CTA's ‘L’ System Visualization and Animation

Julia Finegan
Z1886085@students.niu.edu

Follow this and additional works at: https://huskiecommons.lib.niu.edu/studentengagement-honorscapstones

Part of the Computer Sciences Commons

Recommended Citation

This Student Project is brought to you for free and open access by the Undergraduate Research & Artistry at Huskie Commons. It has been accepted for inclusion in Honors Capstones by an authorized administrator of Huskie Commons. For more information, please contact jschumacher@niu.edu.
NORTHERN ILLINOIS UNIVERSITY

CTA’s ‘L’ System Visualization and Animation

A Capstone Submitted to the

University Honors Program

In Partial Fulfillment of the

Requirements of the Baccalaureate Degree

With Honors

Department Of

Computer Science

By

Julia Finegan

DeKalb, Illinois

May 11, 2024
Abstract

The Chicago Transit Authority (CTA) is a vital public transportation system for the city of Chicago and the surrounding suburbs, and all of its ‘L’ train data was recorded from March 2022 to February 2023 for this research. The main goal of this project was to create interactive/animated charts, graphs, and/or transit maps to present this raw data in a meaningful form that could help future researchers learn more about the CTA system, its patterns, and/or its unexplained inconsistencies/irregularities. A simple animation of the ‘L’ trains running within a specified time frame was created with the Python libraries Pandas, Shapely, and Pydeck utilizing the train timestamps and coordinate locations from the raw data. Many sensors that track train movement along the rails provided incorrect coordinate locations, so new coordinate locations were calculated using interpolation. For improved visual effects, each animated train was color-coded with its corresponding line color (blue for Blue line, etc.) and a background layer of all train line paths with their corresponding line colors was added. Finally, a user-interface was added to provide datetime selection, as well as line, run number, and speed filtering. This interactive train animation provides future CTA researchers with a visual of the CTA ‘L’ trains and will allow for specific analysis by showing all/single train runs at any given date and time.

Link to example run of animation: https://youtu.be/w6MIUurWdXw
Introduction

The Chicago Transit Authority (CTA) is a vital public transportation system for the city of Chicago and the surrounding suburbs. The CTA consists of daily bus and train routes, but for this project, the focus is on their rail system known as the ‘L.’ The ‘L’ consists of eight train routes, 145 stations, and 224.1 miles of track (CTA). As technology has progressed, the CTA has provided many online services to its riders that provide information such as scheduled arrival times, route durations, and in-the-moment predictions for all routes and trains. This information is created from data collected from rail sensors in conjunction with the CTA’s implementation of the General Transit Feed Specification (GTFS). GTFS is a standardized format for public transit schedule data and can be used to provide information on quickest routes, distances, and estimated arrival times (GTFS). The CTA uses the GTFS and rail sensor data to generate train arrival time predictions for each route. This raw data has been made available to the public via the CTA’s Train Tracker APIs (Overview). We recorded all CTA ‘L’ data from March 2022 through March 2023, much of which plays a large role in this project.

Definitions

- **Pydeck**: A Python package for interactive mapping and geospatial analysis
- **CurrentStationId**: A fixed point that marks a location on the train line
• **Intermediate sensors**: Sensors between stations

• **Linestring**: Stores an accurate coordinate path between train stops

• **Interpolation**: The act of inserting or interjecting an intermediate value between two other values

• **Standard Deviation**: A measure that is used to quantify the amount of variation or dispersion of a set of data values

**Project Intent**

The main goal of this project was to create an interactive tool to present the raw data in a meaningful form that could help future researchers learn more about the CTA system, its patterns, and/or its unexplained inconsistencies/irregularities within the data provided by the Train Tracker API and the corresponding documentation. This was to be done as needed through interactive and animated charts, graphs, and/or transit maps.

**CTA Train Animation**

One of the main tools that we created this semester is an interactive transit map that shows all/some of the trains running at a specified date and time. Below details the steps taken towards creating this tool.

**Choosing Pydeck’s TripsLayer.** Pydeck is a python package that provides the ability for geospatial mapping and analysis. This package provides a type of mapping called TripsLayer, which takes a path of coordinates, along with corresponding timestamps to plot any vehicle trip. Sample data, that provided exact coordinates and timestamps, was tested using TripsLayer. A simple animation was created showing a single vehicle trip throughout a city.
Adapting the CTA data to fit TripsLayer data specifications. The raw CTA data we worked with has a list of train locations that had been recorded approximately every twenty seconds. Each data entry records the line, date, time, run number, current station id, and location coordinates. For a preliminary test, we filtered the data to a thirty-minute time frame and sorted each entry by its line and run number. By doing this, we had a list of datetime entries for each train running during this time frame. Each train’s data was then sorted by time, and lists of its coordinates and corresponding timestamps were created.

Addressing issues with provided sensor coordinates. Upon initial CTA animation, it was apparent that the coordinates paths from many of the train trips did not follow the expected path based off of released CTA train line maps. Within the Blue line alone, there were many points where a train visually skipped forward to a future stop and then returned to its previous location and progressed from there. To fix this, we had to (1) retrieve accurate station coordinates from another data set and (2) interpolate the intermediate sensors to calculate new coordinate locations.

1. **Merge updated stations coordinates.** Another data file already contained updated and accurate latitudes and longitudes for each station. This was loaded and merged to replace the previous station coordinates.

2. **Interpolate intermediate sensor locations.** The original coordinates were not all accurate, but there was an additional file that contained the linestrings that existed between every station. To interpolate new coordinates, these linestrings were needed in addition to a percentage estimate of how far along the line each intermediate sensor could be found. The interpolation method used to calculate this percentage is explained in figure 1 below.
Assuming that the starting station is 0.0 and the ending station is 1.0, the sensor location would be between 0.0 and 1.0 along the linestring. We first calculated the median time it took for all the trains to travel between stations, then calculated the median time it took to travel from each station to each sensor. Figure 2 below illustrates how this method corrected the newly calculated coordinates for the Yellow line sensors. They had previously been listed as (0, 0). The red dots that are overlayed on the yellow line represent the new intermediate sensor locations.
Adding a background layer of all train line paths with corresponding line color. The initial animation was able to show the trains moving, but did not show where the tracks were located and the projected path it would take. It was also unclear what train line each train was on, due to the singular color displayed for every train. Figure 3 shows the initial view, its update with corresponding line colors, to the final view that added background mapping of each line.

![Figure 3: Color and Line Updates](image)

Adding user-friendly, interactive filtering. At this point, the animation was running all trains in a pre-set thirty-minute window. This could be changed by providing different data in the actual code. To make this more user friendly, we used Jupyter widgets to allow users to select the date, line(s), run number(s), and speed to run the animation. Upon filtering, the Pydeck animation would reload and be ready to run with the specified filters.

Final Animation View. This CTA animation provides a visual representation of the extensive data collected from March 2022 to February 2023, showing real-time movement and locations. Figure 4 below shows the final animation display window.
Another interactive tool was created to show how train speeds between stations change hourly and monthly. A pre-existing data set already provided an analysis for time in between stations aggregated hourly and monthly. The median times and 95th percentile times were taken from this data to calculate the standard deviation of train speeds/times among each segment. Taking these calculations, the CTA train lines were mapped using Pydeck’s GeoJsonLayer. Each station segment was color coded using the hourly and monthly standard deviations. The red to green
color scale was used, with a red segment having a low standard deviation and a green segment having a higher standard deviation. Figure 5 shows the standard deviation of hourly median speeds.

Specific segments details (start and end station IDs, line, and standard deviation) can be found when placing the cursor over a specific segment. This segment data can then be used to display a chart showing the median and 95th percentile times each hour to allow the user to understand the data used to generate the above map. An example use of this chart is displayed in figure 6. To use this, the user can manually input a Current (or start) station ID and train line. The median and 95th percentile time in seconds will be displayed.
Conclusion

During this semester, two interactive tools were created to visualize the CTA ‘L’ trains in a more digestible and easy-to-understand form.

(a) We have successfully animated the trains to show real-time movement and locations. The coordinates provided by the CTA Train Tracker API were updated to reflect more accurate locations of all sensors. The trains displayed can be filtered by time, line, and run number. This allows for easier understanding of what is occurring on the train lines when compared to a list of train locations that have been recorded in twenty second increments.
(b) For a broader understanding of the data, interactive maps and charts were created to display the standard deviation of train speeds when aggregated on a monthly and hourly basis.

Possible Future Improvements

The CTA ‘L’ train animation is currently able to display any or all trains running at a specified date and time. The following improvements would enhance this animation and allow for greater analysis to be done through this visualization.

- **Display the current time during each run.** This would allow the user to pinpoint any time of interest or irregularities for further analysis.
- **Allow for run-comparisons.** In addition to running a train at a given time, allow the user to select trains from different days and times to see how they differ.
References


Overview (structure, mission, values, etc..). CTA. (n.d.). https://www.transitchicago.com/about/

