CSE 535 Group Project

Zian Zhang
Arizona State University
Tempe, AZ, USA
zzhan329@asu.edu

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

Shanna Peterson
Arizona State University
Tempe, AZ, USA
snpeter6@asu.edu

Kekang Shu
Arizona State University
Tempe, AZ, USA
kshu2@asu.edu

***Abstract*—This project will develop an Android application that allows customers to choose two points in the Tempe area as the start and end points, and can choose to add stops at will between the two points, and the program will use the current user's location, To predict the time, distance, and fuel consumption to reach the destination, and you can specify the most fuel-efficient route. Compare the time of the specified route with Google Maps [2] to allow the user to choose the most convenient option.**

***Index Terms*—Google Directions API,Google Distance Matrix API,Google Maps SDK for Android,Google Roads API,Android Studio, Fuel consumption**

I. Introduction

Navigation system is one of the most important tools that exists on the map. Most maps including Google map utilize Global Positioning System(GPS) to access the user's location, and help the user to find their destination. Moreover, navigation systems help users to find several routes according to their choices. Most frequently, navigation systems will find the shortest route such that the distance of following this route from starting point to destination is minimum regardless of traffic situation, toll fee, etc. Fuel, as an important resource for vehicles, should also be a significant consideration. There are a lot of elements including speed, idling time, distance, etc affecting usage of fuel. Our team designed a novel way that helps users to find a route such that the vehicle spends the least amount of fuel. We developed a mobile application that deployed in Android. The application utilizes Google map API [2] as map foundation. Our navigation system will consider distance, traffic situation, traffic light timing as key elements to compute fuel for each route. Then we utilized Breadth-first search (BFS) as searching algorithm and consider all key elements to find top 5 routes using least fuel. Our application only consider Tempe as our scope in this project.

Because the Tempe City Government has added an adaptive system [3] to the traffic lights, we cannot obtain real-time traffic light information. The intelligent traffic light system consists of four parts: the intelligent traffic command center, ground induction coils, monitoring probes and traffic lights. The ground induction coil is buried under the traffic lights. When the vehicle is driving on the road, the dynamic coil will automatically count and transmit the digital information to the command center. Then the command center guides the signal lights according to the traffic flow to achieve the purpose of automatic timing of the signal lights. This kind of traffic signal light has no time display, which is very different from the traditional real-time control traffic signal light. When the traffic volume is heavy, the "adaptive" traffic lights will automatically extend the driving time, which is of great help to alleviating traffic pressure. This has a great impact on our acquisition of real-time traffic lights. There is no API available to provide real-time traffic signal data, so we have taken a hypothetical measure to help us process traffic signal data. We will introduce this method in detail below.

II. System requirements

- IDE: Android Studio 3.4
- Emulator(AVD): API 28+
- Program language:Java
- Packages:Android.Manifest,Android.location, Android.graphics, Com.google.maps
- Map API: Google Map API

III. Problem Solution

We develop the application by following high level steps listed below:

A. Create necessary UI element for user inputs

In a real world situation, the user will need to register with the server first before he or she can login to our application. But for the scope of this project and testing purpose,we only implemented a login activity with a login button so any user can access the application.

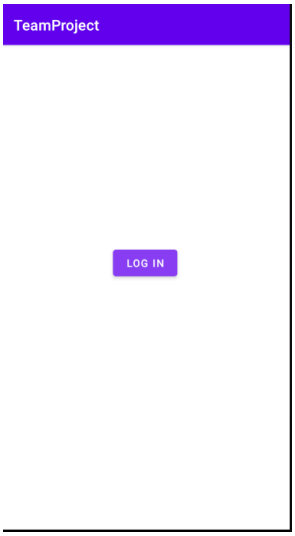

Fig. 1. Image of the Log in activity UI, a login button is provided for any user to test the application

After the login activity, we need to create the map UI activity to handle destination search and route display.

Fig. 2. Map activity UI with a destination search box

After zooming in on the map, we can see the real-time traffic flow on the road. Blue means everything is unblocked, yellow means a bit of congestion, and red means particularly congested. At the same time, we can also see the names of the buildings on both sides of the road and their functionality. Looking the icon, we can distinguish whether it is a supermarket or a gas station.

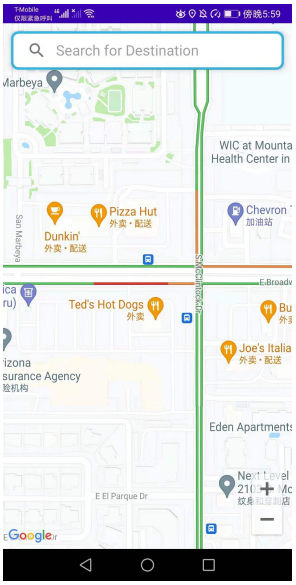

Fig. 3. Traffic conditions and Building name

Then zoom in on the map, we can also see the traffic light.

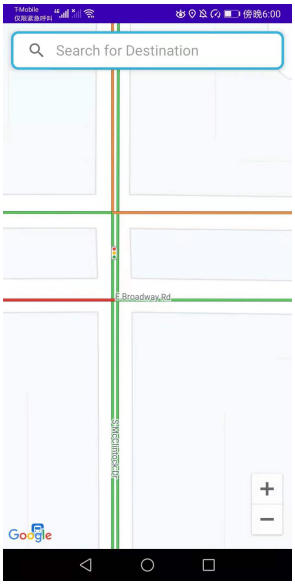

Fig. 4. Traffic Light

Then we added the function of adding stop points, and you can add stop points by touching the screen with your finger. As you can see in the picture, we have successfully added four stop points.

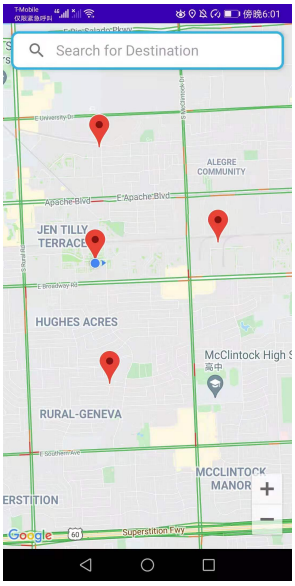

Fig. 5. Stop point

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.

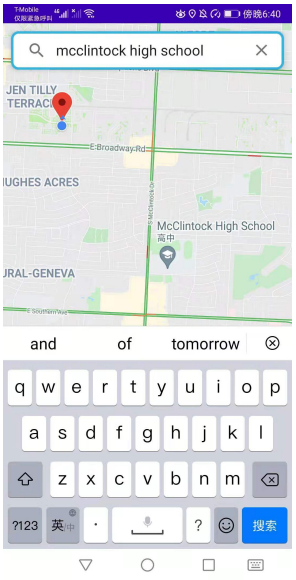

Fig. 6. Search

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:

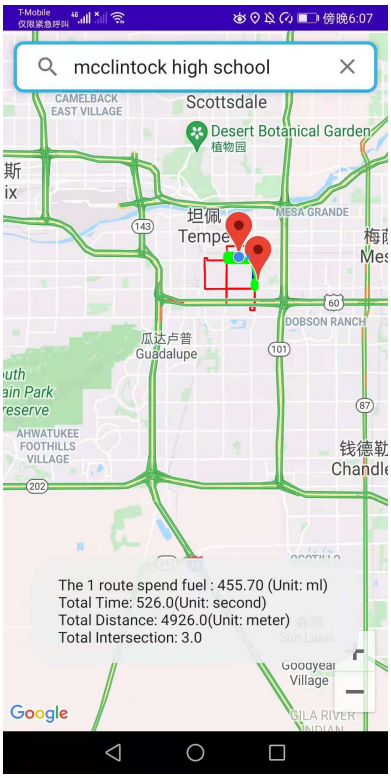

Fig. 7. Route1

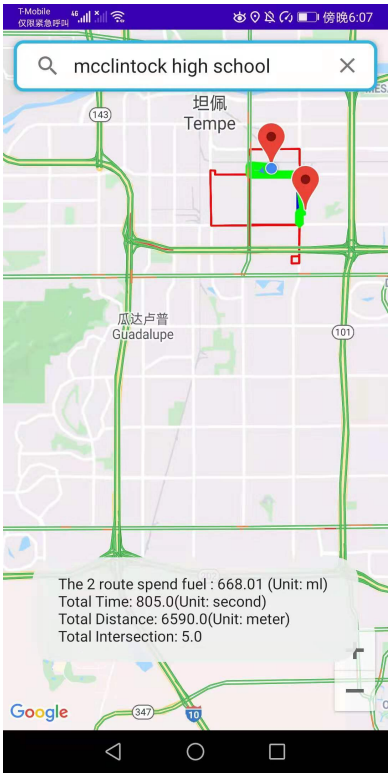

Fig. 8. Route2

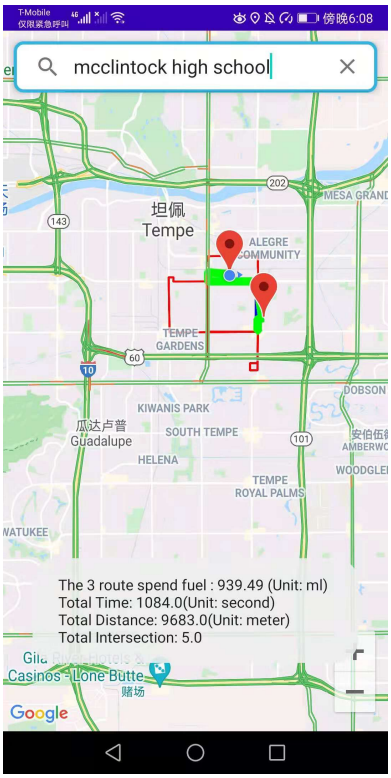

Fig. 9. Route3

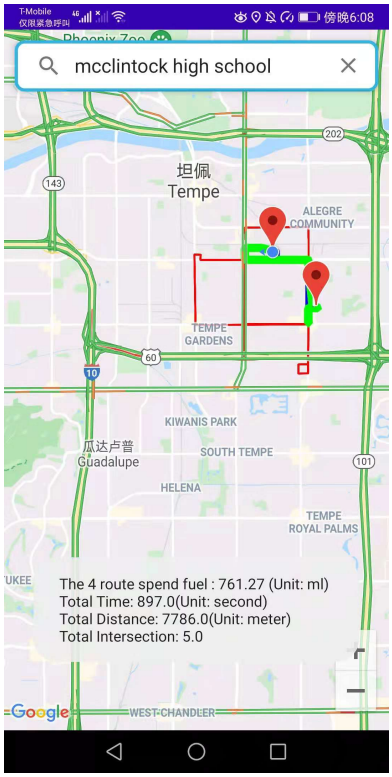

Fig. 10. Route4

Finally, the best answer will be selected based on fuel consumption, distance and time.The blue route in the figure is the best route generated by the google map, and the green route is the best route generated by the application. You can see that the routes basically overlap.

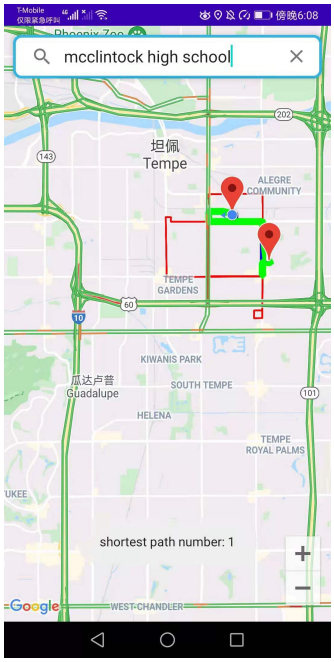

Fig. 11. Best Route

B. Map Implementation

We implemented a map tools using Google Map API and embed Google map on our application. Use's location will be acquired though GPS provided by Android and Google. We also implemented a search function that allows users to find destination according to name or GPS location [2]. Furthermore, we allow user click on map, and it will place a marker on map. The markers will be treated as optional stops while finding route, and each route will consider these stops as waypoints so that each route will go though these waypoints before arrive at destination.

C. Discovery of Route

a) Assumption:

- Longer distance causes more fuel
- Longer idling time causes more fuel
- User will follow speed limit

b) Algorithm Design: First, we defined Tempe as the scope of our project. Only location with latitude within from 33.383582 to 33.439985, and with longitude within from -111.983661 to -111.887204. Any starting location or destination is not within this area would be considered as invalid input and the application would not find route between any invalid input. We recorded main intersections existed in Tempe in our database. Each intersection, we recorded an estimated traffic light timing, estimated location with latitude and longitude. Therefore, we can create a matrix in Tempe according to intersection. In Figure 12, it shows how we created our matrix. The main reason why we only recorded main road including avenue, road and boulevard instead of recording street or drive was that the traffic light system deployed in Tempe used adaptive system which each traffic light timing is not constant. For other intersections that were not recorded, we would use 30 seconds as estimated red light timing.

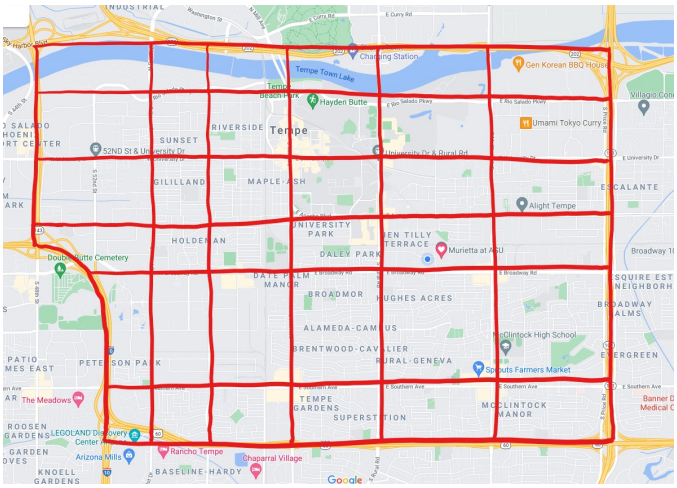

Fig. 12. Estimated Matrix

After user entered a destination, starting point and waypoints, we would call Google Map API to acquire corresponding GPS location with latitude and longitude for each point.

First we found the closest the waypoint and corresponding closest intersection in our database. Starting from staring point, we performed BFS searching algorithm. However, we modified the algorithm in order to achieve our goal. We used priority queue as queue data structure. The value of each node in queue would be computed based on the formula below:

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

$where$:

P : *The current intersection*

S : *Next possible intersection following particular road from current intersection*

$dist(P,S)$: *distance between intersection P and intersection S*

$T_{light}(X)$: *Traffic light timing for intersection X*

Our algorithm used the priority queue and basic BFS where we considered lowest value instead of considered lower level of graph, and explored every possible route until we found 5 routes that each route would go thought each waypoint and starting from staring point and arrive at destination. In this session of our algorithm, we only considered distance and traffic light timing and assumed that the user would follow the speed limit while driving on particular road. However, in next session which we had found five different routes, we would apply speed as key element to be a consideration for each route. The main reason was that we only recorded intersection of main road, and the speed of main road was similar to each other. Finally, each route would contain a set of intersections that contain latitude and longitude. Therefore, we can call Google Direction API to get detail of route so that we can plot each route to the map [2].

D. Calculate Fuel Consumption

After getting five different routes, next step is to calculate fuel consumption for each route discovered. Fuel Economy (mile per gallon) is a common indicator for fuel consumption, if car owner can provide the correct Fuel Economy value for their car, calculating fuel consumption will be much easier. But in reality, Fuel Economy value for each car varies from model to model, and sometime even the vehicle owner doesn't know the exact number. As a result, we must find another way to calculate the average Fuel Economy value for a random vehicle. Another way of calculating fuel economy is by using speed as shown in the graph. We can see that the relationship between Fuel Economy and Speed can be approximate to three linear equations:

When average car speed is less than 25 mph:

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

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

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.

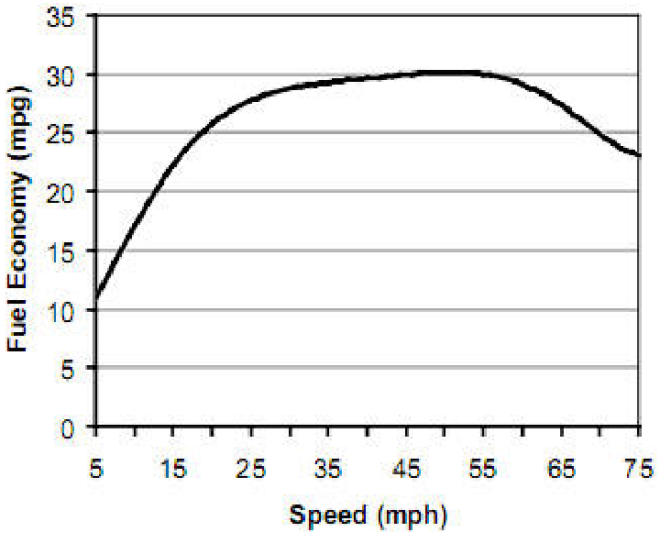

Fig. 13. Relationship between car's fuel economy(mpg) and its' speed(mph)

IV. Appendix

TABLE I
TEAM MEMBERS TASKS

Wenzhe Zheng	Task2, Task17, Task20, Task15, Apply for Google Directions API, Edit demo video,Write group project report
Zian Zhang	Task1,Task4,Task5,Task6,Task11, Apply for Google Distance Matrix API, Write group project report
Kekang Shu	Task7,Task8,Task9,Task10,Task12, Apply for Google Maps SDK for Android, Write group project report
Shanna Peterson	Task13,Task16,Task18,Task19, Apply for Google Roads API, Write group project report,Combine the code

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