3D Tracking Simulation of Orbiting Satellites

PATHFINDER

Doney Peters | Ivan Cisneros | Joey Hubbard | Thang Hin

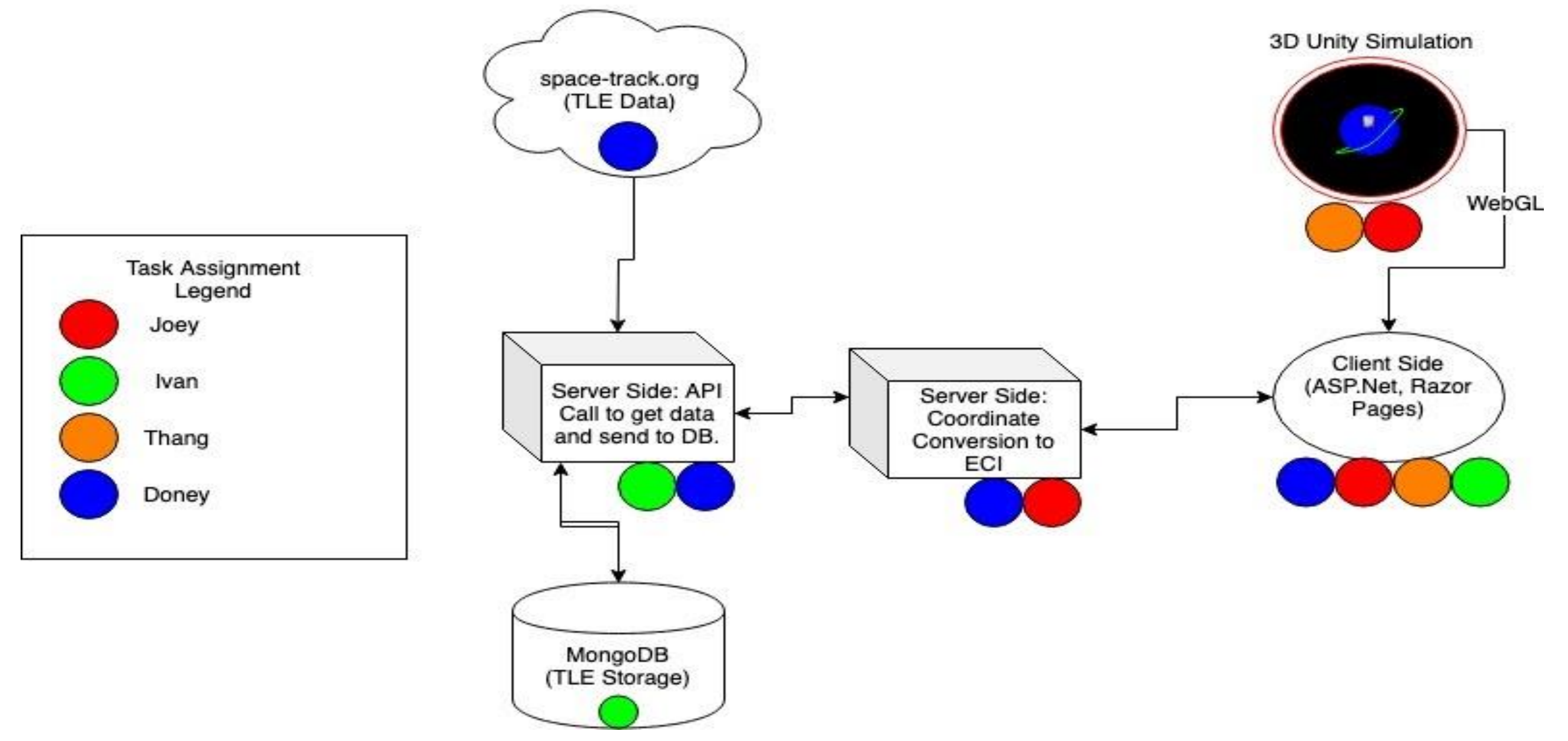
CSU Bakersfield

School of Natural Sciences, Mathematics, and Engineering

Introduction

While advances have been made in web applications that handle satellite tracking and imaging, little effort has been spent on creating highly functional and immersive simulations of the orbits of these satellites. The aim of this project is to create an efficient satellite tracking web application that will give users an accurate representation of the positions of satellites and allow users to see which satellite(s) are passing over at their present location in real-time. In addition, this web application will include a simulation using the Unity Real-Time Development Platform. Satellite tracking has only grown more complex over the years as more and more satellites are launched into space, with sources estimating at more than 1800 satellites currently in orbit. As such, current satellite tracking applications are inefficient and outdated. This project will rectify these deficiencies and will present users with a simple web-based application of the orbits taken by satellites.

Project Architecture


space-track.org (TLE Data)
3D Unity Simulation
WebGL
Task Assignment Legend
Joey
Ivan
Thang
Doney
Server Side: API Call to get data and send to DB.
Server Side: Coordinate Conversion to ECI
Client Side (ASP.Net, Razor Pages)
MongoDB (TLE Storage)

The high-level layout for this project can be broken down into four major components:

- The database storing the two-line element files needed to calculate the coordinates of the celestial objects.
- The server-side module that requests and receives specified TLE data from space-track.org and parses the file,
- The server-side module calculates the actual celestial Cartesian coordinates,.
- The front end encompassing the Unity simulation experience in addition to satellite object information.

TLE Data

Satellites are mostly tracked with computer programs using what are known as TLEs or Two-Line Elements.

TLEs are text files that contain two lines of characters and digits necessary to identify, and accurately track where a satellite may be at a specific time of day to a certain degree of accuracy.

```
ISS (ZARYA)
1 25544U 98067A   19350.13922549 -.00000461  00000-0  00000+0 0  9994
2 25544  51.6422 179.2595 0007472  40.7843  56.7759 15.50118763203489
```

An example of a two-line element.

Unity Simulation

The most important portion of the client interface, the satellite simulation, will be written using the Unity framework to render the Earth and all of the simulated celestial objects.

The coordinates of each satellite are passed via a JSON object into a javascript variable which Unity will use as a reference point to determine where to render the object relative to the Earth. From this point, Unity will also need to know the position and velocity vectors of the object as well, which the server will be able to provide using a simple AJAX request.

From these pieces of information, Unity has successfully rendered the object and properly simulated its orbit and speed.

Client/Server

-Client:
The client side, or user interface of the website uses basic HTML 5 components. Additionally, the client facing portion of the website uses the JQuery and Bootstrap libraries to provide a clean, and simple interface for users. When the Unity simulation is integrated into the webpage, it is converted to javascript via the WebGL plugin, which is a modern graphics rendering framework for the browser. During the simulation, when a user selects a satellite to view,

-Server:
The server side program is responsible for retrieving the TLE data in the form of JSON objects from space-track.org with an API request. The TLEs will be stored into MongoDB where they can be queried. When a query is made, the server will convert the TLE positional data into a series of latitude, longitude, and elevation coordinates using the SPG4 perturbation model. This result will be passed to the client side of the web page which will be displayed for users to see real time as well as be passed to the WebGL simulation to render the choses satellites position.

MongoDB Atlas

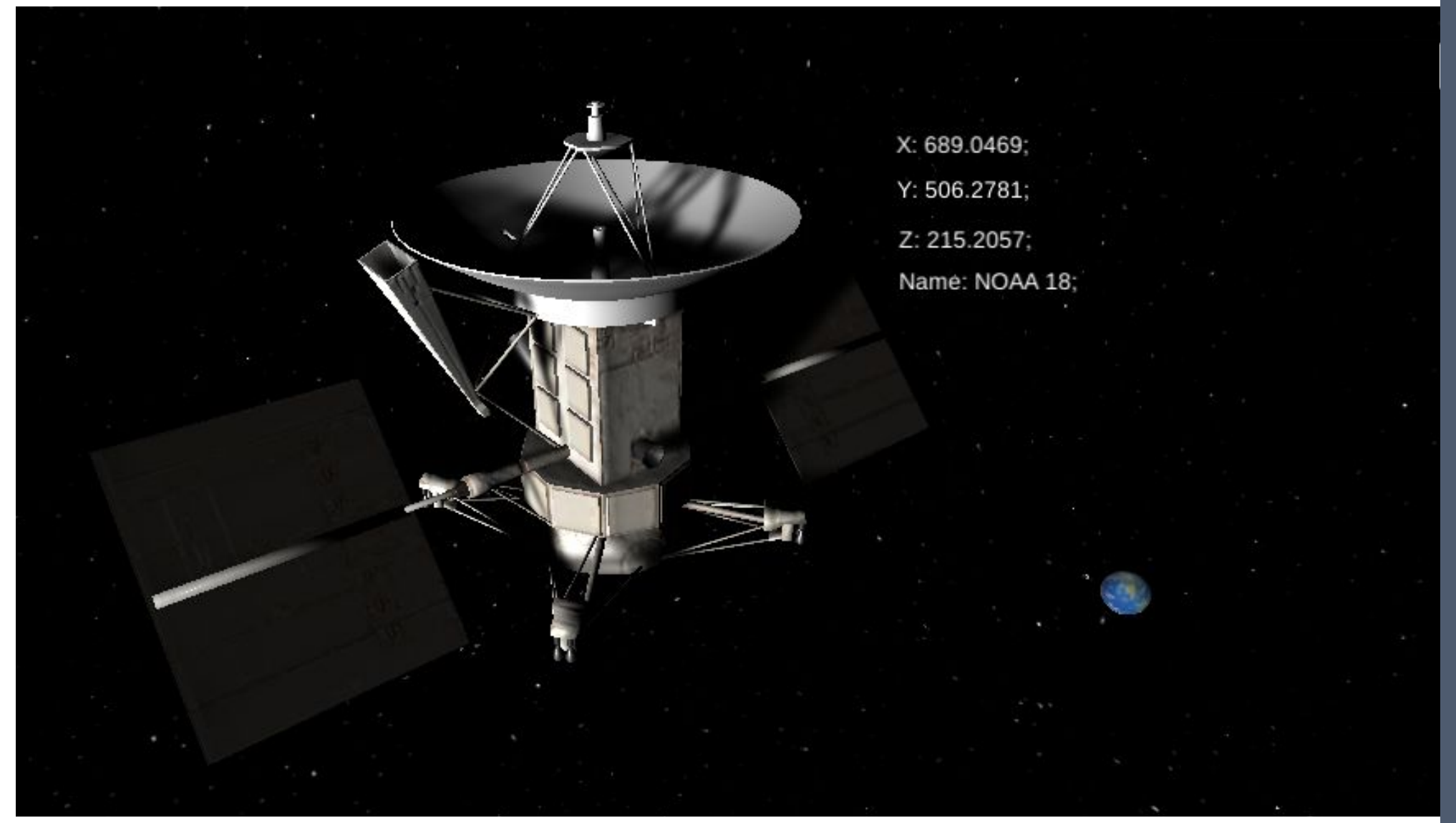
MongoDB's document model serves as a great appliance into our project as it stores data in JSON-like documents, which the TLE data gathered from Space-Track is formatted in.

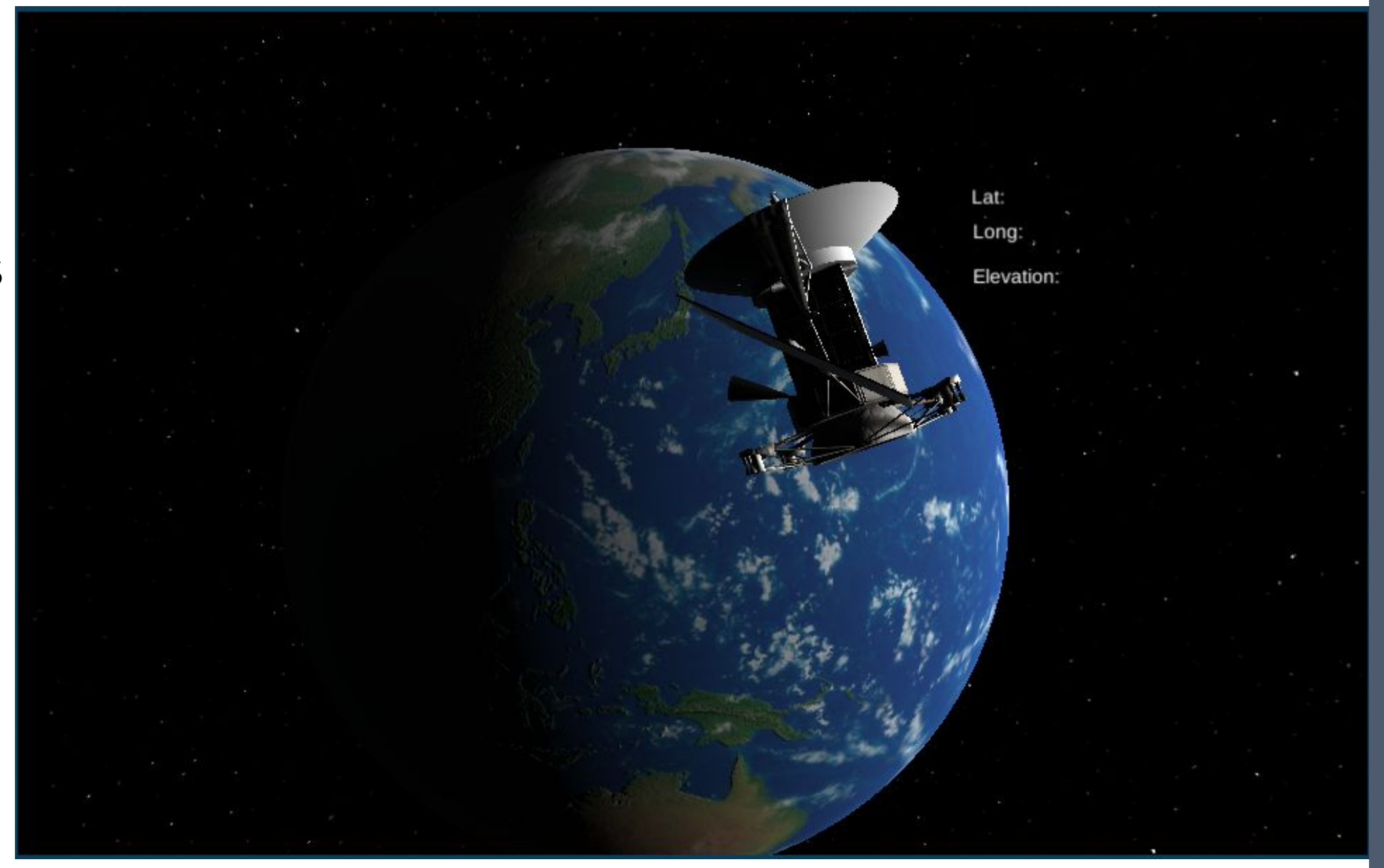
The TLE data pulled from space-track.org will be called from the server side and sent to the database, where it will be stored so that it can be displayed on the client side through the simulation in Unity.

Results

Project Pathfinder gives users an immersive experience by allowing them to select satellites from our database to view their coordinates, and navigate through space within our Unity WebGL simulator.

We were able to integrate all of the components of our project together and achieve accurate coordinates based on the updated TLE data at space-track.org. Our users will be able to see the current position of satellites in real time.





Future Goals for Project Pathfinder

We hope to incorporate the following additional features:

- Mobile App version
- Prediction of satellite overhead pass time
- Unique satellite textures
- Toggling sunlight effects
- Scaling time

Acknowledgements

Space-Track.org

- This website has an API that our web application was able to connect to in order to retrieve the TLE data of the satellites that are tracked.

CelesTrak

- Their website was an important resource for researching the structure of TLEs as well as help in deciphering the SGP4 perturbation model.

Dr. Chengwei Lei.

- Special thanks to our instructor, who has provided us with excellent feedback and guidance throughout the year.

Department of Computer and Electrical Engineering and Computer Science