

# Navigation System for Virtual Reality Visualization of Large-Scale Datasets

## Abstract

*This project involves the design and implementation of an interactive immersive virtual reality system allowing rapid high-resolution rendering of large scale datasets including, but not limited to, astronomical, biomedical, and seismic data. The purpose of the project is to study how large sparse datasets can be rendered interactively, and how the user can be enabled to effectively navigate large, mostly empty spaces without getting lost in a void and by maintaining precision in the navigation when needed, that is in the proximity of the object. A prototype implementation of the system framework will be applied to 3D models of different datasets featuring an interactively controlled camera and guided navigation speed-controlled by the proximity of objects in the scene. Spatial and temporal scaling issues, which are essential to the study due to specifics of sparse data processing, will be addressed; a dynamic navigation method will be developed, with scaling parameters readjusting in real time for efficient representation of the data.*

*Techniques employed in the design of this system can potentially be used to build scalable, easily navigable and extensible models of large-scale entities. During the study, the following data models will be used: (i) the known universe, (ii) a human cardiovascular system, and (iii) a geophysical model of the Earth. All of these data sets are available and share the same properties: large size and sparsity. The system will be designed as an intuitive simulation tool, helpful in identifying novel approaches for control and visualization in such applications as simulated space flights, cardiovascular disease treatment, and earthquake ground motion and structural response analysis.*

## **1. Introduction**

These days, scientific visualization and animation are becoming more useful in various real-world fields ranging from biomedical and civil engineering to biological sciences and physics. The latter includes astrophysical applications such as stellar and planetary simulations, which facilitate space exploration, provide efficient ways of analyzing astrophysical processes, and serve an important role in mission operations and flight planning. In the light of the active development of high speed hardware systems and efficient rendering algorithms, large scale datasets, such as stellar data describing one or several galaxies, can be potentially visualized and interacted with in a virtual reality environment with realistic resolution at real-time data processing rates. This could be useful in observing stellar motion patterns, planning space probe missions and forming new methods of data acquisition.

In a pilot study, a 3D navigation system has been developed featuring a simulation of the solar system and the empty space between planets. This is the first step in the design of a visualization system that will enable interactive three-dimensional simulations of stellar data in virtual environments. Hardware platforms include CRT-based stereoscopic desktop displays and large-scale displays, such as a Responsive Workbench [5], a CAVE™ [2], or a fully immersive curved screen for panoramic stereoscopic image display that will become available in the new California Institute for Telecommunications and Information Technology, Cal-(IT)<sup>2</sup>.

## **2. Previous Work**

As part of an independent research project (ENGRECE 199) this Winter, I was conducting research at the Creative Interactive Visualization Lab (CIVL) under mentoring of Dr. Meyer (Department of Electrical Engineering and Computer Science, UC Irvine). I created a virtual model of the Solar System, which allowed for initial tests of navigation algorithms and represented a good example of a dataset sparse enough to make conventional linear scaling impossible to use, and also simple enough to ease the design and debugging process.

### **2.1. Programming Interface**

To implement the project, I chose the industry's most widely used graphics API, OpenGL [7], complemented with the OpenGL Utility Toolkit (GLUT) interface [4] due to their high portability, reliability, well-defined architecture and stability [1], in effect, ensuring the

universality of our future visualization system and enabling it to run on any platform with minor, if any, changes to the code.

The following section describes details specific to the implementation of the chosen prototype application (space simulation).

## 2.2. Finding and Updating Planetary Coordinates

Internally, an astronomical object is represented with a data structure *Star*, which contains fields indicating the object's parameters such as its name, apparent magnitude, path to texture file, semimajor axis  $a$ , eccentricity  $e$ , orbit inclination, obliquity, and others. From these, semiminor axis is  $b = a\sqrt{1-e^2}$ , the distance from the center of the orbit to the Sun (which is assumed to be fixed at the origin) is  $f=ae$ , and the period is  $P=a^{3/2}$  (by Kepler's Third Law). Then, the mean anomaly is  $M=2\pi t/P$ , where  $t$  is the time elapsed from some initial moment  $T_0$ , at which the planet passed through perihelion, or some other fixed moment. From Kepler's equation  $M=E-esinE$ , the eccentric anomaly  $E$  is determined. A planet's coordinates (defined by  $(x, y, z)$ ) in OpenGL's rectangular coordinate system at time  $t$  are  $x=acosE-f$  and  $y=bsinE$ , while the angle defined by the inclination parameter of the structure determines the tilt of the elliptical orbit and thus the planet's  $z$  coordinate.

## 2.3. Magnitude and Distance Scaling Issues

Although physically correct, the model described above is of little practical use because of the large scale of the Solar System and drastically differing sizes of the planets and the Sun. For example, one of the world's largest scale models of the Solar System in Maine features a 50-foot Sun and one-inch Pluto located 40 miles away [6]. Stellar data, which is potentially going to be visualized in the system, is spread out even more, so we have to introduce some sort of scaling factors for magnitudes and distances. As of now, magnitude is scaled as  $M'=\log_{10}(AM)/B$ , where  $M$  is original value of apparent magnitude mentioned above (in astronomical units), and  $A$  and  $B$  are constants. In order to determine the values of the planet's coordinates, the angle  $\theta$  is found as  $\tan^{-1}(y/x)$ , while the distance  $r = \sqrt{x^2 + y^2}$  is transformed on a linear scale ( $r'=kr$ ), and coordinates are updated:  $x'=r'\cos\theta$  and  $y'=r'\sin\theta$ . As orbit inclinations are initially small, there is clearly no reason to alter the  $z$  coordinate of the planet.

It was shown experimentally that the optimal scaling factors for the Solar System, allowing a good representation of magnitudes and positions of the Sun and the planets, were

determined to be  $A=1$ ,  $B=6.0$  for the Sun,  $A=10$ ,  $B=3.0$  for the planets, and  $k=5.0$ , respectively. This allows all the planets to be visible within the range of the screen at the expense of having a physically incorrect magnitude scaling (e.g. the Sun is just a few times larger than the Earth).

## **2.4. Planetary Object Rendering**

The Sun and the planets are represented with `gluSphere` objects to which appropriate textures are applied (2D mipmapping is used for hierarchical selection of texture resolution). The file format used for textures is the portable pixel map (PPM) format. In the `draw()` routine, a set of translations and rotations is performed every time on each object, so that it moves along its orbit and rotates about its axis according to given parameters and the current time  $t$  (updated every time after `draw()` is executed).

## **2.5. Interactive Controls**

In the prototype application, the motion of the point of view and the motion of the planets in the Solar System model can be controlled in the following way: The user's view (equivalent to the camera view) is pointed at the origin (the Sun). The camera can be rotated about the origin in any of three directions by using the keyboard. The rotation speed of the camera can be changed in a similar way as well. The rate at which the current time  $t$  is changing can be altered and can take both positive and negative values. This allows observation of "future" and "past" motion of the planets and changing the speed at which they rotate around the Sun. Keyboard and mouse can be used to zoom in and out with single clicks of a button as well.

In order to ease orientation in the planetary system, rectangular coordinate axes and planet names are displayed and can always be toggled using a pop-up menu associated with the right mouse button. A star field background can be turned on and off using a toggle switch as well, allowing to concentrate just on the available planetary objects.

## **2.5. Input and Output**

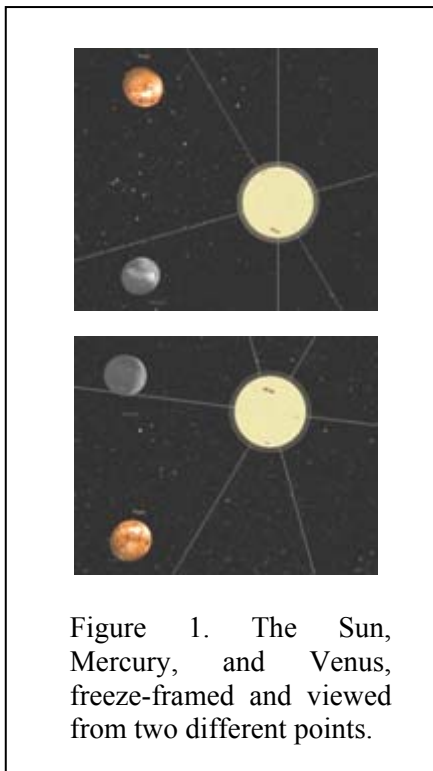
### **2.5.1. Planetary and Stellar Data Input**

As the project is advancing through its early stages, all parameters are loaded from hardcoded arrays of data; however, thanks to a modular loading method, a seamless transition to

loading from a larger data file will be implemented when larger sets of data are processed. The Bright Star Catalogue (BSC) [3], which contains 9110 objects and is considered to be the reference of choice for astronomical and astrophysical data for stars brighter than magnitude 6.5, would be an excellent source for the stellar data visualization part of the project. An appropriate parser will be written to extract data from BSC catalogue files.

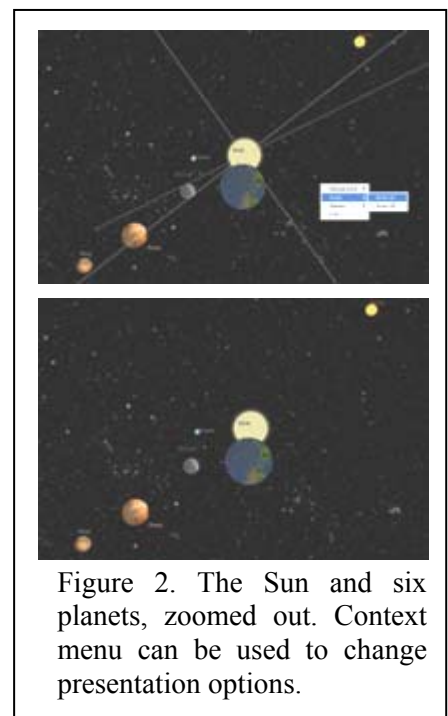
### 2.5.2. User Control Input

As mentioned above, all control is currently done with keyboard and mouse. During summer, use of alternative, more intuitive controls, such as joysticks, wand-type controllers, as well as head and hand tracking specific to the Responsive Workbench or the CAVE™ environments, is anticipated.



### 2.5.3. Output visualization

The Solar System model runs on any monitor or a projector, and, with minimal changes, can be adapted to presentation in stereo mode using 3D shutter glasses, a compatible video card, and a CRT monitor or a projector system that supports stereo (excluding LCD matrix-based projectors). Ultimately, the system should be adapted to run in a CAVE™ virtual environment, which would allow high level of immersion at real-time frame rates.

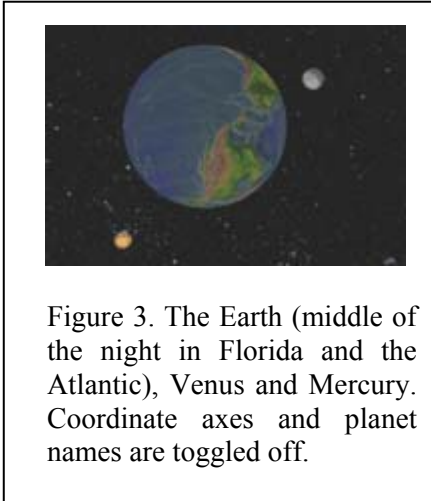


### 2.6. Simulation Results

As of now, the system runs efficiently and yields expected results; the Solar System is visualized fairly realistically at frame rates high enough to maintain interactivity (5-20 fps).

A possible application might include stopping the motion of the planets and analyzing the freeze-framed situation by the moving camera around a static 3D scene (figure 1).

As it was mentioned above, coordinate axes, star field background and planet names can be toggled on/off through a menu (figure 2). The Sun acts as a light source, so a fairly realistic lighting scheme is used to determine how the planets are lit. Figure 3 shows the dark side of the Earth (the opposite side is turned towards the Sun).



### **3. Planned Work and Responsibilities.**

I have designed an interactive 3D model of the Solar System which served as an example of the usefulness of a planetary and stellar data visualization system. Such a system will potentially provide intuitive visual cues absent in purely analytical environments and thus provide insight in applications such as mission planning and control.

This Spring and Summer, I am developing this project in two key directions. The first direction is the further development of efficient visualization algorithms allowing interactive-fast rendering of large volumes of stellar and other data as well as working on dynamic (as opposed to currently used static) spatial and temporal scaling methods which would allow the creation of easily navigable and scalable star data structures featuring high resolution modeling of data arrays ranging from constellations to galaxies to, ultimately, the known universe. This direction will also include a design of hierarchical star object representation, which should further increase the processing rates of the system.

The second prospective work direction, to be done in summer, includes porting the framework into virtual reality environments, ranging from stereo-ready commodity video cards complemented with LCD shutter glasses, to stereo projection systems, to immersive environments such as Responsive Workbench or CAVE™. Keyboard and mouse would be replaced with control devices increasing the “presence feel” such as force feedback joysticks or wand-type controllers in CAVE™. For the prototype implementation, the curved screen in the Cal-(IT)<sup>2</sup> building (see below) will be used, which will become available in Fall. This prospective direction would allow users of the system to have the Universe “at their fingertips.”

The work will result in developing a portable modular navigation system that can be easily integrated into a virtual reality system at the California Institute for Telecommunications

and Information Technology, Cal-(IT)<sup>2</sup>, a joint project of UC Irvine and UC San Diego. This system will be used to visualize a wide variety of simulation tasks from different disciplines and will be instrumental to many studies, including ones that involve large volumes of data. These tasks obviously call for a uniform, universal system that will allow intuitive browsing and handling of individual and grouped objects and should feature dynamic, adaptive navigation. Once completed, this project will provide such a system and provide a very useful tool to be used in a plethora of studies conducted by researchers from Cal-(IT)<sup>2</sup>, UCI, and other institutions.

#### 4. Timeline

May 2004	Study characteristics and ranges of astrophysical, biomedical, and geophysical data presented in different formats (work with CIVL researchers at UCI [EECS] on corresponding subjects).
June 2004	Design algorithms to provide dynamic scaling of hierarchical data. Design and implement a navigation module featuring interactive temporal and spatial scaling and panning controls.
July 2004	Test and debug the module by navigating smaller datasets. Expand browsable data to large-scale limits (entire cardiovascular system, several galaxies, Greater Los Angeles Basin).
August 2004	Porting the system into various virtual reality environments with alternative input/output devices. Complete work on path storage and fly-through recording.
September 2004	Integrate navigation system into VR environment at new Cal-(IT) <sup>2</sup> visualization lab (construction should be finished by that time). Have user groups comprised of students and faculty in associated fields (astronomy, civil engineering, biomedical engineering, etc.) test the system. Enhance navigation based on user input.
May 2005	Present the system and findings at the UCI Undergraduate Research Symposium.

## References

- [1] E. Angel. *Interactive computer graphics: a top-down approach with OpenGL, 2nd ed.* Addison-Wesley Publishing Company, Reading, MA, 2000.
- [2] C. Cruz-Neira, D. J. Sandin, and T. A. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the cave. In *SIGGRAPH 93 Computer Graphics Conference*, pages 135-142, Anaheim, Aug. 1993. ACM SIGGRAPH.
- [3] D. Hoffleit, W. H. Warren Jr. *The Bright Star Catalogue, 5th rev. ed.* Astronomical Data Center, NSSDC/ADC, 1991.
- [4] M. J. Kilgard. *The OpenGL Utility Toolkit (GLUT) Programming Interface: API Version 3.* Silicon Graphics Incorporated, 1996.
- [5] W. Krueger and B. Froehlich. The responsive workbench. In *IEEE Computer Graphics and Applications*. 14(32):12-15, 1994.
- [6] S. H. Samuels. Caution, planets ahead. *Smithsonian Magazine*, 3:32-35, 2003.
- [7] M. Woo, J. Neider, T. Davis, and D. Shreiner. *OpenGL Programming Guide, 3rd ed.* Addison-Wesley Publishing Company, Boston, MA, 1999.