4 Visualization tools

We provide a variety of visualization tools. In addition, we will be providing a graphical interface for creating new visualization tools from simpler elements.

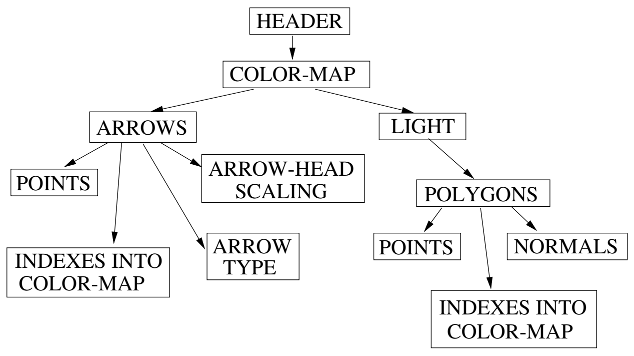


Figure 9: A sample geometry tree.

The current set of tools include both single-element and multiple-element tools. Each tool can be instantiated more than once if multiple copies are needed. Different data sets can be bound to different tool instances using another graphical interface. Each tool may have an optional graphical interface specific to that tool. For example, an isosurface tool will have a slider for specifying a threshold value. Among the tools available in our current release are:

1. Balloon tool for displaying one or more radiosonde data launched from weather balloons. A variation of this tool can be used to display data from drifting buoys. Another variation is used to display data collected from an aircraft (see Figure 11).
2. Contour tool for displaying iso-bars and isotherms.
3. Coastline tool for displaying higher resolution coastline data over selected region.
4. Interpolation element for spatial/temporal interpolation of environmental data.
5. Isosurface tool used to visually delimit the portion of gridded data that is below the given threshold value.
6. Line-of-sight tool for determining visibility and/or occlusions between two geographic locations.
7. Mini-cubes/mini-spheres tool for displaying properties of large volumes at each data point.
8. Pseudo-colored grid tool for displaying 2D layers of data. Several options for mapping data values to color are provided and can further be edited by the user.
9. Quake tool for displaying fault lines and earthquake events.
10. Satellite tool for displaying GOES7, GOES9 (1km and 4km) images at various display resolutions.
11. Station tool for displaying information from one or more stationary meteorological stations and buoy sites.
12. Terrain tool for displaying topography/bathymetry of selected region.
13. Vector plot tool for displaying vector information such as those from CODAR (ocean surface current data) and vertical wind profilers. This tool provides several types of glyphs for representing vector information: arrows, barbs, and uncertainty glyphs [12].

5.5 Multiple graphics windows

During a visualization, the user may create one or more windows for the renderers to send their output. Each window may have its own region, camera, and associated set of tools. Additionally, these object may be shared across multiple windows. The graphics in a window, with its set of tools and datasets, constitute what we refer to as a visualization product. Thus, generating several graphics windows corresponds to generating multiple visualization products. Figures 12 and 13 are some examples of graphics windows or visualization products.

5.6 User customization

The state and contents of each graphics window can also be saved, edited, and customized. Information about tools, datasets, including window placement, etc. are included in the saved information. This is useful if a user usually examines a set of queries over a particular region or if the user need to continue from an earlier session.

5.7 Products on demand

Since each graphics window is a visualization product and since the information required to re-generate the window are saved and can be loaded at a later time, we provide a WWW forms interface for requesting visualization products. A process on our web server processes the forms input and creates a visualization product specification file from a template. Next, PET SLUG is invoked with the requested product. An off-screen renderer generates the visualization which is then sent back to the requester.

6 Results

In this section, we present some of the visualization products from PET SLUG in the context of two different tasks.

6.1 Monitoring the environment

The main goal here is to give the user an idea of the current state of events in the environment. The simplest approach is to obtain the most recent data available, perform time filtering operations, and display data at the sites where they are collected – e.g. Figure 10. However, the process is not as simple as it appears. Some of the issues that need to be addressed are errors introduced in interpolating very sparse data sets, efficient means of dealing with scattered data, and how to treat missing or noisy data. In our collaboration with environmental scientists, there are instances where they may bring in specialized instruments to study an environmental phenomenon. An example of this is illustrated in Figure 11 when a heavily instrumented airplane was used to fly a “grid” pattern over the Monterey Bay.

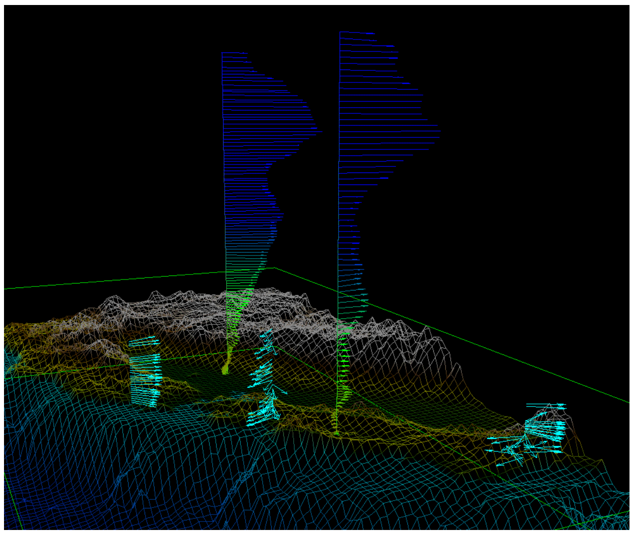


Figure 10: Visualization product showing data from both vertical wind profilers and balloon launches.

6.2 Data assimilation

Data assimilation is the process of resolving the differences between observed and forecasted values and integrating updated conditions back into the numerical forecast in order to improve accuracy [13]. In the process of forecasting, incoming observational data is continuously blended into the model state in such a way as to minimize expected error. There are many parameters to this operation, all of which can have a profound effect on the final forecast. We try to help the makers of the data assimilation models by visually representing the difference made by their parameter choices. By taking the 3D point-by-point difference between the real weather outcome and the forecast produced by a specific data assimilation the meteorologist

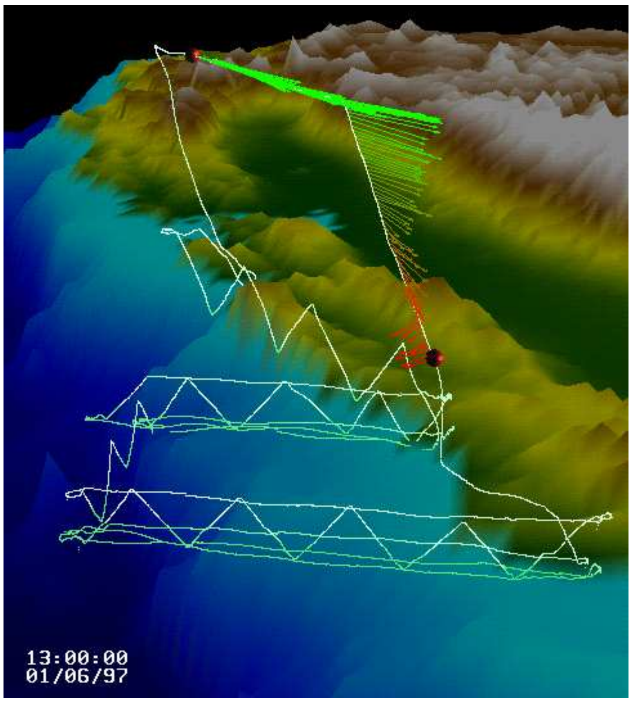


Figure 11: Visualization product showing wind data over a section of an airplane flight path.

can better assess the nuances of the model's performance under a variety of conditions. Figures 12 and 13 are sample visualizations that are used to aid the analyst in the assimilation process.

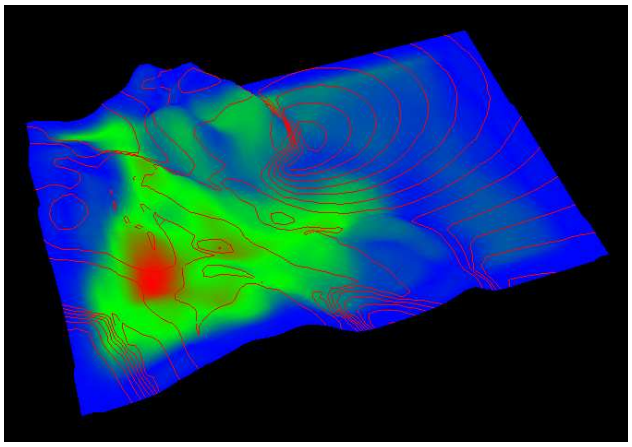


Figure 12: Visualization product pseudo-colored surface of water-vapor difference values overlaid with contour lines of temperature.

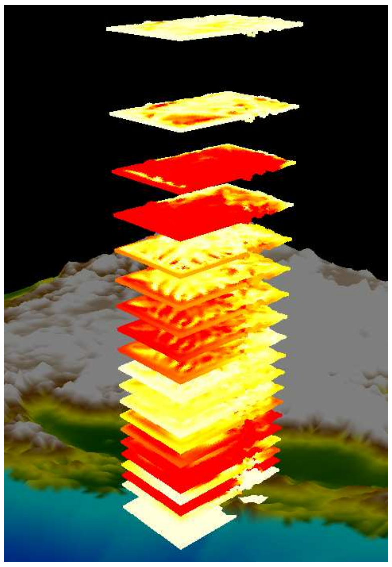


Figure 13: Visualization product using the mini-cube tool showing the temperature differences at different heights.

7 Summary and Conclusions

The current implementation of PET SLUG is based on lessons learned from the earlier Spray system. PET SLUG is still undergoing development and is targeted for completion and distribution in September 1997. Among the items that we plan to include by then are:

1. Making products available through the web.
2. Port and test on other platforms.
3. Graphical editor for composing tools from elements.
4. Collaborative visualization to remote users to share data and create visualizations on different graphics windows

In summary, we have presented the REINAS visualization system. Its main contribution is an easy to use interface for accessing realtime environmental data from the database, and is highly extensible for creating new visualization tools.

Acknowledgements

We would like to thank Dan Fernandez and Wendell Nuss for helping drive the development of PET SLUG from the users' point of view. We would also like to thank Doug Miller for collaboration on data assimilation, and Craig Wittenbrink for collaborations on earlier REINAS visualization efforts. The Santa Cruz Laboratory for Visualization and Graphics provided a wonderful research environment and we would like to acknowledge that as well. Finally, the authors would also like to acknowledge support from ONR grant N00014-92-J-1807, NSF grant IRI-9423881, NASA Cooperative Agreement NCC2-5176.

References

[1] Alex Pang. Spray rendering. *IEEE Computer Graphics and Applications*, 14(5):57 – 63, 1994.

[2] Alex Pang and Dan Fernandez. REINAS visualization and instrumentation. In *Proceedings of Oceans'95 Conference*, pages 1892–1899. IEEE, October 1995.

[3] William Hibbard and David Santek. The Vis-5D system for easy interactive visualization. In *Visualization '90*, pages 28–35, 1990.

[4] M. L. desJardins, K. F. Brill, and S. S. Schotz. Use of GEMPAK on Unix workstations. In *Seventh Conference on Interactive Information Processing Systems for Meteorology, Hydrology, and Oceanography, New Orleans, LA*, pages 449–451, 1991.

[5] Will Schroeder, Ken Martin, and Bill Lorensen. *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*. Prentice Hall, 1996. URL: http://www.cs.rpi.edu/ martink.

[6] Dan Vietor. Wxp-the weather processor. URL: http://wxp.atms.purdue.edu.

[7] Eric C. Rosen. REINAS home page. URL: http://www.cse.ucsc.edu/research/reinas, 1995. Computer Engineering, University of California, Santa Cruz.

[8] Alex Pang and Naim Alper. Mix & Match: A construction kit for visualization. In *Proceedings: Visualization '94*, pages 302 – 309. IEEE Computer Society, 1994.

[9] T.C. Zhao. XFORMS home page. URL: http://bragg.phys.uwm.edu/xforms.

[10] Craig M. Wittenbrink, Eric Rosen, Alex Pang, Suresh Lodha, and Patrick Mantey. Realtime database support for environmental visualization. In G. Grinstein, U. Lang, and A. Wierse, editors, *Lecture Notes in Computer Science, Second Workshop on Database Issues for Data Visualization*, volume 1183, pages 111–130. Springer Verlag, 1996.

[11] NCSA. Hierarchical data format (HDF) home page. URL: http://hdf.ncsa.uiuc.edu.

[12] Craig M. Wittenbrink, Alex T. Pang, and Suresh K. Lodha. Glyphs for visualizing uncertainty in vector fields. *IEEE Transactions on Visualization and Computer Graphics*, 2(3):266–279, September 1996. Short version in SPIE Proceeding on Visual Data Exploration and Analysis, pages 87-100, 1995.

[13] Suzana Djurcilov and Alex Pang. Visualization tools for data assimilation. In *SPIE Visual Data Exploration and Analysis IV*, pages 67–76. SPIE, February 1997.