Imperial College London

Department of Earth Science and Engineering

MSc in Environmental Data Science and Machine Learning

Independent Research Project
Final Report

Building a route optimisation system that takes elevation into consideration

by

Jinsong Dong

Email: jinsong.dong22@imperial.ac.uk
GitHub username: edsml-jd622
Repository: https://github.com/ese-msc-2022/irp-jd622

Supervisors:

Sesinam Dagadu
Dr. Yves Plancherel

August 2023

Abstract

Roughly 200-word.

1 Introduction

1.1 Background

Snoocode Corporation has designed a route optimization system that works offline on a smartphone and provides results 42000 times faster than the conventional method. However, this method only consider a 2D map. The next step for Snoocode is developing a method that can optimize route considering land elevation. Elevation is a very important factor for electric bikes and bicycles which are main transportation of delivery companies in Accra. To be more specific about the function of the route optimization system, suppose we have 20 locations to give delivery to, we need to find the shortest way to go so that we can deliver to every location.

For making a system to do 3D route optimization, we divide the problem into four parts:

- Data processing. Existed software for route optimization or path planing for now only use 2D road data. And the coordinates of available open source road data are 2D format. So the first important task is integrating 2D road data and raster elevation data into 3D road data.
- Building 3D road network with processed data. For a real-world road system, it is not correct to do route optimization only with coordinates(straight-line distance) of several locations. Because of the complex road conditions, it is impossible to walk in a straight line. So a network graph which can easily calculate the shortest distance of any pairs of locations is the key. With the network, the adjacent for any number of locations can be created for doing route optimization.
- Algorithms to do route optimization (Work of Rutvji Kulkarni). After the network is created, the algorithms such as genetic algorithm is implemented to do route optimization. This part of work is done by Rutvji Kulkarni(rutvij.kulkarni22@imperial.ac.uk).
- Deep learning model to do route optimization. Traditional algorithms can achieve a relatively higher efficiency for route optimization than the brute-force method. However, as the number of target locations to optimize grows, the computing time grows as well. When the number of optimized locations reaches a certain level, even mature algorithms take a long time to optimize. A deep learning model to predict the optimized route for a large amount of locations can handle this. Although it may take a very long time to train the model, SnooCODE can integrate it in the offline mobile application to do route optimization in a short time once it is trained. The model architecture used in this research is called Pointer Network developed by Google.

This project mainly focus on real world road network, but it is hard to consider all factors of roads because of its complexity. Some assumptions are raised in this research:

- The ratio of traffic to capacity is 0.5 for all road segments.
- There is no traffic light on the road.
- Vehicles travels at the maximum speed specified for each road segment.

1.2 Literature Review

The route optimisation problem in this project is a kind of Travel Salesman Problem(TSP) [1], which means given a list of cities and distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city [2]. TSP is a classic NP-hard problem, means it can not be solved in polynomial time. One example of a solution of a TSP is

shown in Fig. 1. In this example, there are 35 locations in total, and the shortest way between these locations are drawn in the line. There are $35!$ possible combination of the route totally, the 'brute-force solution' will take a very long time to get the results.

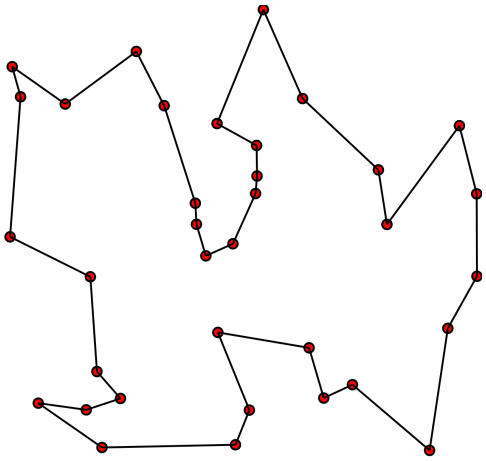

Figure 1: A solution example for TSP.

Algorithms to reduce the calculation time is essential. Many heuristic algorithms have been utilized in solving TSP[3], for example: ant algorithm, greedy method, simulated annealing, tabu search and genetic algorithm [4]. Genetic algorithm randomly samples values of the changing cells between lower and upper bounds to generate a set of combination of possible route, and choose the best one. Genetic algorithm can give a near to optimal solution in a relatively short time, but we cannot know how near it can be to the best route. Ant Colony Optimization algorithm(ACO) is one of the metaheuristic methods for solving TSP, it works like observation of real ants, and upon finding food return to their colony while laying down pheromone trails[5]. The ACO can also get a near-optimal solutions to the traveling salesman problem, it's able to find the global optimum in a finite time. Some improved ACO algorithms have also been proposed like Parallel implementation of ant colony optimization [6]. Christofides algorithm[7] utilizes Minimum spanning tree(MST) to get the approximation solution of traveling salesman problem, which guarantees its solutions will be within a factor of $3/2$ of the optimal solution length. Christofides algorithm has the limitation that it can only be applied on the instances where the distances form a metric space (they are symmetric and obey the triangle inequality) [8]. Simulated annealing(SA) algorithm is a kind of greedy algorithms. It uses randomness as part of its search for the best solution, so it is a stochastic global search algorithm. Simulated annealing algorithm is good at jumping out of the local minimum[9].

The algorithms mentioned above are mainly used on optimizing the shortest way to go, in addition to these, the work by Dijkstra based on the graph-theory also focus on the shortest path. Routing optimization has been extended into the domain of logistic, respective extensions are called VRP[10], VRP not only optimize the distance between two single points, but also optimize the distances between a series points under different criterion and restrictions. Some research extended VRP to green logistic, they optimize the fuel consumption and emissions[11]. Marc Schröder and Pedro Cabral[12] build a model to estimate the CO_2 emissions for road freight transportation and found eco-friendly route can yield up to 20% emissions reduction. Jiquan Wang[13] develop an energy consumption prediction algorithm to make route optimisation for electric vehicles.

These researches about the extensions of route optimisation did case studies on different areas like Korea and Portugal, they focused on freight trucks and electric vehicles. However, no one has done route optimisation for minimize battery consumption research for electric bikes, and no case study in Ghana yet. This project will focus on the eco-friendly route optimization system on electric bikes,

and make several case studies in Accra Ghana.

1.3 Code structure

The codes in this research are developed by Python, and encapsulated into several classes based on their functionality. The structure of the codes is as shown in Fig 2. The Road_network class is the main class for integrating 2D road data with elevation data and creating road network graph. There are also some small functions in this class, for example: 'get_data function' to get the data stored in the class, 'get_shortest_path' function to return all nodes on the shortest path between two locations, 'get_shortest_path_length' function to get the length value of the shortest path between two locations, 'weight_matrix' function to get the adjacent matrix given a list of locations.

The Visualisation class is the class for visualising the path planining between any pair of locations. This class also have some functions to show 3D road map and 2D road map.

The City class is for storing the coordinates of locations. There is a coordinate transformation function in the City class, if the user input a coordinate in latitude and longitude, it will automatically transform it into easting and northing format which is easier for calculating distance.

Besides the codes drawn in the code structure figure, there are also a class storing loss functions for calculating distance/travel time while creating network, a code script for generating dataset and a jupyter notebook containing codes to train the deep learning model.

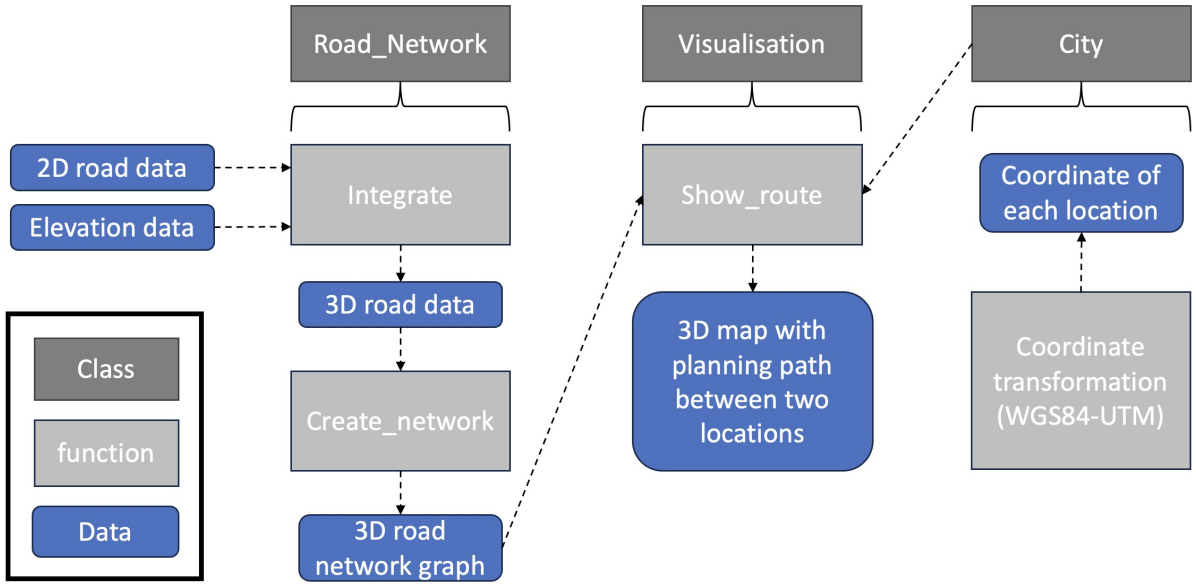

Figure 2: The code structure.

2 Methods

Technical back-end of solutions, describe if it is standalone code or an extension of a pre-existing code and ecosystem I made.
List development, operation tools, development methodologies, why.
Architectural design diagram of your solution if relevant.
Design rationale, implementation strategy, data structure, routine, verification and validation.
Algorithm, pseudo-code.
Creativity

Implementation platform, programming language, libraries.

2.1 Integrate data for 3D road data

The target city of this research is Accra, Ghana, whose range of coordinates is 5.52N to 5.68N in latitude and 0.31W to 0.01E in longitude. The mainly essential data needed in this research is the 2D road information data and the elevation data. The 2D road information data was retrieved from OpenStreetMap by Overpass API, whose last updated time is 2023-06-11T03:07:17Z. The 2D road data is a json file, the main part of the data is in key 'elements', the value of 'elements' is a list contains all road segments in the Accra area. Each element of the list 'elements' is a road segment, contains keys 'nodes', 'geometry', 'tags'. The value of the key 'nodes' is a list contains several integer numbers which refering to the unique ID of a point, The value of the key 'geometry' is a list contains the several dictionaries, each dictionary contains 'latitude' and 'longitude' which refering to the geographical coordinates of a point. In each road segment, the length of 'nodes' is equal to the length of 'geometry', because the elements in 'nodes' correspond one-to-one with the elements in 'geometry' with the same index. Connecting the series of points within each element forms the road segment. The value of the key 'tags' contains the attribute of each road segment, for example: 'highway' is the type of the road segment, 'oneway' indicate wether the road segment is one-way direction. The Fig. 3 shows the road type distribution of Accra, and Fig. 4 shows the access distribution of Accra. The road type of road segments has significant influence on the travel time calculation in later chapter as there are different speed limit for different road type. The access type determines wether a road segment can be used for delivering. The attribute 'oneway' is also considered in this research, and road segments which are 'oneway' can only be used in one direction.

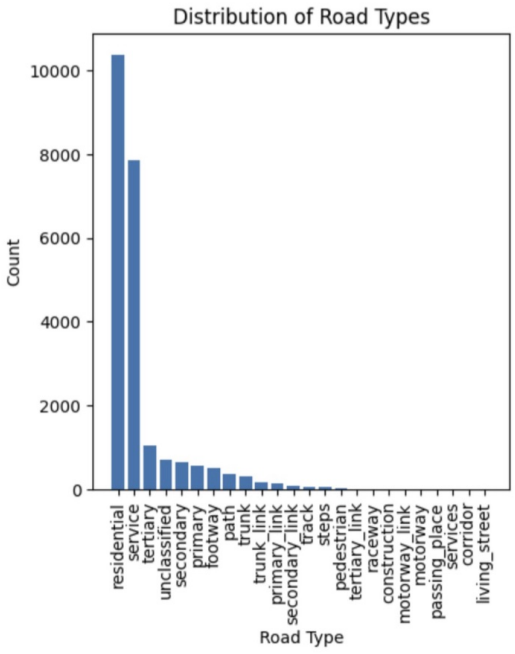

Figure 3: The road types of Accra

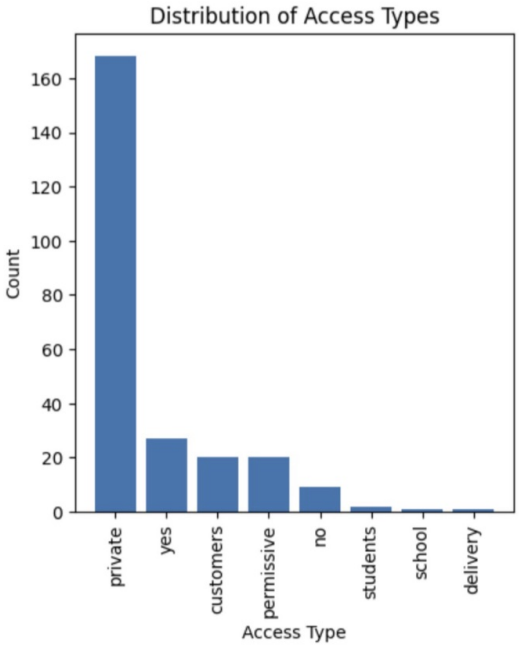

Figure 4: The access types of Accra

The elevation data is SRTM 1 Arc-Second Global retrieved from website USGS. The elevation data is in raster format, whose resolution is about 30 meters/pixel. The integration 2D road data and elevation data is to add one key 'ele' in the key 'geometry' for each point in the 2D road json file,

which represents the elevation of that point. Because the elevation data is a raster data and each pixel of elevation data is in shape of $30m * 30m$, there might be some different road points in the same grid. To make the integration of these two data more precise, interpolation is applied so that every road point is assigned its elevation value according to the values of the four pixels surround it. The schematic diagram of interpolation is shown in Fig. 5. The process of interpolation is as follows:

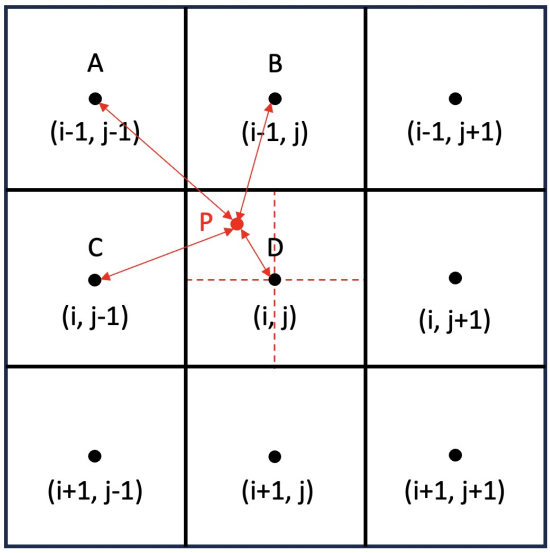

Figure 5: Illustration of interpolation

1. Get the grid length l.
2. Find the current grid the point p belongs to.
3. Find which subregion of the current grid the current point p belongs to.
4. Use the elevation data of four grids surrounding the point p to interpolate the elevation. In this diagram, $(i-1, j-1)$, $(i-1, j)$, $(i, j-1)$, (i, j).
5. Calculate the weight for the point p. For example: the weight of point A is: $w_A = 1/distance(p, A)$
6. Calculate the elevation of point p with the following equation:

$$h_p = \sum_{i=1}^{4} \frac{w_i h_i}{\sum_{i=1}^{4} w_i} \tag{1}$$

The calculated elevation of each point is added in the 2D road json file as the key 'ele' in the key 'geometry'. After this process, the 2D road data is turned into 3D road data.

2.2 Building 3D road network

To easily calculate transportation indicators like distance, travel time between any two pairs of points in Accra, building a 3D road network is essential. In this research, 3D road network graph is built by the open source package NetworkX based on the integrated 3D road data. The transportation road network graph consists of nodes and edges, nodes are points in the 3D road data and edges are formed by sequentially connecting the points of each road segment. Every node will be assigned a unique ID while creating, which is the same ID in the 3D road data file.

The one-way road is considered based on the attribute 'oneway' in the 3D road data when creating edge. If the road segment is a one-way road, the edges of this segment are created only once in the given sequence. However, if the road segment is not a one-way road, the edges of this segment are also created in reverse order. As a consequence, if the road is not one-way, their will be two directional edges in opposite directions between two nodes in the segment, if the road is one-way, their will only be one directional edge between two nodes in the segment.

When creating edges for each road segment, each edge would be assigned with some attributes, like distance and travel time. Users can also define their own attributes like battery consumption in the class 'CostFunctions'. The distance attribute is calculated based on the following equation:

$$distance(i,j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2} \tag{2}$$

where i and j indicate two different nodes, and x, y, z indicate the coordinates of a node. The travel time attribute is calculated based on the following equation:

$$T_i = T_i^0(1 + \alpha(v/c)^\beta) + Gradient_i \tag{3}$$

$$Gradient_i = \frac{G\_D\_}{S_i} \times P_i(P_i > 0) \tag{4}$$

where T_i^0 is the free travel time of the road segment; v/c is the ratio of traffic to capacity, based on the assumption mentioned before, this value is 0.5 for all road segments; α and β are two parameters ($\alpha = 0.15, \beta = 4$); $G\_D\_$ is the road segment length with a slope; S_i is the free flow speed of the road segment; P_i is the slope weight of the road segment.

The free flow speed of different road segments is on the road type, as shown in Table 1.

Road type	Speed limit (km/h)
Primary	50
Secondary	50
Tertiary	50
Trunk	50
Residential	20
Motorway	80
Service	20
Unclassified	50

Table 1: Speed limit of different road types

This network can be used to calculate some information between any pairs of points in the graph. Information including distance, traveling time. And this information can be added according to the purpose of the user. After creating the network graph, users can use it to get the best path between any pairs of locations in Accra based on the cost function they choose. The algorithm to do achieve this functionality is Dijkstra algorithm which is built-in in the NetworkX package.

In this research, I implement a function that can output the adjacent matrix based on the input of a list of locations. Each entry of the adjacent matrix represents the cost between the corresponding pair of locations. This function is the most import part for generating data to train the deep learning model for TSP problem of a real road network.

2.3 Pointer Network for TSP problem of 3D road network

The deep learning model architecture used in this research is Pointer Network[14]. The Pointer Network's structure is as shown in Fig. 6 uses one encoder to receive and transcode the adjacent matrix data into the hidden layer. Then it uses a decoder and an attention mechanism to calculate the mechanism score with the hidden layer of encoder. And the mechanism score is used to produce the pointer or index of the next location to go.

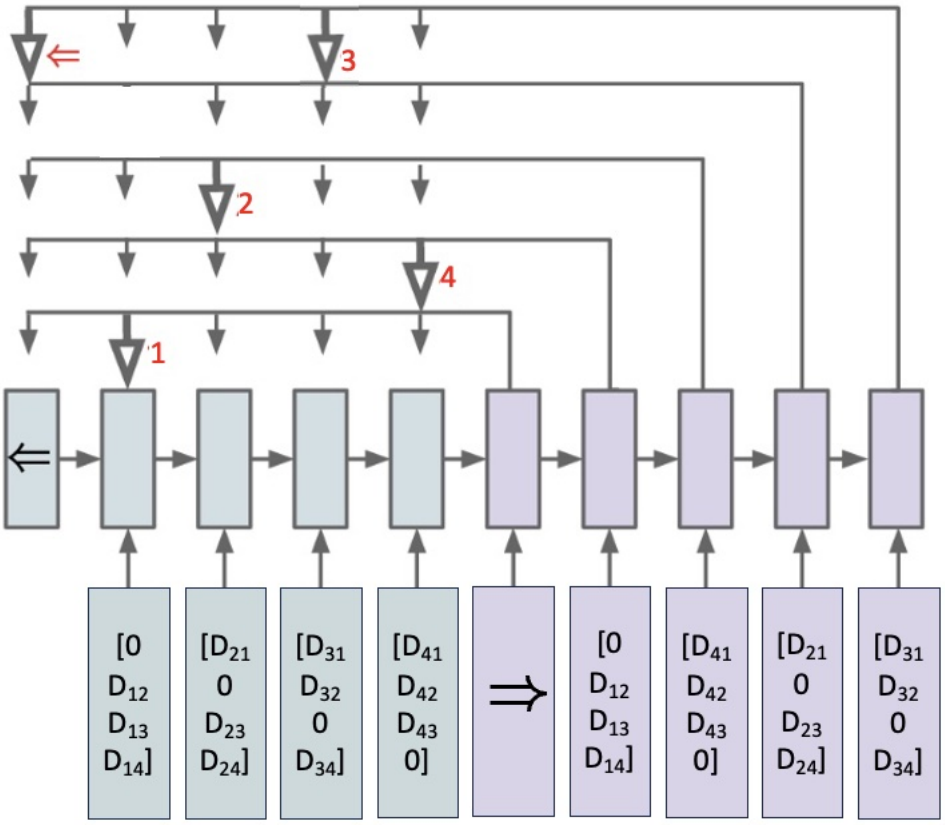

Figure 6: Illustration of pointer network

The decoder is a masked LSTM model, after a location is predicted in a time step, the pointer for this location will be masked.

To train this model, a 100k dataset is generated. Each data point contains one adjacent matrix and one label. The adjacent matrix is of five different random locations in Accra, the entry of the adjacent matrix is calculated by distance cost function, and the label is the optimized route for these five locations in a format of the list of indices. The optimized route for each data point is calculated by brute-force algorithm, so it can be sure that the route is the optimal solution.

The model is trained for 10k epochs, with the 1024 batch size, 0.001 learning rate in the first 5k epochs, and 0.0001 in the last 5k epochs.

3 Results

3.1 Performance of Integration Part

The tool developed in this research shows a great result of calculating the optimized route for any two locations in the Accra. The visualisation of the path planing done by this tool is show in Fig. 7.

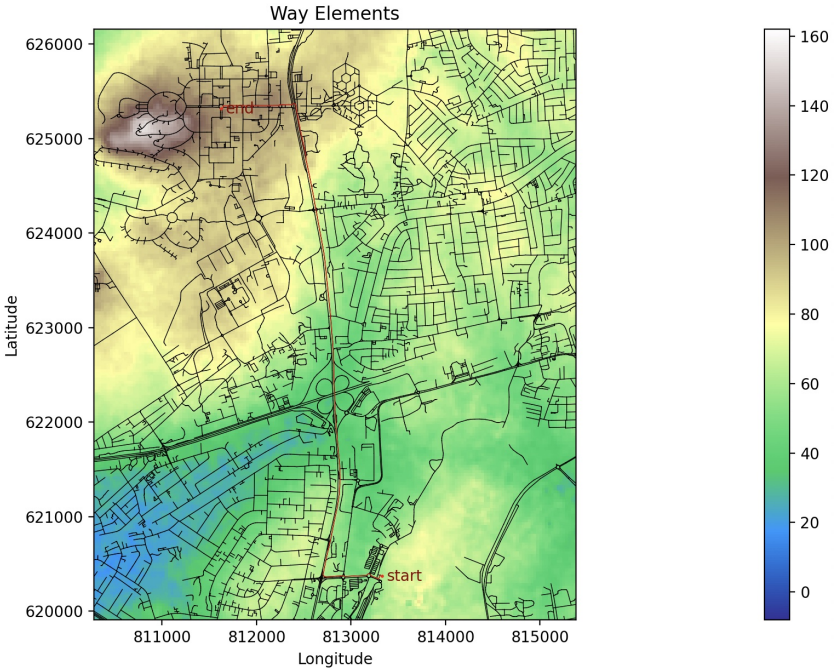

Figure 7: Path planing by this research

The corresponding path planing done by Google is shown in Fig. 8. It can be seen that the travel path are almost the same for this case.

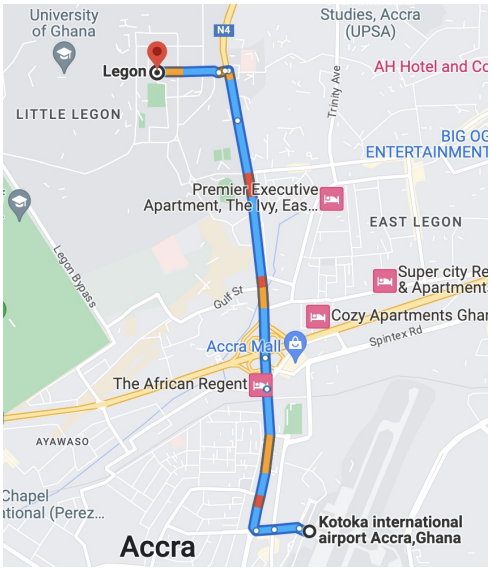

Figure 8: Path planing by Google

To verify the performance of the tool developed in this research, the optimized paths of 5 pairs of locations are compared with the path from Google map for the same 5 pairs of locations. To verify the distance cost function, walking is chosen for transportation in Google map as this is the way to get the shortest way. To verify the time travel cost function, driving with clear road is chosen for transportation in Google map as this is the way to get the least time-consuming path.

The five pairs of locations are as follows:

1. Kotoka International Airport - University of Ghana
2. Kotoka International Airport - Accra zoo
3. Kotoka International Airport - National Museum of Ghana
4. Accra zoo - University of Ghana
5. Accra zoo - National Museum of Ghana

case	distance	distance(Google)	Relative Error	time	time(Google)	Relative Error
1	6.55	6.5	0.77%	9.18	9	2.0%
2	6.94	7.7	9.9%	10.48	15	30.1%
3	7.5	7.4	1.4%	9.53	13	26.7%
4	5.38	6.8	20.1%	9.53	22	56.6%
5	9.96	9.7	2.7%	13.8	21	33.9%

Table 2: route results compare with Google

The compared results are shown as Table 2. As we can see, the mean relative error of distance is 6.97%. This is a relatively good result, means that this tool can find the shortest path in most cases. But there still has some cases that are not the shortest way like case 5. The reason maybe that this tool only find the effective route of road segments with the 'highway' tag, so some road segments without the 'highway' tag may be ignored, the tool has to skip these road segments and use some longer road segments when finding the best route. What's interesting is that there are some paths found by this tool shorter than the results from Google. This maybe because there are some road segments which can not be accessed actually but has no 'access' tag in the json file, so the tool make it able to be accessed by default. And these road segments can be a shortcut which can make the path shorter.

For the travel time part, the tool's result seems to not good enough. The average relative error is 29.86%, which is relatively high. It can be seen from the table that most travel time calculated by the tool is smaller than travel time from Google Map. There might be two reasons for that. The first reason is the vehicle speed of each road segment in this tool is the max speed limit. Although the travel time from Google Map is also calculated by the condition of clear road, the vehicle speed for calculating travel time does not reach the max speed limit in the Google Map. The second reason is that Google map considers how long it takes to wait at a traffic light, even though the condition of roads is clear. Consider these two factors together, the time calculated by the tool can be much smaller than the travel time from Google map. The travel time from Kotoka airport to the university of Ghana seems very close to the travel time from Google Map. The reason for this might be the route from Kotoka airport to university of Ghana is mainly composed of trunk road segments, and there are few traffic lights on the trunk roads. Although the travel time calculation might have deviation from the real-world travel time, the path planing by the tool is with reference significance.

3.2 Performance of Pointer Network

To evaluate the performance of the trained Pointer Network, two criterions are chosen. The first criterion is accuracy. The correct answer should make the values and positions of predicted indices all correct. The second criterion is the route length ratio, which means the ratio between the total length of the predicted route and of the true optimized route. The smaller the route length ratio is, the higher performance the model has, and the best route length ratio should be 1.

A test dataset with size of 10k is created for verify the performance of the model. The result shows that the accuracy of the model is 46.7%, and the mean route length ratio is 1.09. The performance of the pointer network with adjacent matrix as input shows that the model can predict almost the

best optimized route. Although the accuracy of the model is less than 50%, the average route length ratio is very close to 1. The main reason to cause the error, is that there might have two points that are very close to each other or the distance of the two points is close to 0 because the dataset is generated randomly. The order of two locations that are very close to each other has little effect on the route distance calculation, so the model may get confused with this kind pairs of locations and yield a wrong order for the two locations.

4 Discussion and Conclusions

In this research, a tool for integrating 2D road data and elevation data into a 3D road data is developed. The main difficult task for this process is how to merge two different types of data. The 2D road data is consisted of several road segments, and each road segment is consisted of several discrete points, while the elevation data is a raster data with 30 meters resolution. To Address this problem, two methods are considered. The first one is using value of the block where road segment point belongs to. This method is simple but cannot handle the case that two points belong to the same gird of elevation data. The second one is interpolation for each point in the road data. In the tool developed by this research, which of the four subregions of the elevation grid each point belongs to should be found first, and then values of this grid and the other three grids that are adjacent to the subregion are used to do interpolation. The weight for doing interpolation is inversely proportional to the distance between the target point and grids center.

After the integration process, the network for a city is created. The main difficult task to resolve in this process is how to make an easily used network to do path planing. The mature method to resolve this task is creating a graph for the road network, which is consisted of nodes and edges. Nodes are created from the points in integrated road data, and edges are created by connecting the nodes sequentially. When creating edges, two cost function are integrated in: distance and travel time. Users can do path planing based on the cost function they choose. The distance cost function can make a good path planing in the real world circumstance. The limitation in this part is that the travel time cost function is based on several hypotheses, such as the road is clear, there is no traffic light, vehicles go at the max speed. These hypotheses have attribution to the deviation from the real world circumstance. Another main reason for the deviation is the road data itself. The tags of road data collected from OSM may not enough to reflect the real world case. For example, some roads may be only able for pedestrians, and some roads may not be able to access. To improve the performance of this process, data should be refined gradually, which can make the network more close to the real-world case.

After two processes mentioned before, dataset can be generated by the tool for training the revised Pointer Network. The model can achieve a relatively good route length ratio, however the model can not distinguish locations that are close to each other very well, which badly affects the accuracy of the model. The original Pointer Network can be extended for predicting optimized route for more number of locations than the number of training locations. But in my revised Pointer Network, the input of the model is changed to the adjacent matrix. The strengths of this revision is that it can be used for making route optimization of the real-world road network. The limitation of this revision is that it can not be extended to predict the optimized route more locations than the number of locations during training. Because the input layer of the model is fixed, but the size of the adjacent matrix is according to the number of locations to do route optimization. To improve the performance of the model, more data with more complex mode can be fed to the model during training, so that the model can learn how to handle these modes. The structure of the model can also be improved so that it can adapt to the variable size adjacent matrix.

References

[1] E.L. Lawler. *The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Wiley-Interscience series in discrete mathematics and optimization. John Wiley & Sons, 1985.

[2] Wikipedia. Travelling salesman problem.

[3] Shabnam Sangwan Chetna Dahiya. Literature review on travelling salesman problem. *International Journal of Research*, 05(16):1152–1155, June 2018.

[4] Kylie Bryant. Genetic algorithms and the travelling salesman problem. *Department of Mathematics*, December 2000.

[5] Miguel A. Marino Mohammad Reza Jalali, Abbas Afshar. Ant colony optimization algorithm (aco): a new heuristic approach for engineering optimization. *Proceedings of the 6th WSEAS Int. Conf. on Evolutionary Computing*, 16-18:188–192, June 2005.

[6] Mitica Craus Laurentiu Rudeanu. Parallel implementation of ant colony optimization for travelling salesman problem. *Proceedings of the 4th WSEAS Int. Conf. on Soft Computing, Optimization, Simulation and Manufacturing Systems (SOSM 2004),*, Miami(Florida), April 2004.

[7] René van Bevern and Viktoriia A. Slugina. A historical note on the 3/2-approximation algorithm for the metric traveling salesman problem. *Historia Mathematica*, 53:118–127, 2020.

[8] Roberto Goodrich, Michael T.; Tamassia. *Algorithm Design and Applications*. Number 513-514. Wiley, 2015.

[9] Gao Ye and Xue Rui. An improved simulated annealing andgenetic algorithm for tsp. In *2013 5th IEEE International Conference on Broadband Network and Multimedia Technology*, pages 6–9, 2013.

[10] Kris Braekers, Katrien Ramaekers, and Inneke Van Nieuwenhuyse. The vehicle routing problem: State of the art classification and review. *Computers and Industrial Engineering*, 99:300–313, 2016.

[11] Eliana M. Toro O., Antonio H. Escobar Z., and Mauricio Granada E. Literature review on the vehicle routing problem in the green transportation context. *Luna Azul*, 1909-2474:362 – 387, June 2016.

[12] Marc Schröder and Pedro Cabral. Eco-friendly 3d-routing: A gis based 3d-routing-model to estimate and reduce co2-emissions of distribution transports. *Computers, Environment and Urban Systems*, 73, 08 2018.

[13] Jiquan Wang, Igo Besselink, and Henk Nijmeijer. Battery electric vehicle energy consumption prediction for a trip based on route information. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, 232(11):1528–1542, 2018.

[14] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks, 2017.