Probe Data Analysis For Road Slope

Geospatial Vision and Visualisation

Haoyu Wei
Tzu-jui Liu
Yuchao Wang

Overview

Objective

To map each probe to the correct link road and calculate the slope of the link roads.

Code:

https://github.com/yuw72/Probe-Data-Analysis-for-Road-Slope

Execution:

Requirements (tested on):

python == 3.7
pandas == 1.0.3
numpy == 1.18.4
tqdm == 4.46.0

Run:

```
python3 Preprocess.py
python3 Division.py
python3 Viterbi.py
python3 Slope_cal.py
```

Our Approach

Preprocessing:

- Convert WGS 84 coordinate to Cartesian coordinates
- Generate a new csv file that replace WGS 84 coordinate to Cartesian Coordinate

Link grouping:

- We divided the world into 250*250 square groups, each group encoded with an ID and contains all the links going through the group.
- The ID is specially encoded by the x and y coordinates of the points in the group. Concretely, we divided coordinates of each point by 250, and get the floor integer and concatenate to form a special ID for the group. In this way, for any probe point with x and y coordinate, we can easily calculate the group ID it belongs to and get all the links inside the group in O(1).

```
# Calculate g_id of probe
g_lat = int(float(probe['latitude'])/250)
g_lon = int(float(probe['longitude'])/250)
g_id = str(g_lat).zfill(5) + str(g_lon).zfill(5)
```

Our Approach

Map Mapping Algorithm:

- We implemented the Viterbi algorithm to find the route with highest probability without having to list all possible routes. This method highly reduced computation time by tens of times and enabled our fast computation of results.
- Pseudocode for Viterbi Algorithm:

```
For probe points of each distinct sampleID:
      # T1[i,j] stores probability of path so far with ith link of probe point j.
      # T2[i,j] stores index of the link of the last probe that goes through ith link of probe point j and
      has max probability so far.
      # Initialize
      For each link i of first probe point:
            T1[i,0] <- p_init(link_i) * p_emis(probe_0 | link_i)
            T2[i,0] <- 0
      # Forward
      For each probe point j:
            For each link i of probe_j:
                  T1[i,j] <- max_k(T1[k,j-1] * p_trans(link_k | link_i) * p_emis(probe_j | link_i))
                  T2[i,j] <- argmax_k(T1[k,j-1] * p_trans(link_k | link_i) * p_emis(probe_j | link_i))
      # Backward
      For each reversed probe point j:
            Append link of probe_j to routes according to index of T2[i,j]
```

Our Approach

$$P(b|\mathrm{k}_i) = \frac{1}{\sigma_B\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{b}{\sigma_B}\right)^2}$$

$$P(\Delta\phi|\mathrm{k}_i) = \frac{1}{\sigma_\Phi\sqrt{2\pi}}e^{-\frac{1}{2}\left(\frac{\Delta\phi}{\sigma_\Phi}\right)^2}$$

Calculating Probabilities:

- Initial Probability:

  Given a probe point, we simply calculate the number of links “k” that lie within the same block. The initial probability is set to 1/k.

- Emission Probability:

  We model the emission probability with two Gaussian distributions, one corresponds to the angular difference between heading and the direction of a given link,and the other corresponds to the distance from probe point to a given link. The variance is set empirically. In this case, we set the variance of angular difference to 1 and the variance of distance to 40.

- Transition Probability:

  Given two links, we first determine if they are connected. If they are not connected, the transition probability is set to 0. Otherwise, we traverse all links that lie within the same block, and calculate the number of links “k” that connects to the intersection based on the node ID. The transition probability is set to 1/k.

Our Approach

Post-processing:

- Convert Json file to Dictionary;
- Transform the dictionary to make the key as linkPVID, the value as a list of sampleID.

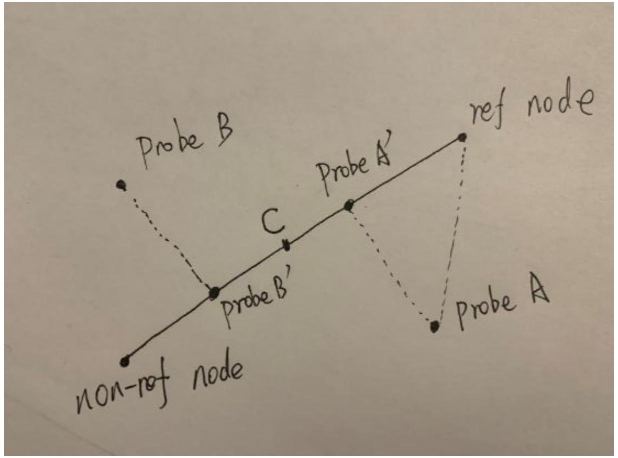
For example, the dictionary looks like:

{"3496": [567329767, 567329767, ...], "4552": [51881767, 51881672, 51881767, 51881672, ...] }

For each link data, we map its probe points into the link data like in figure 1.

Our Approach

- For each link data, we map its probe points into the link data like the figure below.
- Find two closest points to the point in the “slopeInfo” of link data.
- Calculate the slope of the point with weighted mean slope of these two closest points probe A’ and probe B’ based on their distance. The formula is RISE/RUN: (Y2-Y1)/(X2-X1) where Probe B(x1,y1) Probe A(x2,y2).



Data Examples

Convert WGS84 Coordinate to Cartesian Coordinate

sampleID	dateTime	sourceCo	latitude	longitude	altitude	speed	heading
3496	########	13	3925774	648923.8	4968302	23	339
3496	########	13	3925789	648935.7	4968289	10	129
3496	########	13	3925784	648953.6	4968291	21	60
3496	########	13	3925776	648981.6	4968294	0	360
3496	########	13	3925760	649010.9	4968300	0	360
3496	########	13	3925748	649038.7	4968304	5	89
3496	########	13	3925745	649043.7	4968306	1	288
3496	########	13	3925745	649044.4	4968305	0	310
3496	########	13	3925744	649045.3	4968304	0	274
3496	########	13	3925745	649046.8	4968304	0	226
3496	########	13	3925745	649047.8	4968304	0	201
3496	########	13	3925746	649048.1	4968304	0	182
3496	########	13	3925746	649048.2	4968304	0	232
3496	########	13	3925746	649048.4	4968304	0	202
3496	########	13	3925746	649048.4	4968304	0	199
3496	########	13	3925746	649048.7	4968304	0	179
3496	########	13	3925746	649048.8	4968304	0	184
3496	########	13	3925746	649049	4968304	0	199
3496	########	13	3925746	649049.1	4968304	0	178
3496	########	13	3925746	649049.1	4968304	0	183
3496	########	13	3925746	649049.2	4968304	0	194
3496	########	13	3925746	649049.3	4968304	0	176

Probe Point

timeZone	shapeInfo	curvatureI	slopeInfo
0	3925673.730393739/648921.7818651145/4968125.580539024|392		
0	3925673.730393739/648921.7818651145/4968125.580539024|392		
0	3925713.598342879/648835.466257237/4968105.486761032|3925		
0	3925713.598342879/648835.466257237/4968105.486761032|3925		
0	3795873.475060901/		0.00/-0.090|110.17/0.062
0	3795798.5	111.93/-0.	0.00/0.062|111.93/0.170|212.54/0.081
0	3795656.459079601/		0.00/0.081|186.28/-0.114
0	3795528.2123008794,		0.00/-0.114|140.13/-0.120
0	3795432.4	42.93/-0.0	0.00/-0.120|42.93/0.162|85.54/0.328
0	3795375.8	29.82/0.00	0.00/0.228|29.82/0.076|59.45/0.222
0	3795332.706353231/587482.0668235394/5075170.386655566|379		
0	3795332.706353231/587482.0668235394/5075170.386655566|379		
0	3795348.2509661047,		0.00/0.222|31.10/0.118
0	3795321.7	11.68/0.010145|35.05/0.006893	
0	3795271.0	22.86/-0.0	0.00/0.025|22.86/0.075|43.98/0.194
0	3795147.0	20.94/-0.0	0.00/1.338|20.94/1.638|41.28/1.499
0	3795125.5	20.65/-0.0	0.00/1.500|20.65/1.171|75.11/0.513|116.39
0	3795063.6	31.55/0.00	0.00/0.165|31.55/0.247|60.25/0.861
0	3795023.9	1.30/-0.00	0.00/0.859|1.30/0.899|19.74/1.137|30.18/1
0	3795004.2	30.86/-0.0	0.00/1.019|30.86/0.356|78.82/-0.185|109.9
0	3794936.726681821/		0.00/-0.755|13.41/-0.978
0	3794928.035600302/		0.00/-0.979|12.70/-1.161

Link Data

Data Examples

Json file generated to record the link IDs that each probe point is mapped to. The format is as follows: {"sampleID”: [linkPVID0, linkPVID1, ...], "sampleID": [linkPVID0, linkPVID1, ...] }


```
"6222": [548078324, 548078324, 548078324, 548078324, 548078324, 548078324, 548078324, 548078324, 548078324
, 548078324, 548078324, 548078324, 762432155, 762432155, 762432155, 762432155, 762432155, 762432156,
762432104, 762432104, 762432104, 762432104, 762432104, 762432104, 762432105, 762432104, 762432105,
762432104, 762432105, 762432105, 762432104, 762432105, 762432104, 762432105, 762432106, 554795744,
762561459, 762561459, 762561459, 762561459, 762561459, 762561459, 548077870, 548077870, 762561458,
762561458, 762561434, 762561434, 762561434, 762561434, 762561434, 762561434, 762561434, 762561434,
762561434, 762561434, 762561434, 762561434, 762561434, 762561434, 548136059, 548136059, 548136059,
548078837, 548078837, 548078837, 548136059, 548136059, 548136059, 548136059, 548136059, 548136059,
548078837, 548078837, 762561463, 762561463, 762561463, 762561463, 762561463, 762561463, 762561463,
762561463, 762561463, 762561463, 586509795, 586509795, 540627507, 540627507, 586514897, 586514897,
586514897, 586514897, 51790143, 51790143, 51790143, 51790143, 586514898, 586514898, 586514895, 586514895,
586514895, 586514895], "6223": [565420739, 565420739, 565420739, 565420739, 565420739, 565420739, 565420739
, 548072638, 548072656, 548072656, 548072656, 548072656, 548072656, 548072656, 548072656, 548072656,
548072630, 548072630, 548072489, 548072489, 548072590, 548072590, 548072489, 548072489, 548072490,
548072489, 548072489, 548072489, 548080686, 548080686, 548080686, 548080686, 548086177, 548086177,
548087485, 548087485, 762561468, 762561468, 762561468, 762561468, 762561468, 762561468, 548080686,
548080686, 762561440, 762561440, 762561440, 762561440, 762561440, 762561440, 762561440, 762561440,
762561439, 762561439, 762561439, 762561439, 762561443, 762561443, 762561443, 762561443, 762561445,
762561445, 762561445, 762561445, 762432150, 762432150, 762432150, 762432150, 762432149, 762432149,
762561445, 762561445, 586828231, 586828231, 586828229, 586828229, 762432118, 762432118, 762432118,
762432118, 711688562, 711688562, 565415676, 565415676, 762432118, 762432118, 711688562, 711688562,
565415665, 565415665, 711688562, 711688562, 565415670, 565415670, 762432121, 762432121, 762432121,
810164998, 810164998, 810164998, 810164998, 810164998], "7353": [764758872, 554808815, 554808815, 764758872
```

Data Examples

Slope statistics

linkPVID	GivenSlope	SlopeExperiment
716671988	-0.0038266926303175600	0.0
572245505	0.006	0.0
67948961	0.617789999417555	0.6117463801250640
565390255	0.5794424001354580	0.6029253149186340
760095050	1.086803823448920	1.1203395650036100
554728244	-0.041	0.0
548084580	1.0763815746273600	1.1301800517253500
724414857	0.06076433436532510	0.0
760063138	1.1681738117175200	1.1038542757170900
67942291	0.495	0.5668653480301430
762800686	0.40886772784899300	0.48879639724651200
565390067	0.332	0.24654309889585600
572245502	0.10950560155869500	0.0
726159763	0.2853243116688520	0.40739416441150000
51882112	-0.032263571810338200	0.09434754432812730
51867118	-0.158	0.0
548085080	0.765	0.9260416924246600
572245503	0.049	0.2128134039988280
762783421	0.36681345126644100	0.5412061941466880
586532797	1.6790465157553400	1.4998712891908400
548084552	1.6020791371106600	1.4160294785004900
83662590	0.054759134289022900	0.24475123191056800
760095044	0.9322465694063270	0.7407004315979030
586810050	1.081669227424260	0.8812495446419960
762732451	-0.009	0.20520869312997200
51865408	-0.016664815549385200	0.1987422152911990
51883668	0.6169627635987390	0.3933466030686290
548085081	0.5583024861878450	0.7945697740688340
572257880	-0.036974977366187800	0.20376074572062200
716600457	-0.038	0.2038599426997350
67942289	0.389	0.633992472233845
724414853	0.4327067474048440	0.6823607827104870
51881430	0.1713078947368420	0.42334747978088300
762732452	-0.002	0.25972086163125000
554714470	0.26315748941085900	0.0
51867774	0.11519567727335700	0.3809235393374810
67948945	0.2703241828943980	0.0
548029653	0.492	0.22155519648476100
51881767	0.1117016673344220	0.41165966253316900
51882109	-0.015	0.2874270557588150
548082797	-0.23895910035629500	0.06872280744641400
51867645	-0.134321728739295	0.17956272711874400
51882114	0.067	0.3856092279323970
67948522	0.15330071813285500	0.47443082387381
760063136	0.1790409595619390	0.5223241864100560
67942290	0.344	0.0
51874400	0.040390259369572100	0.3860700533618880
51888926	-0.28091543150848700	0.07307403223004510

Evaluation

We computed Mean Square Error of Ground Truth and our result:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

The Mean Square Error compared to the ground truth： 1.86

```
C:\Users\YUW72\Desktop\Geosp
pe>python code\Slope_cal.py
1.858658662292324
```

Fine Tuning of Our Approach
Part1

- Problem
  Calculating k-nearest-neighbor links to a certain probe point.

- Description
  Initially, we tried traversing all link data and locate links within an area based on coordinates and then find the top k links with nearest distance. But for this method, we need to calculate O(#probes * #links) times.

- Solution
  Instead of iterating all of the link data for every probe point, we partition the whole link data into several sub-regions by precomputing groups of links with specially designed group ID.

- Consequence
  The processing time for each probe point is now O(1).

Fine Tuning of Our Approach
Part 2

- Problem
  Find route with highest probability.

- Description
  Even with a fast way to compute k nearest links, if we enumerate all routes given a set of n probe points, there will be k^n different routes to compute, which is very impossible to work in reality. The enumerate method was implemented in Naive.py, by comparing the speed of two methods, we can see the huge difference.

- Solution
  Instead, we implemented the Viterbi algorithm[1] which doesn’t calculate every possible route but dynamically programmed to reduce the time complexity. Implementation details are described in previous slides.

- Consequence
  We improved efficiency by thousands of times.

Discussions

Limitations

- The accuracy of the slope we get is lower than we expect. It is probably because the probe mapping algorithm we implemented generates those errors or the approach we used to approximate slope by using weighted mean makes the accuracy lower.

Conclusion

- We have used many techniques to improve our performance in order to run all data and it proves it is very successful, but the accuracy still needs further improvement.

Reference

[1] https://en.wikipedia.org/wiki/Viterbi_algorithm#Pseudocode

[2] Luo, An, Shenghua Chen, and Bin Xv. "Enhanced map-matching algorithm with a hidden Markov model for mobile phone positioning." *ISPRS International Journal of Geo-Information* 6.11 (2017): 327.

[3] Newson, Paul, and John Krumm. "Hidden Markov map matching through noise and sparseness." *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*. 2009.