CS486C – Senior Capstone Design in Computer Science
Project Description

Project Title: A GUI interface for all-sky astronomical survey data exploration	
Sponsor Information:	David Trilling, Professor Astronomy and Planetary Science David.trilling@nau.edu Mike Gowanlock, Assistant Professor School of Informatics, Computing, and Cyber Systems Michael.gowanlock@nau.edu Northern Arizona University

Project Overview:

1.1 Background and Our Work

Understanding the physical properties of small bodies (asteroids, comets, etc.) helps us understand the formation and evolution of the Solar System and how similar we might be to other planetary systems. Additionally, creating a catalog of asteroid properties will help us prepare for the eventuality of an asteroid impacting the Earth. The primary limitation to acquiring this knowledge is simply the number of asteroids that can be observed, which in turn depends on the size of telescope and amount of data that can be collected. In other words, this science investigation is fundamentally a data science question.

Now, the big data revolution is coming to astronomy. We are entering the era of all-sky surveys, where most or all of the entire sky is imaged every night. This is a dramatic change from old school astronomy, where each individual object (star, asteroid, galaxy) was observed one at a time; work in astronomy is becoming more and more a data science – that is, using big data tools to understand the nature of the universe.

Figure 1: The center of the Milky Way as observed from the Grand Canyon.

As our data collection changes to a massive big data scale, it has become harder for astronomers to keep up with the volume of data being produced using traditional "manual" methods of analysis. In order to make progress in this big data context, astronomers must create new, automated computational analysis tools that are able to somehow pre-process this massive data stream to identify "interesting" elements, and then archive, organize, and index the results for easy access by scientists.

A great example of this big data revolution is the Vera C. Rubin Observatory's Legacy Survey of Space and Time (Figure 2). The Rubin Observatory is being built in Chile, and first light (that is, the first data from the telescope) is expected in 2022. Full science operations will commence in 2024. LSST will produce 20 terabytes of data every night, for 10 years. At this scale, no person could ever look at an entire LSST image in any detail, and all discovery will need to be carried out with computational approaches.

There is probably no part of the astronomy research community that is ready for the massive volumes of data that will be produced by LSST at this time, and we urgently need to develop big data management tools and concepts so that we will be ready when it happens. Fortunately, a smaller (but still large-scale) prototype program called the Zwicky Transient Facility (ZTF), located in San Diego County, California, is operating in an LSST-like mode, producing essentially a one-tenth scale model of the LSST data stream. This makes it a perfect testbed for developing big data management tools that explore ways to manage the large data volumes that LSST will produce.

Figure 2: The Rubin Observatory under construction in Chile.

At NAU we are ingesting the part of the ZTF data stream that is broadcasting information related to asteroids, i.e., rocky bodies in the Solar System that are tracers of the formation and evolution of the Solar System since its formation 4.5 billion years ago. This information comes from ZTF in real-time during the night and passes through a "broker" in Tucson; we then ingest this data stream here at NAU.

1.2 The Problem

We have a rudimentary GUI interface that has very limited functionality. Users of the GUI interface currently have access to coarse-grained information regarding overall database statistics (e.g., number of observations, number of objects), and basic scatterplot information when drilling-down to understand the properties of individual objects. However, the current data visualization tools are not scalable to understand the population of objects in the database, where a user may want to interact with 100,000+ data points at a given time. Furthermore, our team has developed a ranking system of interesting objects. That information needs to be displayed by a new GUI module to be developed by the team. Additionally, there is a minor problem that needs to be resolved, which is moving the current GUI to be hosted on Linux instead of Windows server. This will allow us to host the website at NAU instead of Amazon Web Services.

The main task for the team is to design a new GUI for data exploration of the population of objects in our database. Because LSST will discover many new Solar System objects, we will have 10 million+ objects that will need to be visualized across various data dimensions (think of 2- and 3-dimensional scatter plots). Interacting with plots of this size is non-trivial because the time needed to load and interact with these plots is typically intractable using standard software packages such as Matplotlib, Excel, Google plots, and others. Thankfully, there are several large-scale data visualization software packages that have been designed for our use case. The team will use these software packages, and will not invent their own tools from scratch.

1.3 Solution Overview and Details

The solution is to build an advanced data exploration module for the web interface that is able to interactively visualize a large number of data points at potentially high data dimensionalities (e.g., more than three dimensions). The team will need to research different software packages that can perform this task and determine which ones will best fit our use case. In addition, using Linux as the host operating system instead of Windows server will be required, as will be connecting to our live database. While we have a basic prototype that allows us to examine the properties of individual objects in 2 dimensions, the new data visualization module will require minor interaction with this code base. The new module will be isolated from the prior code base -- only a new button/link will need to be added to the main page that connects to the new data visualization module.

Key features are outlined as follows:

Minimum viable product

- Change current GUI to run on Linux instead of Windows server. This should require minor effort, as the current GUI was developed with open source tools. Ideally, this will be completed before Christmas break.
- Connect the GUI to the live database that is hosted at NAU. Queries will need to be sent from the machine hosting the GUI to the live database. We are currently using a snapshot of the database which is hosted on the web server itself, which is sufficient for prototyping, but is not scalable to the production environment.
- Data visualization for the population of asteroids in our database for a small number of objects. Here, think of a 2-D scatterplot where a user has <1,000 data points that are plotted for a single object (e.g., asteroid brightness as a function of

time). The user will want to interact with the scatterplot, such as zooming in and out, clicking on data points that link to other information, and saving the plots in various formats.

- Data visualization functionality for a large number of objects/data points. When interacting with a large number of data points, i.e., >1 million, performance is critical. As described above, there are many very slow data visualization software packages. Using these will make the GUI very frustrating to use, and thus intractable for our purposes. The team will need to assess the best software packages for interactive data visualization. The software will be different than that required for visualizing a small number of data points, described in the bullet point above. Anticipated issues that will need to be addressed are as follows: server vs. host side processing, supporting parallel construction of plots and visualization components, performance issues with multi-user load on the server, and other issues.

A useful system

- The GUI needs to allow users to display outlier rankings for each object. Each object will have multiple measures of outlierness, and one goal of the project is to allow users to filter objects based on various outlier scores. Outlier rankings will need to be visualized, and some of the data visualization components can be reused between the population of objects in the database (bullet point above) and visualizing objects with high outlier scores. Similarly to visualizing the population of objects in the database, this task will require minimal interaction with our current code base, and will simply require adding a new button/link to the existing GUI.
- Easy export of SNAPS analyses, e.g., graphs or data tables in a variety of forms: csv, pdf, png, FITS.
- Performance testing of the product with datasets of various sizes, specifically to show potential performance with larger scale projects like LSST. We will provide synthetic datasets for examining data visualization performance/interactivity at LSST scale.
- Automatic access to other existing data sources, such as ANTARES, JPL Horizons, etc., to pull in additional information so that information regarding interesting objects is nicely aggregated in one place, avoiding the need to laboriously visit other webpages.

Stretch goals

- Mechanism for scientists to share their analyses with colleagues, e.g., by making a particular JupyterHub notebook selectively available to other users who could then attach review/commentary.

1.4 Impact of a successful product

Advanced data visualization and analytics tools will be critical for realizing near real-time surveillance of the Solar System.

Knowledge, skills, and expertise required for this project:

- Some relational database knowledge and familiarity with SQL. This project will not involve building or maintaining a database; rather you will query our database.
- Interest in/experience with web design (both aesthetics and function). We can provide examples to emulate.
- Interest in/experience with API design.
- Working with (human) clients – both internal and external testers of our beta version.
- Some familiarity with client/server architectures.

Equipment Requirements:

- No special equipment outside of regular computer access and commonly available open-source tools and IDEs. Our database lives on a server hosted at NAU.
- Windows server will be needed to port the existing GUI to run under Linux.

Software and other Deliverables:

- A strong as-built report detailing the design and implementation of the product in a complete, clear and professional manner. This document should provide a strong basis for future development of the product.
- Complete professionally-documented codebase, delivered both as a repository in GitHub, BitBucket, or some other version control repository; and as an archive stored on Monsoon.
- A web interface and API, as described above.
- All documentation, including speed/bandwidth test outcomes.