CSE535 Portfolio Report

Wenzhe Zheng
Arizona State University
Tempe, American
wzheng41@asu.edu

***Abstract*—This project developed an Android mobile phone application software as required. Customers can choose the starting point and ending point in a selected area, and can choose to add a stop between the two points, and then the mobile phone program will use the user's starting point position to predict the distance to the end point, fuel consumption and time. Finally, the most optimized route is generated and the most fuel-efficient route is sent to the customer.At the same time, this ascending order allows the customer to map the time of the specified route to the Google map [2] for comparison, so as to allow the user to choose the most convenient option**

***Index Terms*—Google map API [2],Android Studio,Java,Fuel consumption,GPS**

I. INTRODUCTION

The map is the most important part of the mobile phone software, and the navigation system is one of the most important functions of the map. There are many companies in the world that provide global positioning system (GPS) functions and provide them to customers who need them. For example: Google, Baidu, Gaode, etc. These systems will help users find the starting point and ending point according to their needs. The most commonly used routes. The most common recommended route is the shortest route. Regardless of the traffic conditions on the road, fuel consumption and how the time is calculated, the distance from the starting point to the destination is the smallest. As fuel is used as the driving force for vehicle driving, the factors in calculating the road should also be taken into consideration. So we designed a novel calculation method to help customers find the route recommended by the vehicle with the least fuel consumption. This is an application we developed in Android. This program uses the Google Map API as the basis for the map. At the same time, we take distance, real-time traffic conditions on the road, traffic signal timing, and fuel consumption as the key elements for calculating each road. Then we use BFS as the search algorithm and calculate the top 4 routes that use the least fuel for all key elements. The map scope of our application this time is set to Tempe. Because the City of Tempe has added an adaptive system [3] for traffic lights, we cannot obtain real-time traffic signal data from the city government database or Google Api. So we took a hypothetical measure to help us process traffic signal data, and we will introduce this method in detail below.

A. System requirements:

- IDE: Android Studio 3.4
- Map API: Google Directions API [2],Google Distance Matrix API,Google Maps SDK for Android,Google Roads API
- Emulator(AVD): API 28+,Huawei Mate Pro10
- Program language:Java
- Packages:Android.Manifest,Android.location, Android.graphics, Com.google.maps

II. SOLUTION

In this part, we are divided into three parts to complete, the first is about the UI interface part of the phone:

A. Create UI element for Android Phone

The first page is the login interface. This project only provides the illusion of an algorithm, so the main page does not have too much design. You can see that after directly entering the page, there is a button to activate the GPS map:

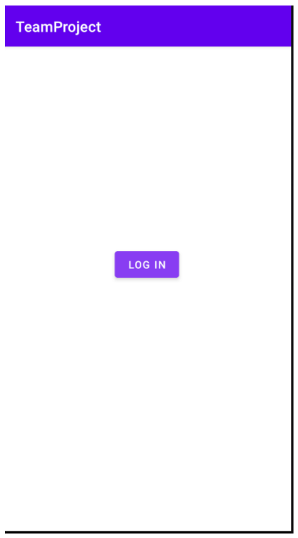


Fig. 1. Login Page

After clicking the Button, you will enter to the second page, and the GPS map function will be activated at the same time. The latitude and longitude that appears directly below the page

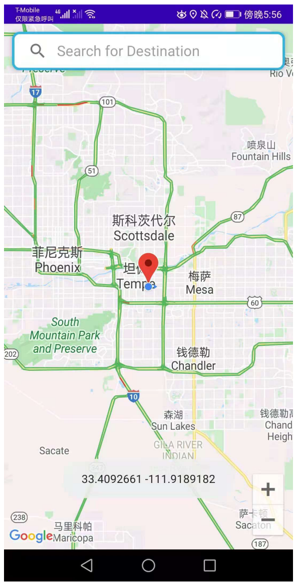


Fig. 2. Map UI with Search Box

In this interface, you can see that when we zoom in on the map, we can see the real-time traffic flow on the road. Blue means everything is unimpeded, yellow means some congestion, and red means particularly congested. At the same time, we can also see the names and functions of the buildings on both sides of the road. By looking at the icon, we can distinguish whether it is a supermarket or a gas station

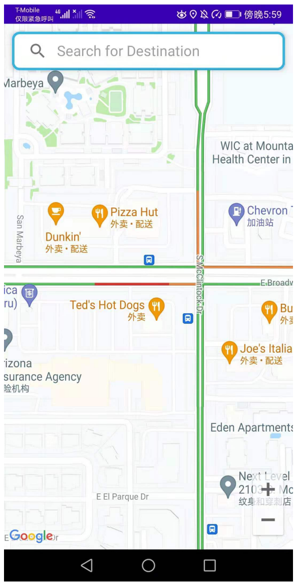


Fig. 3. Traffic conditions and Building name

We can see the traffic light after continue zoom in:

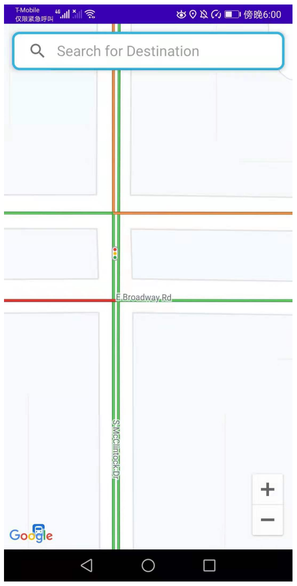


Fig. 4. Traffic Light

We can add a stop point by touching the screen with our finger, and we can also click the stop again to cancel it

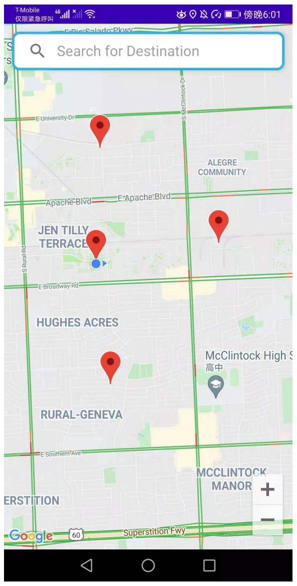


Fig. 5. Stop point

We can write the destination we want to go to in the search form and click on the search in the lower right corner. Here we enter Mcclintock high school

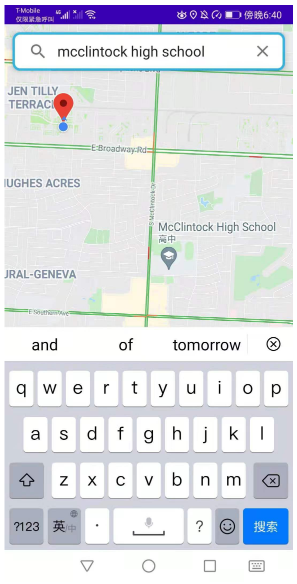
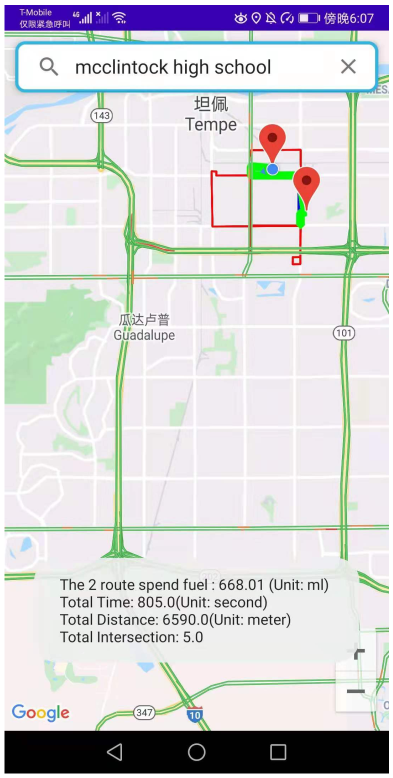


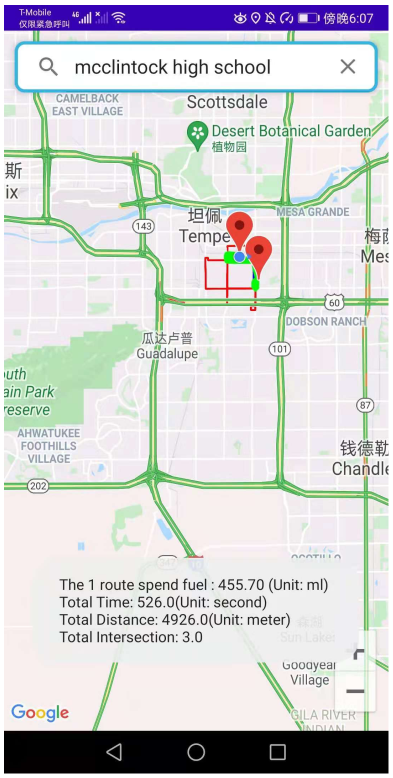
Fig. 6. Search

Then the map will calculate the four feasible roads to reach Mcclintock high school, and will also display the fuel consumption, time required, total distance and number of intersections passed by each corresponding road:



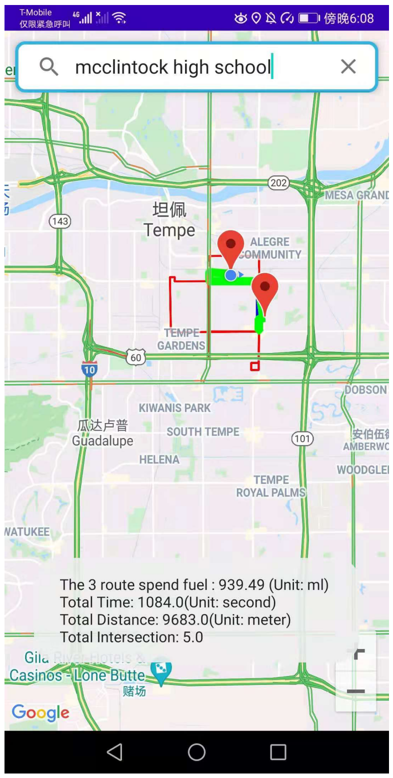
The 2 route spend fuel : 668.01 (Unit: ml)
Total Time: 805.0(Unit: second)
Total Distance: 6590.0(Unit: meter)
Total Intersection: 5.0

Fig. 8. Route2



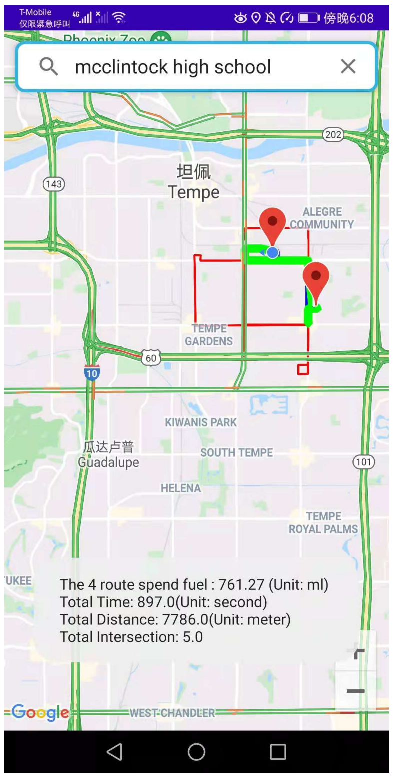
The 1 route spend fuel : 455.70 (Unit: ml)
Total Time: 526.0(Unit: second)
Total Distance: 4926.0(Unit: meter)
Total Intersection: 3.0

Fig. 7. Route1



The 3 route spend fuel : 939.49 (Unit: ml)
Total Time: 1084.0(Unit: second)
Total Distance: 9683.0(Unit: meter)
Total Intersection: 5.0

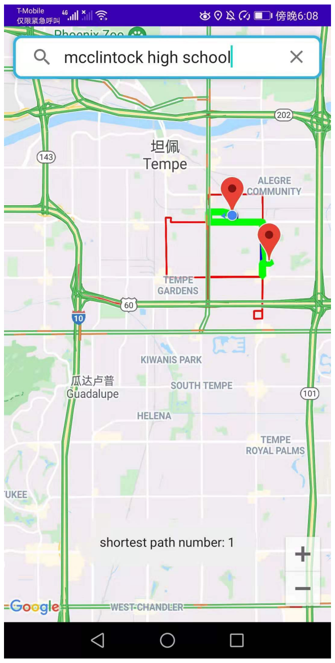
Fig. 9. Route3



The 4 route spend fuel : 761.27 (Unit: ml)
Total Time: 897.0(Unit: second)
Total Distance: 7786.0(Unit: meter)
Total Intersection: 5.0

Fig. 10. Route4

The blue route in the last picture is the best route chosen by the app for users based on fuel consumption, distance, time and road conditions. The blue route is the best route generated by Google [2], and the green route is the best route generated by the reference program. We can see that these two routes are basically the same.



shortest path number: 1

Fig. 11. Best Route

B. Map operation logic

The logic of this part is quite critical. It solves the problem of not getting real-time traffic light data and finding the path. I will explain it in detail below. We first applied for the Google Map API [2] to embed a map for the program. When we click the button on the main page, we will get our current location through the GPS provided by Google and display it in the reference program. Later, we also added Google Directions API, Google Distance Matrix API, Google Maps SDK for Android, Google Roads API provides us with a series of requirements such as road condition information. Then, we set the scope of the project to Tempe, and manually entered the latitude interval (33.383582, 33.439985) and longitude interval (-111.983661, -111.887204) of this location. All other coordinates and positions that are not in these intervals are regarded as invalid input, and the reason: no invalid input can be found. Displayed on the screen. At the same time, as shown in the figure below, we roughly divide the entire Tempe into a 7*7 matrix according to the direction of the road, manually input the longitude and latitude of each intersection, and at the same time we manually record the traffic signal time of each intersection, these All the information is included in 2.txt, which also solves the problem of not getting traffic signal information because the Tempe city government added an adaptive system [3] to the traffic lights. At present, we set the signal time of each intersection as 30 seconds of red light time for calculation.

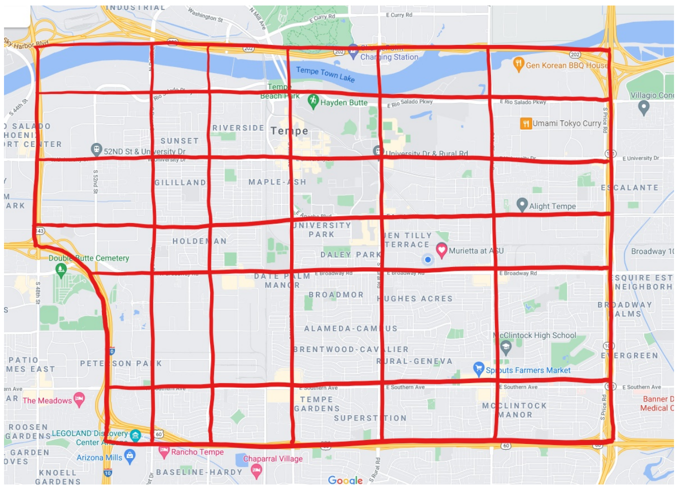


Fig. 12. Map Matrix

When the user confirms the starting point, stop point and destination, we will call Google Map API to determine the latitude and longitude corresponding to each point, and then we will find the closest point and The corresponding intersection on the map. From the starting point, we used the logical algorithm of BFS search. But we slightly changed the algorithm to achieve the goal i we want to find the path. We use the priority queue as the queue data structure, and each node in the queue will be calculated according to the following formula:

$$Value(P,S) = dist(P,S) + 0.8 * [T_{light}(P) + T_{light}(S)]$$

$where:$
P : *The current intersection*
S : *The next intersection along the road*
$dist(P,S)$: *Distance between intersection P and intersection S*
$T_{light}(X)$: *Traffic light timing for intersection X* Our algorithm this time uses priority queues and BFS, and searches every possible path until four paths are found. In the algorithm, we only consider the distance and the time of the traffic signal. Below we will explain the fuel consumption algorithm in this project.

C. Calculate Fuel Consumption

After we have four different routes, the next step is to calculate the fuel consumption of each route. Fuel economy (miles/gallon) is a common indicator of fuel consumption, but because each car has a different model and power, the economy of each car is different, so we find another way to Calculate the average fuel consumption of a random vehicle. [4] Another way to calculate fuel consumption is to use the speed shown in the figure. From the figure, we can see the relationship between fuel consumption and speed: When average car speed is less than 25 mph: [3]

$$AverageSpeed < 25(mph) \tag{1}$$

$$FuelEconomy = 5.75 + 0.85 \times AverageSpeed \tag{2}$$

When average car speed is greater than 25 mph but less than 57 mph:

$$25(mph) < AverageSpeed < 57(mph) \tag{3}$$

$$FuelEconomy = 24.5 + 0.1 \times AverageSpeed \tag{4}$$

When average car speed is greater than 57 mph:

$$57(mph) < AverageSpeed(mph) \tag{5}$$

$$FuelEconomy = 47.1 - 0.3 \times AverageSpeed \tag{6}$$

By using three formulas from above, the application is able to approximate fuel economy using only average route speed.

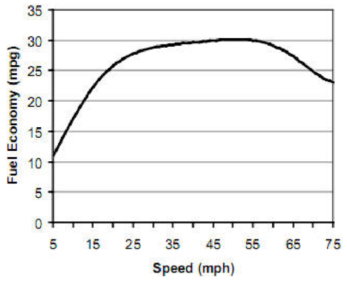


Fig. 13. Relationship between fuel economy(mpg) and speed(mph)

III. CONTRIBUTIONS

This is a team project. I am honored to complete the development of this project with Zian Zhang, Kekang Shu and Shanna Peterson. Thank them very much. I also participated in this project. Like:

- I manually input all the coordinates of 7*7Matrix and record the time of each signal light and input it into the database file
- Responsible for organizing remote meetings
- Create task form and assign it to everyone
- Use Github to manage the code of team members
- Apply for various Google APIs [2]
- Establish and use DFS to find the path
- Record and edit demo video
- Participate and write final group report

IV. LESSONS LEARNED

I learned a lot from this lesson, for example

- Understand and be familiar with using Android Studio
- Learned how to develop a GPS software logic
- Understand the process of developing an Android mobile phone software
- Learn to find the knowledge we need from Research
- Learned how to apply for API
- Learned how to use Github to manage the team
- Learned how to reasonably express one's point of view in the team

REFERENCES

[1] Kamga, C., Tario, J. D., Ancar, R., Yazici, A., Almotahari, S. Mudigonda, S. (2018). Reducing incident-induced emissions and energy use in transportation: use of social media feeds as an incident management support tool: final report. New York State Energy Research and Development Authority.
[2] Google API: https://developers.google.com/maps/documentation/javascript/reference
[3] Adaptive signal control technology. (n.d.). Retrieved May 02, 2021, from https://www.maricopa.gov/4553/Adaptive-Signals
[4] Gross, R. (2018). Standby Diesel Generator Fuel Consumption Calculation. https://doi.org/10.2172/1466196