Geospatial vision & visualization: HW2

Probe Data Analysis for Road Slope

Levi Todes
Jordan Haskel

Task at Hand:

The goal of this assignment is to:

-- Map match probe point data to road link data

-- Derive the road slope for each road link

-- Evaluate the derived road slope with the surveyed slope in the link data file

Getting data

-- Map-Matching

-> The first step of this is to read and sort through both the Probe and Link Data.
-> This was done through manipulation of pandas lists such that the Probe data is sorted as follows:

Out[3]:

	ID	DateTime	SourceCode	Lat	Lon	Alt	Speed	Heading
0	3496	2009-06-12 06:12:49	13	51.496868	9.386022	200	23	339
1	3496	2009-06-12 06:12:54	13	51.496682	9.386157	200	10	129
2	3496	2009-06-12 06:12:59	13	51.496705	9.386422	201	21	60
3	3496	2009-06-12 06:13:04	13	51.496749	9.386840	201	0	360
4	3496	2009-06-12 06:13:09	13	51.496864	9.387294	199	0	360

-> And the Link Data is sorted as follows:

	LinkID	RefID	NRefID	Length	Class	DOT	SpeedCat	FSpeedLim	TSpeedLim	FLanes	TLanes	Digitized	Urban	TimeZone	ShapeInfo	CurvatureInfo	SlopeInfo	ShapeArr	SlopeArr	CurvatureArr
0	62007637	162844982	162809070	335.04	5	B	7	30	30	0	0	F	T	0.0	51.4965800/9.3862299/|51.4994700/9.3848799/	0	0	[[51.49658, 9.3862299], [51.49947, 9.3848799]]	[[0.0]]	[[0.0]]
1	567329767	162844982	162981512	134.56	5	B	7	0	0	0	0	F	T	0.0	51.4965800/9.3862299/|51.4966899/9.3867100/|51...	0	0	[[51.49658, 9.3862299], [51.4966899, 9.38671],...	[[0.0]]	[[0.0]]
2	62007648	162877732	162844982	97.01	5	B	7	30	30	0	0	F	T	0.0	51.4962899/9.3849100/|51.4965800/9.3862299/	0	0	[[51.4962899, 9.38491], [51.49658, 9.3862299]]	[[0.0]]	[[0.0]]

Pre-processing data

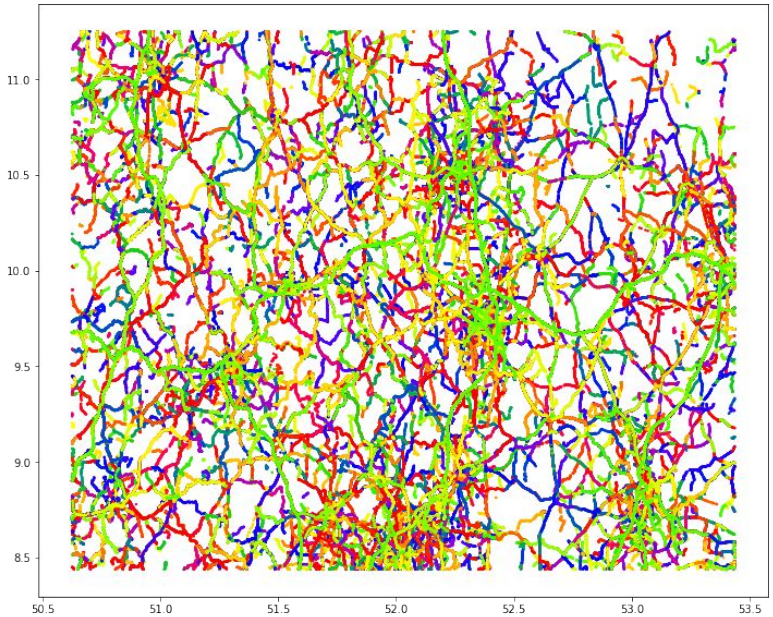
The data was cleaned and pre-processed in various steps, namely:

- Changing any temporal data into standardized Date-Time format
- Dropping duplicate probe data
- Replacing NaN values with zeros
- Splitting any textually separated data into arrays

First visualization:

-- Map-Matching

-> Visualising the raw data (here the probe longitude data (y) is plotted vs the probe latitude data (x):



Map matching:

-- Process

-> The probe points were matched to links using a haversine distance metric (https://pypi.org/project/haversine/) This was convenient as this came from an existing library.
-> Unfortunately this meant that the distance calculations were based on the end points of the links. This is fine for short links but this would be problematic for mapping probe points to long links.
-> The solution to this problem would be to implement a perpendicular distance metric for evaluating this, however time did not permit.
-> Further improvements would involve using heading and curvature information to make these calculations more robust.

Map matching:

-- Process

-> The Haversine formula used for distance is as follows:

$a = \sin^2(\Delta x/2) + \cos x1.\cos x2.\sin^2(\Delta y/2)$

$c = 2.\text{atan2}(\sqrt{a}, \sqrt{(1-a)})$

$d = \mathbf{R}.c$

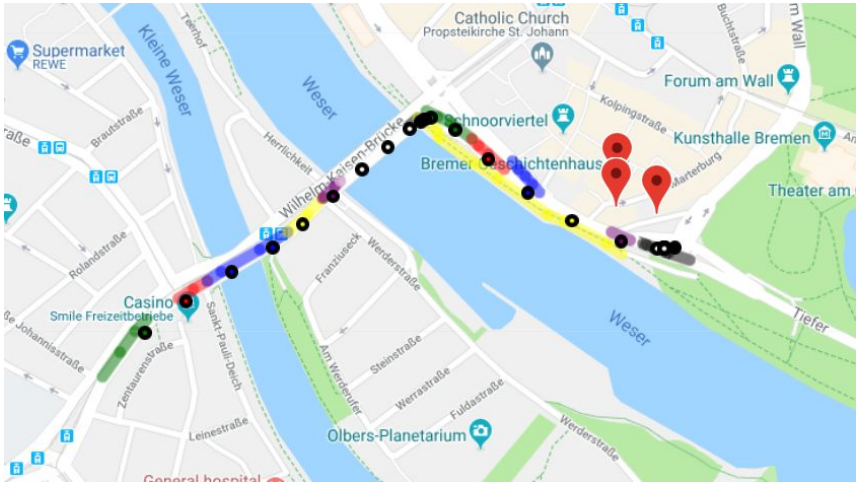
Where x = latitude; y = longitude and **R** = radius of earth

-> Inspiration for the ‘lambda x’ function used in our ‘closest()’ function (used to find map probes and links) was taken from https://github.com/kapil1garg/eecs495-map-matching

Map matching validation:

-- Map-Matching

-> The validation step is to use the Google Maps API to plot probe points and links onto a google map. The probe points are matched with their corresponding links by colour. Here you can see how a small subset of probe points were matched with links:



Gradient calculation:

-- Process

-> In order to calculate the derived slope values for the processed data, the data was grouped by the link to which they belong, after which the data was sorted first by PID and then by time so as to obtain sequential data for each probe (matched with a link).
-> the difference between altitude values of sequential data points were divided by the difference in distance of these data points. The arctan() of this value was taken to give an angle and the average slope for each link was determined.

Results:

-- CSV:

-> The resulting matched points are saved into .csv files
-> The matched link probe data is saved as matched_points.csv (this only includes the first 500 points - due to time).

->The slope results are saved as slope.csv

->The evaluated results are saved as test_comparison.txt

Problems:

Problems that we would like to fix given more time.

- Slow to calculate distances:
  - This could fixed with calculating for only on a subset of links that are closest
- Distances calculated based on end points of links:
  - not a problem with short roads but definitely a problem with longer roads
  - This could be fixed with perpendicular metric of distance
- The slope is different to the true value:
  - This could be due to many simplifying assumptions made in the calculation of the slope. This could be improved through more detailed calculations that make up for the sparsity and noise of the probe data.