AIAA 99-0691

Real-Time Lunar Prospector Data Visualization Using Web-Based Java

D. G. Deardorff and B. D. Green

NASA Ames Research Center
Moffett Field, CA

37th Aerospace Sciences Meeting & Exhibit
January 11-14, 1999 / Reno, NV

REAL-TIME LUNAR PROSPECTOR DATA VISUALIZATION USING WEB-BASED JAVA

D. Glenn Deardorff* and Bryan D. Green*

NASA Ames Research Center, Moffett Field, California

ABSTRACT

The Lunar Prospector was co-developed by NASA Ames Research Center and Lockheed Martin, and was launched on January 6th, 1998. Its mission is to search for water ice and various elements in the Moon's surface, map its magnetic and gravity fields, and detect volcanic activity. For the first time, the World Wide Web is being used to graphically display near-real-time data from a planetary exploration mission to the global public. Science data from the craft's instruments, as well as engineering data for the spacecraft subsystems, are continuously displayed in time-varying XY plots. The craft's current location is displayed relative to the whole Moon, and as an off-craft observer would see in the reference frame of the craft, with the lunar terrain scrolling underneath. These features are implemented as Java applets. Analyzed data (element and mass distribution) is presented as 3D lunar maps using VRML and Javascript. During the development phase, implementations of the Java Virtual Machine were just beginning to mature enough to adequately accommodate our target featureset; incomplete and varying implementations were the biggest bottleneck to our ideal of ubiquitous browser access. Bottlenecks notwithstanding, the reaction from the Internet community was overwhelmingly enthusiastic.

LUNAR PROSPECTOR MISSION OVERVIEW

The primary focus of the Apollo missions was to demonstrate the feasibility of technologies for successfully sending people and rock samples to and from the Moon. Most of the scientific data collected was concentrated in narrow regions around the lunar equator, leaving 75% of the Moon's surface unmapped. Since then, all other U.S. lunar explorations have been flybys, except for the Clementine mission, a Department of Defense mission whose purpose was to provide detailed surface images and conduct laser-ranging altimetry, as well as some charged-particle experiments. This mission, which flew for 2 months in 1994, produced findings which hinted at the possibility of water ice at the south pole.[1]

Lunar Prospector, with its complement of five scientific instruments and its low-altitude orbit, was designed to greatly extend the quality and quantity of data on surface composition, as well as to provide detailed magnetic and gravitational surveys. At the end of its nominal one-year mission, its circular 100 km high orbit will be changed to an elliptical one which will bring it to within 10 km of the lunar surface for highly detailed explorations of regions of interest. Its instruments include a neutron spectrometer for mapping the concentrations of hydrogen, and by inference, water ice; a gamma-ray spectrometer for mapping the abundance of uranium, iron, titanium, and other elements; and an alpha-particle spectrometer for detecting outgassing related to tectonic and volcanic events. Its magnetometer and electron reflectometer are mapping its magnetic field, while Doppler studies of its radio signal are providing maps of the lunar gravity field (or mass concentrations).[2]

Within 2 months of its launch on January 6th, 1998, definitive evidence was found for the existence of water ice trapped in the perennially-shadowed crater bottoms at the north and south poles, in an amount of several gallons per cubic yard. [3] Also at this time, the first global gravity maps were made, showing the locations of mass concentrations. Since then, the spectrometers have yielded enough data to create maps of all the sought-after elements. [4] These maps should be considerably refined during the low-orbit phase of its mission.

Lunar Prospector is a NASA Discovery Mission.[5] This new mission philosophy places an emphasis on sci-

*MRJ Technology Solutions, NASA Ames Research Center, M.S. T27A-1, Moffett Field, CA 94035

ence and "Faster, Better, Cheaper" mission design and development. Lunar Prospector exemplifies these goals. It is a small, spin-stabilized craft that uses flight-qualified, modern technologies and instrumentation to ensure results and minimize risk. The design is simple: a small, graphite-epoxy drum (1.4m x 1.2m) with surface mounted solar cells and three 2.5m masts which carry the instruments and isolate them from the bus. There is no on-board computer, although there is a microchip used to receive ground commands and control the spacecraft, and store the data collected while traversing the lunar far side.

The entire mission was realized well within a budget of $63 million, in a time frame of two years from initial hardware development to final testing. The results of this mission are important not only for basic scientific inquiry, but also have implications for the question of how much natural resources are available for the support of long-lived manned lunar bases, from which further lunar explorations could be conducted, or which could be used as a stepping stone to exploration of Mars.

DATA VISUALIZATION OVERVIEW

From the outset, the Lunar Prospector mission had the following goals: [6]

1. Obtain high-quality scientific data about the Moon's structure, composition, and resources, thereby providing insight into lunar origin, evolution, and utility for exploration.

2. Demonstrate that the philosophy of "cheaper, better, faster" can be aggressively applied to yield a rapid development, very inexpensive planetary science mission.

3. Create an innovative education and outreach program which uses novel partnering arrangements and new technologies to stimulate public interest in planetary exploration.

In regards to goal 3, part of the charter of space exploration missions is not only to return data to the mission scientists, but also to engage the public to the greatest extent possible. At the time of the mission, Web technologies had begun to mature to the point where it became feasible to present the raw data in near-real-time, virtually at the same time the mission investigators had access to it, in a way that was compatible with the "better, faster, cheaper" philosophy of the Discovery series. The World Wide Web provided a pre-existing global infrastructure through which the data could be disseminated as soon as it reached Earth. Interpreted results would, of necessity, lag behind weeks to months, but the excitement of accessing the raw datastream from a planetary explorer was now available to anyone in the world with a computer, modem, and Internet connection. While the earlier Sojourner and Pathfinder Martian expeditions had made good use of the Web[7] to post their exciting images obtained from the surface and from orbit (as well as to numerically display the telemetered craft engineering data), this was the first time that armchair space explorers the world over could connect to a continuous, graphical display of the science datastream in virtually real-time.

Java was selected as the Web technology to use because it has a practical means of automatically updating browser displays at the required 32-second intervals, while incorporating the necessary features of network connections, file reading, data manipulation, math operations, graphical data display, and graphical user interface. Other Web technologies such as CGI scripts, plug-ins, or Javascript are simply too cumbersome or too lightweight to meet all of these requirements. In addition, since Java is a full-fledged object-oriented interpreted language, it allows the use of pre-existing class libraries to construct the plots themselves. Unfortunately, as will be discussed later, Java implementations on various platforms and browser versions are not necessarily compatible or complete, thus diluting its ideal goal of "Write Once, Run Anywhere". In particular, older browsers have incomplete or ill-performing implementations of Java featuresets. This was of concern since the desire was to maximize access to computer users throughout the world, including those with low-end or outdated systems and browsers. As will be seen, some compromise was necessary here.

The data visualization ("Data Viz") pageset is one part of the Lunar Prospector Web site located at Ames Research Center (http://lunar.arc.nasa.gov). This site includes much information about the mission and science, an image archive, streaming audio and video clips about the mission and the Moon, VRML renditions of the spacecraft and parts of the Moon, and Shockwave animations illustrating some of the instrument operations. The Web server consists of a collective farm of 4 Sun U1-170s, an Enterprise-3000 which delivers the documents to the U1-170s via NFS, and part of an Enterprise-5000 which collects and processes all of the access log files.

The three main components of the Data Viz portion (craft location, science data display, and craft health display) were originally envisaged as co-occupying one page. It soon became evident, however, that both for performance reasons, and for visual space reasons, it made more sense to launch separate Java applets in separate windows for each of the myriad graphs and displays. The following sections will examine each of these three main components in detail.

SPACECRAFT LOCATION APPLETS

The spacecraft location display [Figure 1] utilizes two Java applets, one (the "Whole Moon" applet) which displays a craft icon relative to the whole Moon, and another (the "Lunar Terrain" applet) which displays a stationary craft as detailed images of the lunar terrain scroll by underneath, such as an observer tethered several meters from the craft might see. The current latitude and longitude of the craft are shown as well.

The "Whole Moon" applet displays either the near or far side (depending on craft longitude), and then updates the position of the craft icon (a transparent GIF) every second. The position is determined by opening a network connection to ephemeris tables on the Lunar Prospector Web server and a connection to a file containing the current date and time in Greenwich Mean Time, and consulting the tables to retrieve its current position. These tables are also used to update the position as long as the applet is active. Additionally, an orbital arc is drawn to indicate the craft's trajectory.

The "Lunar Terrain" applet uses lunar surface images

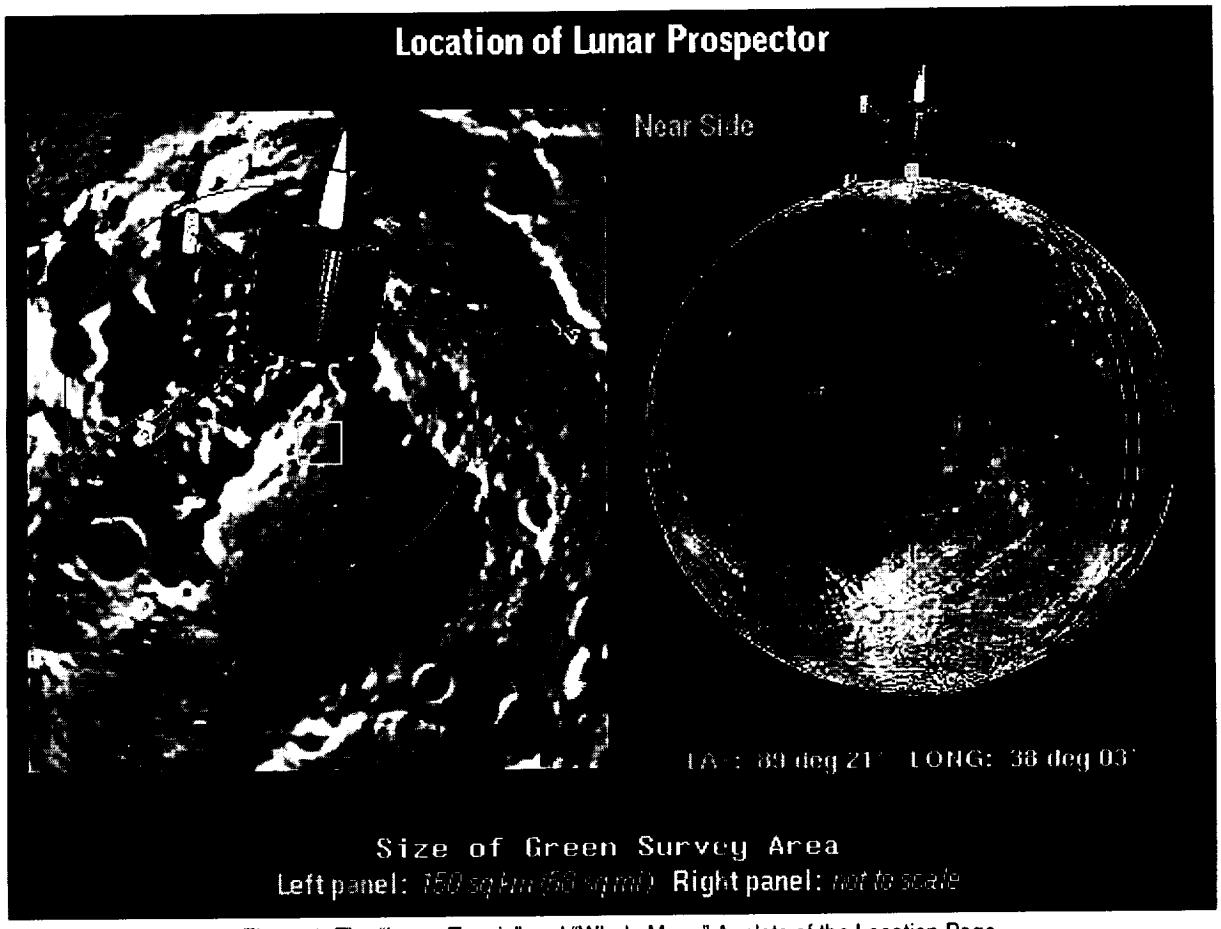


Figure 1: The "Lunar Terrain" and "Whole Moon" Applets of the Location Page

obtained from the Clementine lunar mapping mission.[8] 962 JPEG enhanced images database were used to provide global coverage of the Moon. These 962 images were subdivided into smaller images (15238 in all) for faster network loading, and stored on the Web server. On startup, the applet loads a custom metadata file describing the entire imageset, and from that constructs a lookup table for resolving positions into image files. As the applet runs, it consults the table every second to see which images will fall within the viewer window within the next minute or so, and begins loading those images so that they will be within the viewer when the time comes. The applet is capable of preloading and holding in memory over nine images at once. Nine is the minimum because some images are small enough that this many images can be simultaneously visible.

Each second the applet draws the images which are currently in view, whether they are fully loaded or not. As images fall out of view, the applet attempts to free their memory - a tricky issue in browsers not based on Java 1.1 because of the lack of a documented method for

synchronizing with the image-loading threads to stop an unfinished image from loading.

Unfortunately, there was not always a seamless continuity between the individual images of the lunar surface. Instead of allowing these discontinuities to be visible, an approach was taken in which, when the edge of an image reaches the boundary of the view window, the lunar surface image becomes stationary and the spacecraft image (a transparent GIF) instead moves toward the boundary of the view. When the position of the craft reaches the edge of the view, the new image replaces the old one and the craft image is repositioned accordingly.

SCIENCE INSTRUMENT DATA APPLETS

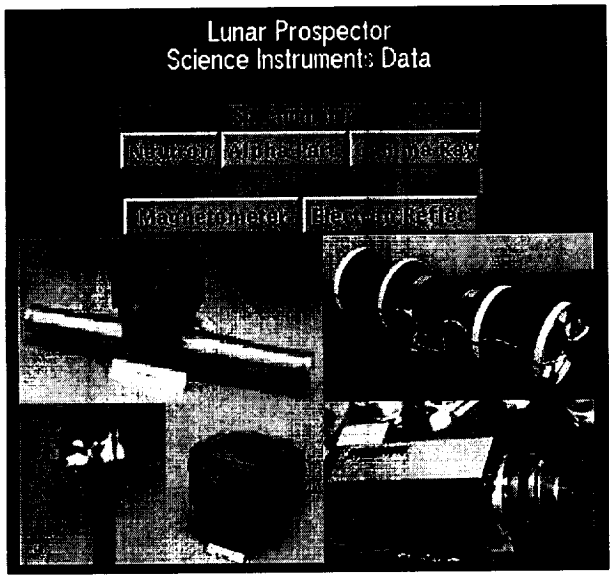


Figure 2: Science Instrument Console

When a user elects to view the near-real-time instrument data, a window is launched which houses an "instrument console" [Figure 2], whereby selecting an instrument button will 1) highlight its respective image in the accompanying imagemap, and 2) launch another window with the data graphs for that instrument. Custom JPEG images are used for the buttons, and Javascript is used to provide button behavior, since the Java user interface components (the Abstract Windowing Toolkit Toolkit, or AWT [11]) has not been implemented fully on all computer platforms. In particular, button behaviors (as the mouse traverses or clicks on the button), and button appearance attributes were not implemented in earlier browsers on the Macintosh.

When an instrument button is selected, the instrument is passed as a parameter, using the Netscape-specific LiveConnect protocol, to a Java applet which performs the imagemap highlighting. LiveConnect[9] enables communication between Java and Javascript. The display of the instrument's graphset is performed with a Java applet using a public domain Java class library called "Graph" (written by Leigh Brookshaw[10]) to construct the plots. Upon inception, the applet opens a network connection to server files containing the instrument data.

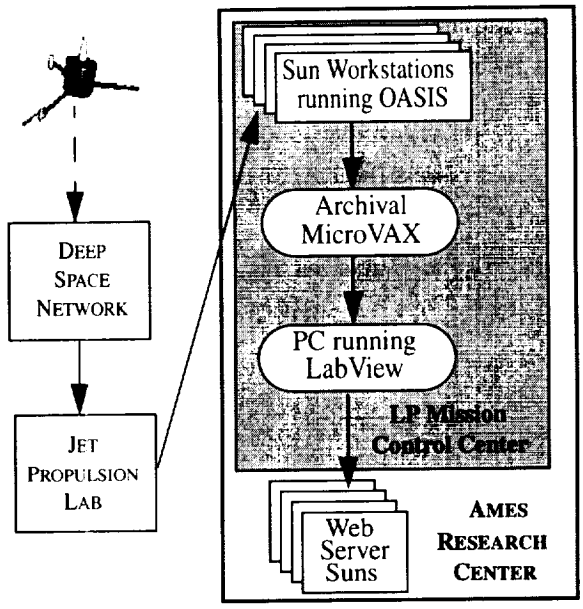


Figure 3: Datastream Path to Web Server

The path to the Web server files is as follows:

When the craft is surveying the near side, the raw datastream is collected continuously through the Deep Space Network, and routed through the Jet Propulsion Lab in Pasadena, CA to the Mission Control Center at Ames Research Center in Mt. View, CA. When the craft is on the far side, out of telemetry range, the data is stored onboard until the near side is reached again.

The SPARCstations at the Mission Control Center use the OASIS program to process and display the received data to engineers, and to merge the current near side and stored far side data to form one continuous timeline. It is then sent to an NFS-mounted archival host (a microVAX). A Perl script on this machine reads the last five minutes of the raw telemetry stream (from the SPARCstations) and saves it for use by a dedicated networked PC running customized LabView applications.

Different LabView applications display the spectrometer data and the magnetometer and electron reflectometer data to consoles for mission scientists. Modified versions of these LabView applications are used to create the ASCII data files for use by the Web. The PC is NFS-mounted on the Web server, and creates new data files in a data directory on the Web document tree every 32 seconds, when the craft is on the near side (32 sec-

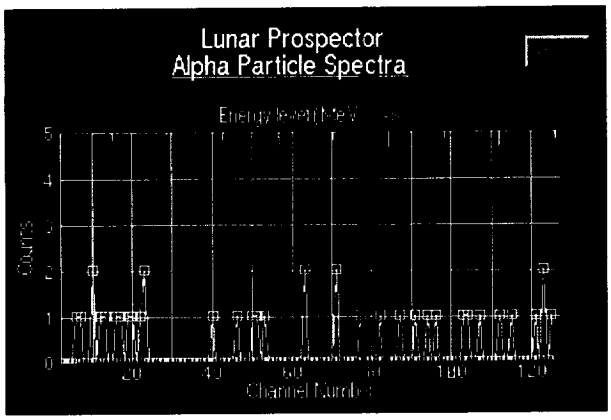
onds is the interval at which a major data frame is received). This PC can be set to select either current near side or stored far side data, and is generally set to the near side data, so this is usually what is displayed on the Web.

Every 32 seconds, the Data Viz applets re-read these files containing the latest science instrument data, fill the necessary data arrays used by the Graph package, and update their displays.

Additionally, when a user has selected an instrument to display, a separate window is launched which displays the date and time of downlink for the currently displayed data. This is accomplished with an applet that reads the file containing the craft health data for the time it was downlinked to Earth. (All of the data files for both science and health share the same downlink time.) This is also updated every 32 seconds.

The individual science instrument graphs are discussed next. In each of the instruments' windows, the title serves as an HTML link to a separate window which contains explanatory text for the graphs. This text is largely the same as is included in the following sections.

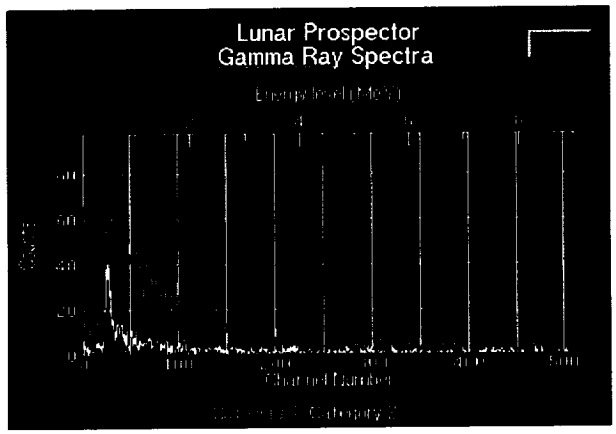
Alpha Particle Spectrometer



[Figure 4] The alpha particle spectrometer measures alpha particles emitted by radioactive gases, such as radon and polonium, which are often released by tectonic and volcanic events. The location and frequency of these gas release events help determine how active the Moon currently is and help identify one of the major sources of the tenuous lunar atmosphere.

The "Counts" (along the Y axis) are the number of particles received at the detector. The number at a given channel number is not expected to exceed more than five or so, except when encountering outgassings, when the range could be in the thousands The "Channel Number" (along the X axis) represents the varying energy levels detectable by this instrument. The plot, as for all of the spectrometer plots, simply displays the data for the current data frame (they do not scroll with time).

Gamma Ray Spectrometer



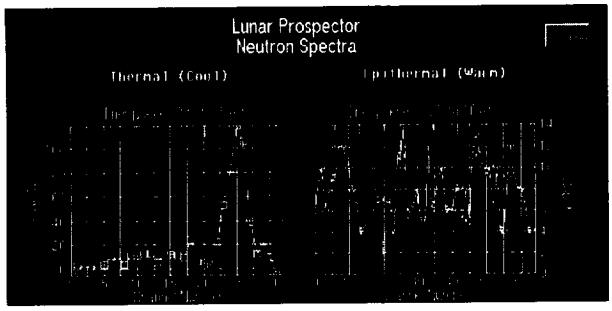
[Figure 5] The gamma ray spectrometer is used to map the composition of the lunar surface. By detecting gamma-rays from radioisotopes in the surface (as well as the gamma emissions induced by the galactic cosmic ray background), it measures the surface abundances of several key elements: Uranium, Thorium, Potassium, Oxygen, Silicon, Magnesium, Iron, Titanium, Calcium, and Aluminum.

The "Counts" (along the Y axis) represent the number of gamma rays encountered by the spectrometer. The "Channel Number" (along the X axis) represents the varying energy levels detectable by this instrument, from 0 to 9 MeV.

The two curves are the measured spectra (including noise) of the currently-surveyed area. Category 1 events are gamma-rays which only interact with the primary detector, a bismuth germanate (BGO) scintillator. Category 2 events are gamma-rays which interact with both the BGO and an anticoincidence shield (ACS) made of borated plastic. The category 2 events represent the background continuum of the lunar surface, and by subtracting this signal from the category 1 signal, it is possible to obtain more precise spectra from the surveyed area.

Many surveys of a particular area are needed to establish a reliable signal and to reduce the background noise, before the signatures of the various elements are evident. Depending on the element, anywhere from 1 or 2 months to over a year is necessary to collect an adequate sample at the equator. Collection times at the poles are much shorter, since the craft passes over the poles each orbit.

Neutron Spectrometer

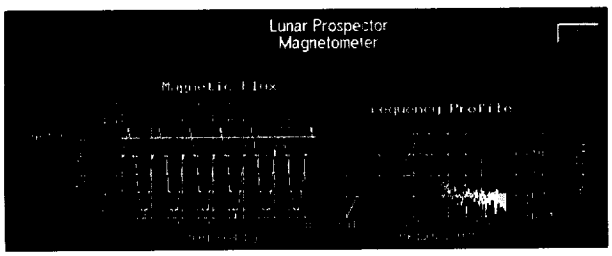


[Figure 6] The neutron spectrometer is used to detect the presence of hydrogen, and by inference, water ice (thought to reside primarily in the bottom of polar craters), by measuring the flux of neutrons. Since the craft surveys the polar regions with each orbit, within 6 weeks of mapping, evidence for water in the amount of several gallons per cubic yard of lunar soil was found in these regions.

The "Counts" (along the Y axis) represent the number of neutrons encountered by the neutron counters. The "Channel Number" (along the X axis) represents the varying energy levels detectable by the neutron counters. The thermal neutrons are in the range of 0 to .25 electron volts (eV) while the epithermal neutrons occupy the range from .25 to 10 eV.

The curves represent the measured spectra of the surveyed area. Three categories of neutrons are necessary to detect water. The epithermal (warm) neutrons are detected by a neutron counter wrapped in cadmium, while the thermal (cool) neutrons are neutrons detected by a second neutron counter wrapped in tin, after the count from the cadmium counter has been subtracted. The ratio of thermal to epithermal neutrons increases in the presence of water. Measurements of a third type of neutron, the fast neutrons, are also needed. These are provided by the gamma ray spectrometer.

Magnetometer

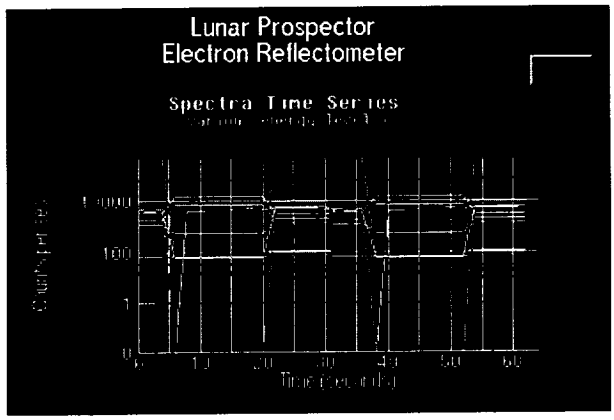


[Figure 7] The magnetometer measures magnetic fields in the vicinity of the spacecraft with a triaxial fluxgate magnetometer. The magnetic fields measured include the fields of the Earth and Sun and the much weaker field of the Moon. It is necessary to know the local field around the craft in order to properly analyze the data from the Electron Reflectometer, which will infer the magnetic field at the Moon's surface by measuring electrons reflected off of it.

The "Magnetic Flux" graph displays the magnetic flux through its three orthogonal axes. The Z axis is parallel to the dominating lines of magnetic flux from the Earth. The X and Y axes are at right angles to these lines. As the craft spins, the magnetic flux through the X and Y axes fluctuate (causing the cyclic nature of the curves). The flux through the Z axis fluctuates in much smaller cycles, since it is more-or-less parallel to the magnetic field lines. 32 seconds worth of data are displayed (each data frame contains 16 samples two seconds apart).

The "Frequency Profile" graph displays the frequency distribution of the flux, for all three axes. The peaks of the X and Y fluxes generally occur at a frequency corresponding to the spin of the spacecraft. This data is also used in analyzing the magnetic field data from the Electron Reflectometer. Since the Graph package did not support log-scaled axes, it was modified to display these for this graph as well as the electron reflectometer graph.

Electron Reflectometer



[Figure 8] The electron reflectometer is used to measure the Moon's magnetic field at the surface. It makes these measurements indirectly by measuring electrons reflected off of the surface. The magnetic field in the local vicinity of the craft, as measured by the magnetometer, is needed to analyze the electron collisions to properly infer the field at the surface.

Each of the colored curves represents a different energy level (higher colors are shown in "warm" colors). For each energy level, "Counts per second" (Y axis) is plotted against "Time" (X axis). The counts per second is a measure of the rate of electron collisions. 64 sec-

onds worth of data are displayed; the latest two data frames (each containing 32 seconds) are always shown.

SPACECRAFT HEALTH DATA APPLETS

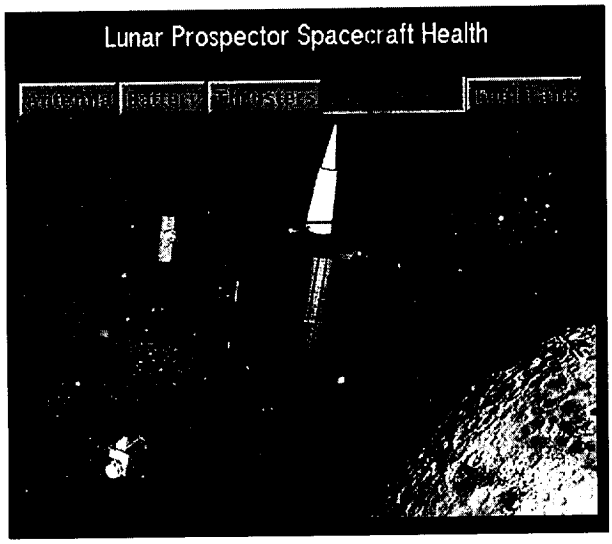


Figure 9: Spacecraft Health Console

In a manner similar to the science instrument console, when a user elects to view spacecraft "health" (engineering) data, a window is launched [Figure 9] which houses a console containing buttons for each of the major craft subsystems, and a picture of the craft. Selecting one of the subsystems will highlight its part of the craft image, and launch another window for that subsystem's graphset. The Java and Javascript mechanisms to accomplish this are similar to that of the science instrument console. Data files for the engineering data are re-read every 32 seconds, and consist of name-value keyword pairs. These keywords are parsed to find the quantities of interest for a particular subsystem.

Since only the last data frame is available at any one time (the previous ones are overwritten whenever the data file is refreshed), a mechanism is employed whereby a UNIX shell script runs on the server to accumulate the health data for the previous 20 data frames (equivalent to 10.6 minutes of data), so that when launched, the applets can start by displaying the time history of the most recent 10.6 minutes. Every 32 seconds, the graphs scroll by, always displaying the 20 most recent data frames.

As in the science instrument display, a separate window is launched which displays the date and time of downlink for the currently displayed data (if it is not already displayed).

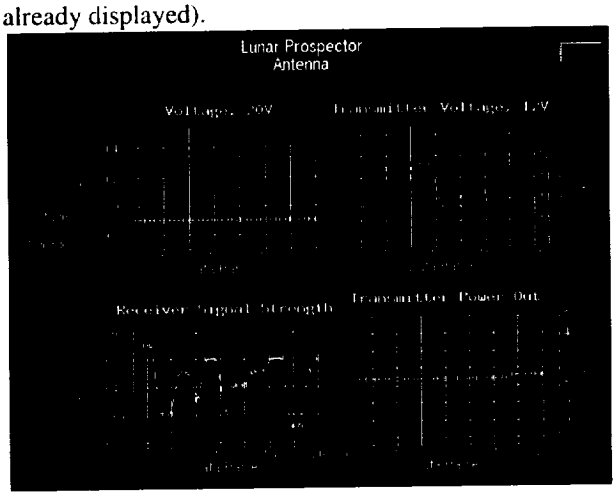


Figure 10: Antenna Graphset

Not all of the available engineering data is available for plotting. In the interests of timely development, a subset of the more critical data was selected for displaying. In particular, engineering data for the science instruments is not yet part of the available options. Currently (as of this writing), what is offered are engineering data (e.g. voltage, temperature, current, etc.) for the following subsystems: receiver and transmitter [Figure 10], battery, thrusters, solar panels, and fuel tank.

INTERPRETED DATA MAPS

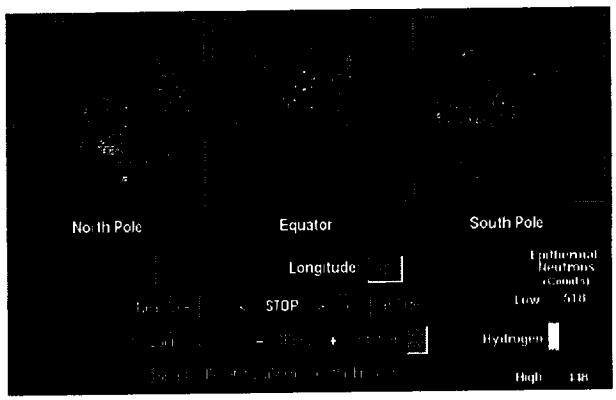


Figure 11: Javascript Animator from Data Maps Page

After enough orbits had been completed, mission investigators were able to use the spectrometer data to infer the distribution of the sought-for elements in the lunar soil. Likewise, lunar gravity maps were built using data collected from the Gravity Doppler Experiment, wherein Doppler shifts in the craft's radio transmissions were used to infer the distribution of mass concentrations. The DataViz pages were augmented to display maps of these data types: gravity, as well as tho-

rium, potassium, iron, and hydrogen (which, at the poles, is thought to be bound in water ice).

The image maps were created by producing color-coded shaded relief images from the processed data, and then overlaying these semi-transparently onto an image of the moon's visible surface. These maps were wrapped onto a sphere and animations were made of a "rotating moon" for the Web page. A Javascript animation controller (with VCR-like controls) allows for frame-stepping through the animations [Figure 11]; each frame in the animation serves as a link to a high-resolution version of itself. There are also Quicktime and MPEG animation options, and links to Mercator projections of each data map.

In addition, the Web page contains 3D VRML "globes" with texture maps for each of the data types. The VRML scenes contain preset viewpoints for the near side, far side, north pole, and south pole. The "high res" VRML option also includes a background of stars and Earth, which results in much longer loading times, but is much more visually compelling.

Figure 12: VRML Scene from Data Maps Page

RESULTS

At the outset, we defined the lowest tier of technology in our target user population as consisting of Netscape Navigator (v3 or greater) over a 28.8 KBaud modem. We realized that many Internet users had slower modems, but felt that 28.8 was probably the slowest that would give reasonable download times for the features we intended to implement. Similarly, we knew that much of the user population used other browsers such as Internet Explorer and the AOL browser. While we attempted to keep our implementation as browser-independent as possible, the Web site in general was configured to be a "Netscape site", and limited development time constrained us in some cases to resort to Netscape-specific technologies to accomplish our intended featureset (e.g. use of LiveConnect for Java/Javascript communication in the instrument and craft health panels).

Expectedly, network bandwidth was crucial in determining the speed of the applets. Sometimes it took unacceptably long for images in the "Scrolling Terrain" applet to load, in addition to the reading and plotting of data. Having high-end server engines helped, but this was still oftentimes a bottleneck.

Unfortunately, the implementation of the Java virtual machine was not uniformly well-implemented on all platforms. Specifically, Netscape 3 on the Macintosh displayed some problems, as did browsers running on Windows 3. Other browsers, such as those of AOL and WebTV, were also problematic in that they didn't load the graphics at all.

The Macintosh/Netscape 3 configuration was particularly important to us since this is what many of the NASA administrators had in Washington. The problems with this configuration included the inability to multi-task the loading of multiple images, a surprisingly slow image loader, and an inability to load images ahead of time, among other problems. Unfortunately, the "Lunar Terrain" applet depends on the ability to load images ahead of time. These problems have been circumvented by resorting to non-standard Java image-handling techniques; custom methods were provided to achieve the same functionality. Specifically, the MediaTracker is no longer used for loading successive images, and the imageUpdate method of the ImageObserver interface was overridden.

Other problems associated with the Macintosh were related to usage of the Graph package; axis labels which were oriented vertically didn't appear, for instance. Javascript was used to detect if the browser platform was a Macintosh, in which event HTML tables were used to construct the vertical axis labels.

Because Java was (during this development) a nascent technology, its implementation on older-generation browsers (which themselves are rapidly evolving) was incomplete. We needed the Java technology in order to implement the features we deemed important (active, frequent reading of datafiles, re-loading of graphics, image manipulation, etc.), but we were at a juncture in which implementations of the Java virtual machine within browsers were just beginning to accommodate these features adequately.

During this development period, the Java interpreters built into the browsers conformed to a dated version of Java (1.0 rather than 1.1). The newer browsers of the time (Netscape Communicator v4, Internet Explorer v4) were beginning to add support for 1.1 features, but the

majority of Internet users were still using v3 (or older) of these browsers, which supported only Java 1.0. Java 1.0-based browsers share the problems of not having a documented method for unloading images. Freeing of memory can be problematic for these browsers. In addition, we were prevented from using some of the advanced features of the Java 1.1, such as its enhanced GUI elements, more elegant event model, nested inner classes, etc. There is a certain frustration in being forced to use a more primitive version of a technology while knowing a more advanced one is available. Likewise, we were prevented from using the resident Java GUI builder on our development platform (SGI), because it conformed to Java 1.1, and thus wouldn't run on the browser Java virtual machines. (This was all before the release of JavaSoft's Java Project Activator, which allowed the more recent Java versions to be used in the older browsers.)

Although we inevitably disappointed some Internet citizens who were not set up with our target Web technologies, in general public response was overwhelming and enthusiastic, with teachers, science fiction writers, students, and space industry professionals expressing gratitude for being able to enjoy such immediate access to the Lunar Prospector. The Data Viz portion of the Lunar Prospector Web site accounts for, on average, about 25% to 50% of the total hits. In the first two months of the mission, for instance, the site as a whole has proven to be very popular, with well over 50 million hits recorded; an estimated 10 million visitors visited the Web site on the day of the press conference in which the discovery of water ice was announced.

CONCLUSION

This initial attempt at providing access to the data streaming from a spacecraft to the world at large via the Web was instructive in its usage of technologies (Web browsers, Java, Internet connections) that were still experiencing growing pains. Java implementations on various platforms and browser versions were far from universal. Almost all of the initial design requirements were met, however, for what we feel was a significantly large fraction of the target population of Web users, although network bandwidth and speed was, as always, an issue. As evidenced by the huge volume of hits on the Website, people relish the opportunity to have immediate access to data from a planetary exploration spacecraft. The capability for raw data access allows much more of the public than ever before to connect in a more immediate way with remote space exploration, while the displays of the interpreted results (the lunar "data maps") impart a sense of what this data can yield.

ACKNOWLEDGMENTS

The authors would like to acknowledge the encouragement and support of other members of the Lunar Prospector Web team, specifically Ken Bollinger of Recom Technologies and Larry Kellogg of Orbital Sciences Corporation, as well as the mission manager for the Lunar Prospector, Scott Hubbard of NASA.

REFERENCES

[1]Weidenschilling, S. J., Nozette, S., Shoemaker, E. M., Spudis, P., and Lichtenberg, C. L., "The Possibility of Ice on the Moon", Science, Volume 278, pp. 144-145, 1997.

[2]Kawthar-Ali, M. H., and Acharya, M., "Artificial Neural Networks for Suppression of the Dynamic Stall Vortex Over Pitching Airfoils," AIAA Paper No. 96-0540, AIAA Aerospace Sciences Meeting, Reno, Nevada, January 15-18, 1996.

[3]Morse, D., "Lunar Prospector Hits Paydirt", Ames Astrogram, March 6, 1998.

[4]various authors, collection of reports on Lunar Prospector latest results, Science, Volume 281, pp. 1475-1500, 1998.

[5]"Lunar Prospector" NASA Facts report, #FS-1997-01-01-ARC.

[6]Hubbard, G. S., Binder, Alan B., and Feldman, W., "The Lunar Prospector Discovery Mission: Mission and Measurement Description", submitted to IEEE Transactions on Nuclear Science, 1997.

[7]Mars Pathfinder Web site, http://mars.jpl.nasa.gov/default.html, Jet Propulsion Laboratory.

[8]NASA's Planetary Data System (PDS), Imaging Node (http://www-pdsimage.jpl.nasa.gov/PDS/).

[9]Netscape LiveConnect, http://home.netscape.com/eng/mozilla/3.0/handbook/Javascript/live-con.htm#996824.

[10]Brookshaw L., Java 2D Graph Package, http://www.sci.usq.edu.au/staff/leighb/graph/, Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139.

[11]Flanagan D., Java in a Nutshell, O'Reilly & Associates, Inc., Sebastopol, CA, 1996.