Navigating a Galaxy of Observations: From Frustration to Innovation

Daniel Q. Oliphant[+], Brian A. Cherinka[*], W. Michael Wood-Vasey[*], Jeffrey A. Newman[*],

Alex Labrinidis[+], Panos Chrysanthis[+], G. Elisabeta Marai[+]

University of Pittsburgh, [+]Department of Computer Science, [*]Department of Physics and Astronomy

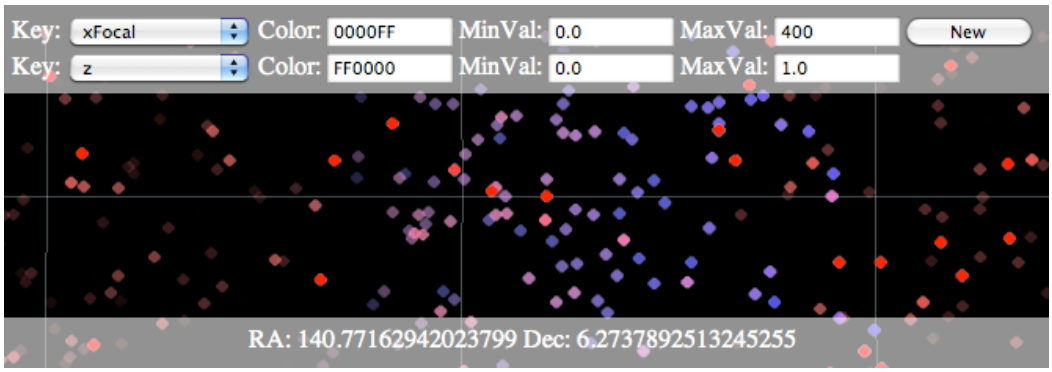

Figure 1. 831 points from the Sloan Digital Sky Survey database are visualized efficiently using PNG encryption. Two query results are overlaid on top of each other (brighter red intensities, respectively blue correspond to greater values), revealing spatial patterns in conjunction to attribute overlaps. The interface allows for flexible control of the resulting visualizations.

1 Introduction

Over the next decade the amount of information available to the typical astronomer will grow by two orders of magnitude, thanks to programs such as Pan-STARRS (Panoramic Survey Telescope and Rapid Response System) [4] and LSST (Large Synoptic Survey Telescope) [5]. However, we lack an easy-to-use and scalable way to collect, analyze and distribute anything beyond the most basic data on the thousands to billions of individual events and objects studied. Existing tools lack the capability to link data on an individual object or event together, the flexibility to allow large numbers of expert annotations and knowledge sharing, or both.

The most important challenge here is that of scalability. The sheer volume of data and meta-data (e.g., expert opinions) necessitates new paradigms to search, browse, and visualize, so that human users do not get lost in a sea of data. For example, Fig. 2 shows a typical visualization of about 60 annotations (of different types) on objects in a small part of the sky. Even at such low numbers, the icons used to represent each piece of data quickly fill the screen space and become difficult to sift through. Thus, this approach scales very poorly while astronomy research requires efficient scalability. Second, the icons seldom give information beyond what type of object they represent without actually being clicked on to bring up their full data in a pop up window. This makes it difficult to compare and contrast the attributes of similar objects. This gives the false impression that the data is homogenous. Even in this reduced example, it is very hard to answer questions based on these annotations without tools for effectively representing and querying the data.

Department of Computer Science, University of Pittsburgh, Pittsburgh, PA 15230

Email: {dqo1, bac29, wmwv, janewman, labrinid, panos, marai}@pitt.edu

Given the inherent visual nature of astronomical data, developing such tools requires tight interaction between visualization and data management, along with domain expertise. We propose a prototype system for the scalable collection, analysis and sharing of astronomy data via a client-server architecture; the prototype was developed in collaboration with astronomy experts.

2 Methods

The prototype system follows a client-server architecture, in which a web-browser client interacts with astronomy databases through a PHP-based web-server. For demonstration purposes we use a synthetically-generated MySQL database, respectively the Sloan Digital Sky Survey (SDSS) database [3]. A PHP script on the server interacts directly with the database, sending specific queries and then generating a PNG format image based on the result of the query. The PNG image is sent over a network to a web-browser based client. The client receives the image and displays it on a globe rendered in an HTML Canvas element.

Database The relational database stores the astronomy data; we assume that each tuple in the database includes Right Ascension and Declination attributes (the equivalent of latitude and longitude on the sky), indicating the location of the object stored. This assumption holds for all major astronomical databases.

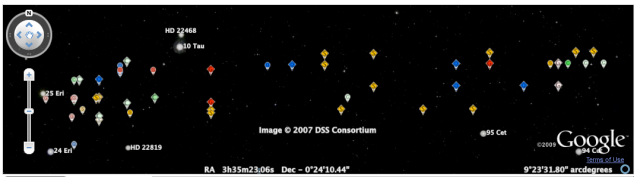

Figure 2. Traditional annotations in an early prototype system, built on top of Google Sky. The picture contains only about 60 annotations, yet the information is almost illegible.

Server The PHP script on the web server is invoked when the client application requests a new image with specific requirements. The PHP script takes the client requirements --- desired resolution of the output image, the minimum and maximum right ascension and declination values, attribute thresholds, desired color-mapping, and any other optional filters on the other parameters present in the database --- and submits them as a query to the database. Upon receiving a response from the database, the PHP script creates a new PNG image of the client-specified resolution and proceeds to draw on the image each tuple returned by the query. The right ascension and declination columns in each tuple are used to position the drawing within the image. The closer the value of the key attribute is to the maximum threshold, the brighter the color will be drawn at that point. Conversely, a lower value will result in a dimmer color. All data tuples are added to the image, which is then compressed and returned to the client application.

Client Application The client application is a web interface that consists of W3C standard html, css, and javascript. The visualization is accomplished using WebGL -- a web standard that provides a 3D graphics API implemented in a web browser without the use of plug-ins [2]. The client renders the scene to an HTML Canvas element, where the visualization scenegraph consists of a sphere with the camera at the center looking out. The generated images are mapped to the visible portions of the sphere to give the impression of looking out into space. The width and height of the viewport are used to specify the resolution of the desired image. The user generates all other elements of the query using standard HTML input controls.

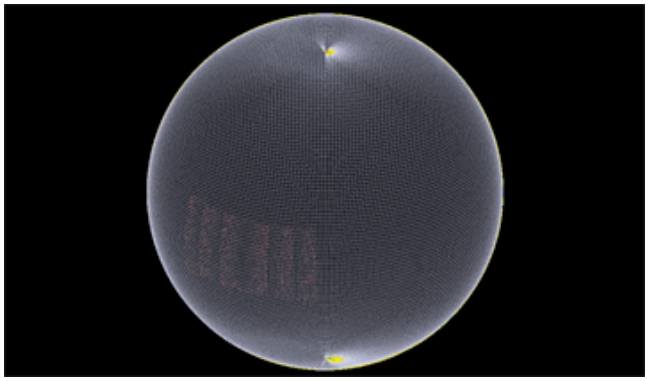

Figure 3. Client-based WebGL scenegraph consisting of a sphere and camera (outside the sphere here for demonstration purposes). The default camera is generally located at the sphere center, looking out. The sky data is drawn upon the sphere (synthetic dataset shown here in red).

3 RESULTS

The prototype system was tested on both synthetic and real datasets. Both tests focused on an area of the sky between 138 and 141 degrees Right Ascension and between 4 and 6 degrees Declination.

The synthetic dataset was stored in a database on the same server as the PHP script. The synthetic data was given a redshift parameter (a measure of how far objects are away from us) that would result in a gradient in the final image to ensure that the visualization accurately represented data patterns and did not introduce artifacts (Fig. 3). In terms of performance, the average time to query the database and generate the image was 3.51 seconds. The result mapped 10,000 data points to a 1200x800 pixel image that is 382,337 bytes in size in compressed PNG format.

The real astronomy data was taken from the Sloan Digital Sky Survey SpecObj database. The redshift of every object within the bounding box was returned and used to generate the image overlay in Fig. 1. The average time to query the SDSS database and generate the image was 2.03 seconds. The result efficiently mapped 831 data points to a 1200x800 pixel image that is 110,185 bytes in size in compressed PNG format. Rendering to a canvas of 1024 by 768 pixels, the interface ran at 35 frames per second.

4 DISCUSSION AND CONCLUSION

As indicated by Fig.1, earlier iterations of the current prototype employed as an interface the Google Sky internet browser plugin [1]. Despite the manifold capabilities of the plugin – many inherited from Google Maps -- it quickly became apparent that the Google Sky Javascript API was not flexible or versatile enough to implement a scalable solution. Later iterations used direct access to the astronomy databases and a working sky interface that was built from the ground up using WebGL [2].

In conclusion, we presented a prototype system for generating and mining scalable, graphical representations of astronomy data. The system overcomes current limitations in astronomy database visualization by following a WebGL - PHP client-server architecture; the results of querying astronomy databases are compressed as PNG images. The advantages of this approach are its versatility and visual scalability (to the pixel level), enabling the visual analysis of large datasets. The resulting versatility allows for flexible control over the visualization and the client-side scripts. Preliminary feedback from astronomy researchers shows they appreciate the versatility and visual scalability of this solution.

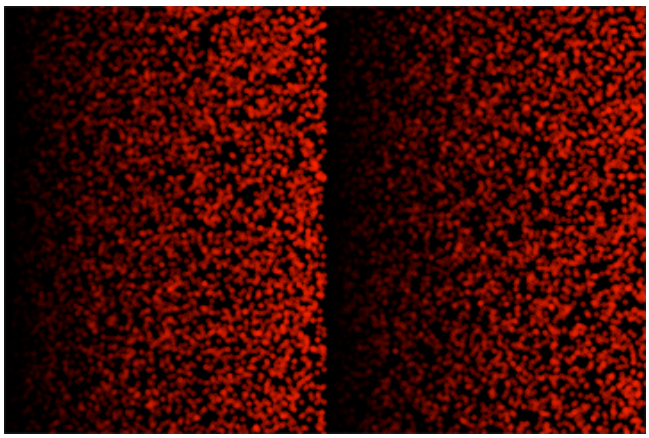

Figure 4. Overlay produced using synthetic data (for validation and performance evaluation purposes) that clearly shows the gradient pattern in the data. 10,000 data points are efficiently mapped to a PNG file.

REFERENCES

[1] *Google Earth*. http://Earth.google.com/sky.

[2] "WebGL - OpenGL ES 2.0 for the Web http://www.khronos.org/webgl/.

[3] *SDSS SkyServer DR7*. http://skyserver.sdss.org.

[4] PAN-STARRS www.pan-starrs.ifa.hawaii.edu

[5] LSST www.lsst.org