

**NANYANG  
TECHNOLOGICAL  
UNIVERSITY**  
**SINGAPORE**

## **AI6128 URBAN COMPUTING Project 2**

AY2022/2023 Semester 1

<b>Personal Particulars</b>	
<b>Student Name</b>	<b>Matriculation Number</b>
Luai Wei Jie Jonathan	G2102392J
Li Xuemeng	G2202226E
Ma Lingjie	G2202658A
Lu Pai	G2102572F

## Introduction

We completed the project as a team while each of the members focused on different essential tasks. This project is to conduct several tasks of preprocessing, visualizing and query processing trajectory data, where one important process is to map the trajectory data to a road network. The project consists of 5 tasks: Data Preparation, GPS Point Visualization, Map Matching, Route Visualization, and Route Analysis. Li Xuemeng and Lu Pai completed the data preparation (task 1 and 2); Ma Lingjie and Luai Wei Jie Jonathan focused on the map matching (task 3-4). Route analysis was completed by all members. All codes used in this project are included in the AI6128\_Project\_2.ipynb notebook, which is preferred to be tested in Google Colab.

In task 1, we explored the library *osmnx* and retrieved the first 1000 trips from the given dataset.

In task 2, we explored the library *folium* and *seaborn* to plot the given geodata of the trajectory on the map.

In task 3, we explored the library of FMM and ST-Match to map the trajectory data to the sequence of road nodes.

In task 4, we explored the library PyTrack to implement the visualization of the mapped roads.

In task 5, we calculated and visualized the road segment that is most frequent traversed or with the longest average travelling time based on the unique road segments.

## Task 1 (Data Preparation)



Figure 1: Network data map of Porto, Portugal



Figure 2: Interactive map of 'Porto' city on OpenStreetMap

The provided GitHub code implemented the road network data part of Task one. Also, the *OSMnx* tool was used all through this assignment. To show the map, we called *ox.graph\_from\_place()* function. This function can display the network map by the desired city and country name. The parameter *network\_type* was set to 'drive'.

Fig.1 was plotted by nodes connected with edges. Fig.2 shows the interactive map of Porto city. And comparing Fig.1 with Fig.2, we can see that the shape and the outlier of the two images are basically all matched.

Secondly, we retrieved the first 1000 trips from *train.csv* file by using the *dataframe* from *pandas*. Also, the 1000 sets of data were converted to a new .csv file called 'train-1000.csv'.

## Task 2 (GPS Point Visualization)

To visualize the raw GPS points of the first selected 10 trips, we kept using *dataframe* to read the 10 trajectories by taking the 'POLYLINE' column, which was a list of x and y coordinates data points of each trip. Moreover, *seaborn* library was used to manage the different colours of each trip, and *folium* library was used to help the plot.

	POLYLINE
0	[[-8.618643,41.141412],[-8.618499,41.141376],[...
1	[[-8.639847,41.159826],[-8.640351,41.159871],[...
2	[[-8.612964,41.140359],[-8.613378,41.14035],[...
3	[[-8.574678,41.151951],[-8.574705,41.151942],[...
4	[[-8.645994,41.18049],[-8.645949,41.180517],[...
5	[[-8.615502,41.140674],[-8.614854,41.140926],[...
6	[[-8.57952,41.145948],[-8.580942,41.145039],[...
7	[[-8.617563,41.146182],[-8.617527,41.145849],[...
8	[[-8.611794,41.140557],[-8.611785,41.140575],[...
9	[[-8.615907,41.140557],[-8.614449,41.141088],[...

Figure 3: Points / Nodes for each of the 10 trips

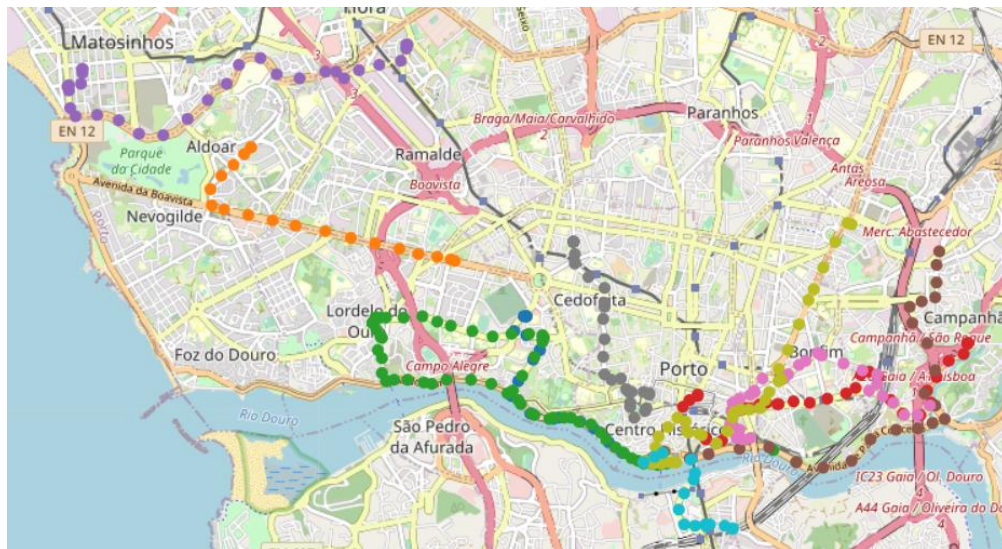


Figure 4: Results of raw GPS point visualizations for first 10 trips

Fig.4 shows the plotting result of our implementation. All ten trips managed to show on one map. The ten trajectories were shown in purple, orange, green, grey, red, deep blue, sky blue, dark yellow, brown, and pink individually.



### Task 3 (Map Matching)

In this project, we used the official library of FMM [1] and ST-Match [2] algorithms available online to perform map matching of taxi trajectories in Porto City. Since FMM is designed for small scale road network and ST-Match is for larger network, we primarily used FMM for map matching, and used ST-Match as back-up if FMM failed. Fig.5 shows one example where FMM failed but ST-Match managed to match the route.

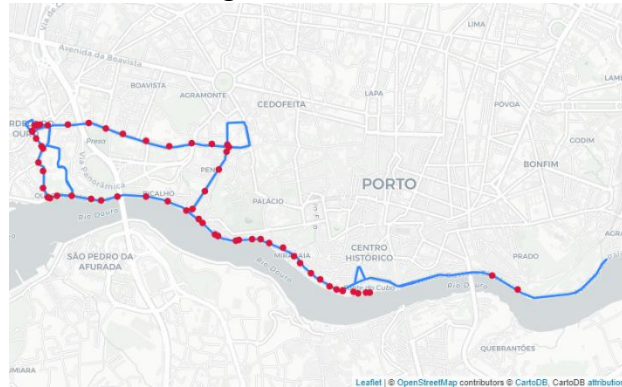


Figure 5: Example of ST-Match managed to map the route which FMM failed. Trip ID = 2

By such approach, total 682 out of 1000 trajectories have been successfully matched. There are two main causes. First, some trajectories went out of Porto City, where the road network data of Porto City queried in task 1 are not sufficient for matching, as shown in Fig.6 (left). To solve the issue, we should query the road network by block which all GPS observations should be inside. We used this query method in task 4 for better visualization of the first 10 trajectories. Second, some trajectories contain outliers GPS observations. The GPS observations were recorded every 15 seconds as per dataset specification [5], thus, GPS points that are ridiculously far to reach by taxi within 15s should be considered as outlier and be dropped during map matching.



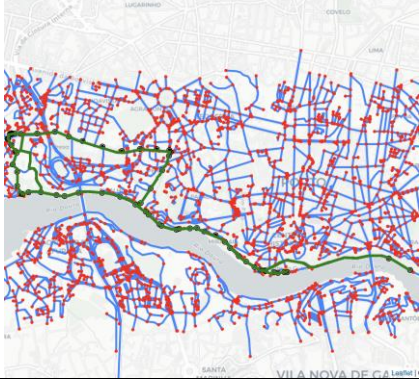

The full list of mapped routes of 1000 trajectories can be found in the attached csv file. The matched route is under column **cpath**.



Figure 6: Example of matching failures: (left) trajectory goes out of available road network data. Trip ID = 9. (right) trajectory contains outlier GPS points far away from the majority. Trip ID = 15

## Task 4 (Route Visualization)

For task 4, we implemented the visualization of the mapped roads with the help of a library package known as 'PyTrack'. From table 1, the red points and blue lines represent the nodes and edges of a specified dimension. The grey circles represent the points which are taken from the csv files and the green lines represent the estimated mapped routes taken based on the estimated points. The algorithm was also unable to map trip 10 likely due to the presence of a poor sequence of input GPS coordinates.

Trips	Mapped Roads
Trip 1	
Trip 2	
Trip 3	
Trip 4	

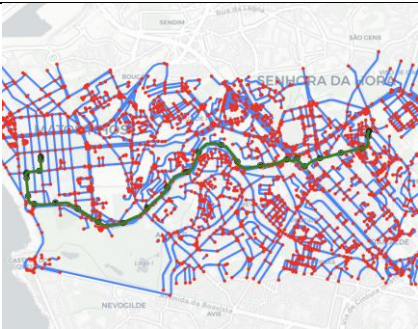


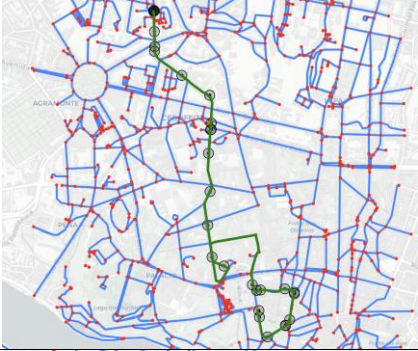

Trip 5	
Trip 6	
Trip 7	
Trip 8	
Trip 9	
Trip 10	Cannot be generated

Table 1: Visualization of the first 10 trips of the mapped roads



## Task 5 (Route Analysis)

In task 3, a hidden Markov model (HMM) is used to map GPS trajectories to routes, which firstly maps the GPS points to road segments and then generates a continuous route based on those road segments. The list of corresponding road segment of each GPS point is named **opath**, and the continuous route is named **cpath**. An **opath** could contain duplicated road segments, and segments are not necessarily connected. A **cpath** only contains unique road segments, and all segments should form a continuous path.

To find the road segments that are traversed the most often, we counted the traverse time of a road segment by the number of trajectories whose **cpath** contain the road segment. We only counted each valid trajectory once regardless the appearance times of the segment in **cpath**, as we noticed some road segments are repeatedly visited in single trip, for example, a roundabout. The top 5 most traversed road segments are shown in Fig.7. All 5 segments are at the center of Porto City.

To find the road segments with longest average traveling time, we used **opath** to roughly estimate the average traveling time of a road segment in a trajectory by counting 15 seconds per segment appearance in the **opath**. The time was then averaged over all trajectories traversed the segment. The top 5 road segments with longest traveling time are shown in Fig.8. All segments are at the road intersection, which explains the long traveling time by heavy traffics.



Figure 7: Top 5 most traversed road segments

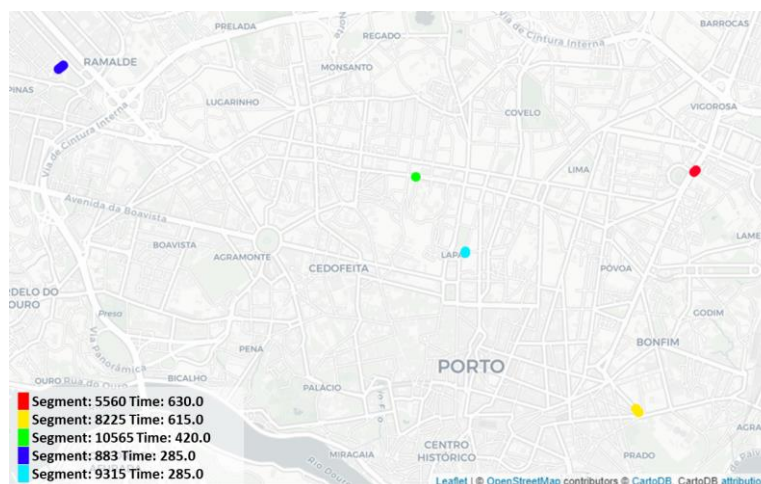


Figure 8: Top 5 road segments with longest traveling time



## Conclusion

In this assignment, our team finished all the essential tasks from 1 to 5 followed the requirements. In Part 1, we showed the road network map of the city 'Porto' and pre-processed the data file. In Task 2, we dealt with the first selected 10 trips and plotted all their nodes on OpenMap. For Task 3, we matched the taxi trajectories data to the road network from Part 1. Moreover, For the route visualisation of Part 4, we showed the results of visualization in the table above. At last, we finished the analysis about the most frequent traversed route and most average-time-costing route. And succeed in visualizing them.

## References

- 1 C. Yang and G. Gidofalvi, "Fast map matching, an algorithm integrating hidden Markov model with precomputation," *International Journal of Geographical Information Science*, vol. 32, no.3, pp. 547-570, 2017.
- 2 Lou, Yin, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. "Map-matching for low-sampling-rate GPS trajectories." In *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*, pp. 352-361. 2009.
- 3 Tortora, M. Cordelli, E. and Soda, P. (2022) "Pytrack: A map-matching-based Python Toolbox for vehicle trajectory reconstruction," *IEEE Access*, 10, pp. 112713-112720. Available at: <https://doi.org/10.1109/access.2022.3216565>.
- 4 OpenStreetMap, <https://www.openstreetmap.org/>
- 5 ECML/PKDD 15: Taxi Trajectory Prediction, <https://www.kaggle.com/c/pkdd-15-predict-taxi-service-trajectory-i/data>