CSCI 720 - Big Data Analytics

Term Project

Final Report

Nihal Parchand (np9603@rit.edu)

Rohit Kunjilikattil(rk4447@rit.edu)

Vaibhav Joshi (vj3470@rit.edu)

Date : 12/02/2019

Abstract :

Professor Kinsman wants to minimize the time taken for his commute to RIT and vice versa. Given the GPS data generated by the professor, this project aims to achieve the goal via examining different parameters of the GPS data. We initially convert the GPS files into KML and render the path generated on Google Earth. Following that, we have designed a cost function to determine the optimum path of commute. A report is prepared at the end which summarizes the results and analysis.

Overview :

This report details the analysis and results achieved across all checkpoints constituting the term project. The brief overview of each section of the report that is to follow is listed below :-

- **Converting GPS to KML :-** This section discusses the process of converting a given GPS file to a KML file which can be rendered on Google Earth

- **Cost function :-** Discusses the cost function design and the final cost function that was obtained.
- **Stop Signs :-** This section talks about the stop sign detection technique and the issues faced during detection.
- **Turns :** This section talks about the left/right turn detection technique and the issues faced during detection.
- **Program Design :** This section documents the design approach, techniques, data structures used during the implementation**.**
- **Results :** Talks about the results generated by implementing the techniques. Screenshots, tables etc. are provided for understanding.
- **Workload Distribution :** Discusses about the tasks divided amongst all teammates.
- **Discussion :** Talks about the general issues faced during the implementation and noise processing.
- **Conclusion :** This section states the summary of all the points discussed above and the learnings achieved via the medium of this term project

Converting GPS to KML :-

The process of converting the GPS files was a fairly straightforward process. Having knowledge of how a program writes another program from one of the assignments proved to be handy. Other things to keep into consideration was the conversion from GPS coordinates to latitude and longitude.As for the anomalies and data cleaning, we looked at the GPRMC and GPGGA fields and filtered the entries by following the below condition :

- For GPRMC, we only select the data points having the third field value as ‘A’ since that indicates a valid data.
- For GPGGA, only select the data points having the fix quality parameter as 1.

Looking at some of the GPS files, there were a few things that we spotted such as :-

- For few of the files, there were data points where the GPS coordinates moved by a lot of degrees. This is an anomaly since even a 1 degree of movement indicates a significant distance travelled on Earth which is practically not possible

in a few seconds. Thus, we kept track of the difference in coordinates and eliminated the ones not falling within the threshold.
- Some files had both GPRMC and GPGGA entries on one single line.
- Some files had missing data in GPRMC and GPGGA fields.

Final Cost Function:

Thus the final cost function for GPS project is :

Cost_Function = (T)/30 + 1/10 * (T1 + T2 + T3)

Where T -> duration of a trip in mins

T1 -> average time spent at stop signs in mins

T2 -> average time spent below 20mph in mins

T3 -> average time spent below 65mph in mins

Stop Signs :

- This was a tricky part to work on considering that a lot of time the car had speed which was less than that of the decided threshold. For instance, when the car was parked in the driveway, parking lot, there were a lot of false positive pins indicating that it was a stop sign. That was mitigated by using noise floor which helped in cleaning out those entries
- Additionally, there were instances where we found multiple consecutive points satisfying the stop threshold. This was because we had to take into account the possibility of a rolling stop. Hence, for a few seconds before and after the stop sign there were multiple points marked. We have replaced those points by one representative point indicating the stop sign.
- For the stop signs, speed was the feature which was used for determination. However, there were cases such as when the professor ran an errand, etc. which were falsely determined as stop signs. The noise floor did help us determine that

if a car is parked on in motion. However, for special case like the one mentioned above turned out as a false positive for a couple of instances.

Turns :

- This was another tricky computation. Since angle was the most popular and straightforward attribute available, we chose to go ahead with that.
- For computing a potential turn, we looked at the GPS file manually and rendered few coordinates on Google Earth to determine how turns work with angles.
- For a potential left turn, we observed that if an angle starts with x degrees then the GPS updates 7-8 entries where the angle reduces consistently and then we get a constant angle. This indicates that the car turned left and continued straight. The difference was found to be in the range of 85-95.
- We implemented a sliding window technique where we start with each entry and capture a window of 7-8 points. We then check if the difference is falling in the threshold. If it does than we mark it as a potential left turn and we slide the window so as to avoid multiple marking points.
- For right turns the same procedure is followed.
- In order to distinguish between right and left, a simple check is made. For right turns the angle always increases at the end and for left turn it decreases. This gives us the distinction between the two turns.

Design:-

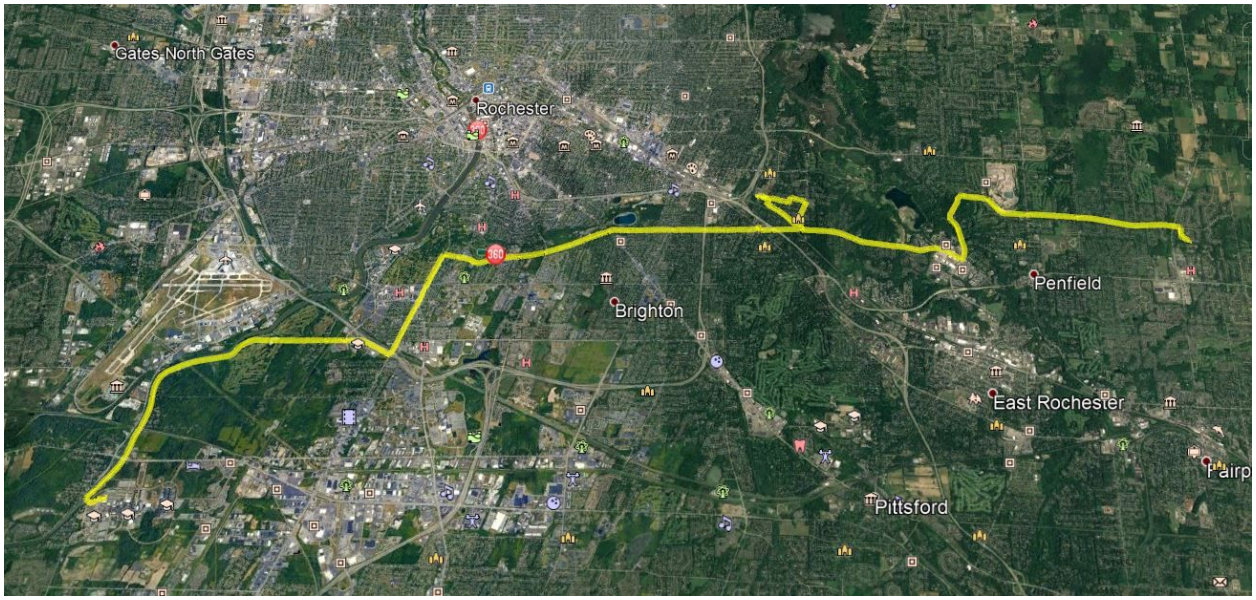
The approach for writing this program was we first looked at the different files that were provided. Then the first step we took was to calculate the noise floor using the gps data for the professor's car in a parking lot. Then we wrote the program which consisted of different functions for performing a particular task. The resultant program has the following flow :- For all the .txt files, the file first is opened in read mode. Then the latitude, longitude, speed and time of each valid point is stored in a dataframe. Then using this dataframe, we calculate the average time spent at stop signs and the average time spent between 20 mph to 60 mph. Using all these we calculate the final cost function and minimize. Each file along with is its cost function is stored in a dictionary. Then we iterate over this dictionary and find the file with the least cost function and

retrieve its corresponding key (filename). Finally, we compute the path for this file and calculate the hazards for it in a different kml.

Results :

Here are the results of the implementations.

Sample KML file rendered on Google Earth :

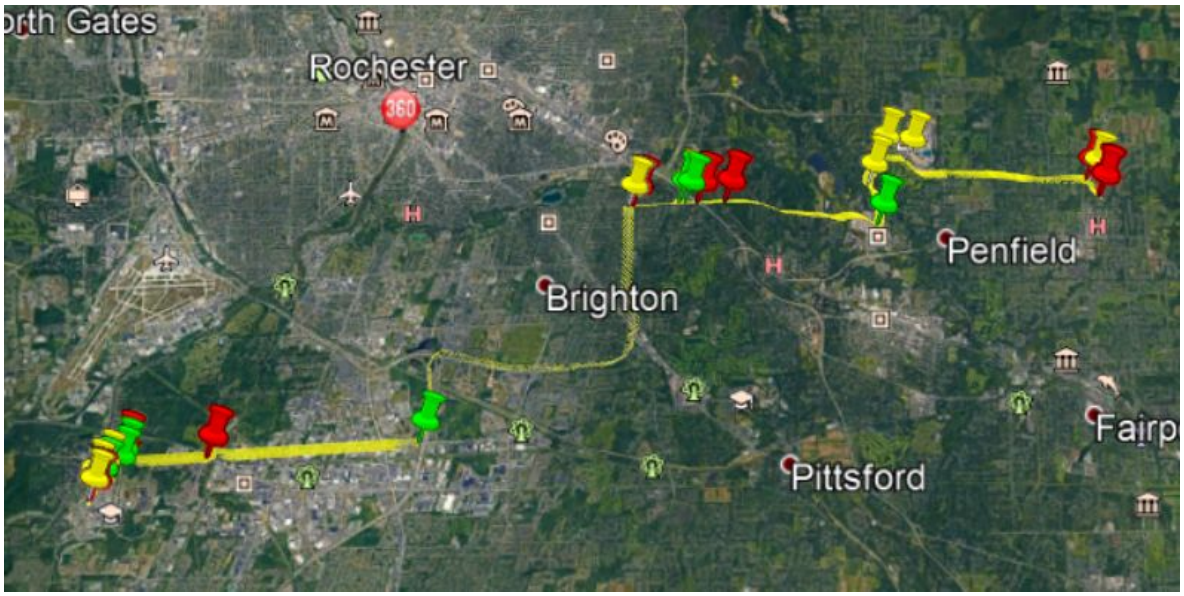


Hazards :

Red pins - Stop signs

Green Pins- Right turns

Yellow Pins - Left Turns



Generated KML File : Indicates left right turn pins

```
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
<Style id="yellowPoly">
  <LineStyle>
    <color>Af00ffff</color>
    <width>6</width>
  </LineStyle>
  <PolyStyle>
    <color>7f00ff00</color>
  </PolyStyle>
</Style>

  <Placemark>
  <description>Left turn</description>
  <Point>
  <coordinates>-77.437805,43.138332

</coordinates></Point></Placemark>
  <Placemark>
  <description>Right turn</description>
  <Style id="normalPlacemark">
  <IconStyle><color>ff00ff00
  </color>
  </IconStyle>
  </Style>
  <Point>
  <coordinates>-77.437732,43.138535

</coordinates></Point></Placemark>
```

Workload Distribution :

The workload was evenly divided among the team members :

Rohit and Nihal dealt with the cost function computation and stop signs. Vaibhav handled the turn computation and documentation.

Discussion :

Noise Floor Computation :

A file by the name of GoingNoWhereFast was provided in the project folder. The file had GPS data of professor's car in a parking lot. Using this file, we calculated the noise floor for professor's gps device. The noise floor indicated the maximum variation in the gps coordinates even when the car is parked. We calculated the noise floor by calculating the difference between the maximum coordinate in the GoingNoWhereFast and the

minimum coordinate. Thus the noise floor for both latitude and longitude was calculated. The noise floor was then used while cleaning the data to consider only those data points that show more variation than the noise, else we conclude that the car is not moving.

Conclusion :

Overall, it was a very interesting as well as challenging project. We learned a lot from this project. Learning how to deal with gps data of NMEA format, converting the points to longitude and latitude were just some of them. One of the things we implemented on the basis of our knowledge was the detection of the noise floor.