Outstanding Academic Papers by Students

學生優秀作品
University of Macau

Faculty of Science and Technology

Big Data for Smart Transport: Augmented Reality based Transportation System

by

WONG KA CHON, Student No: D-B2-2761-2

Final Project Report submitted in partial fulfillment of the requirements of the Degree of Bachelor of Science in Computer Science

Project Supervisor

Prof. Leong Hou U, Ryan

21 July 2016
DECLARATION

I sincerely declare that:

1. I and my teammates are the sole authors of this report,
2. All the information contained in this report is certain and correct to the best of my knowledge,
3. I declare that the thesis here submitted is original except for the source materials explicitly acknowledged and that this thesis or parts of this thesis have not been previously submitted for the same degree or for a different degree, and
4. I also acknowledge that I am aware of the Rules on Handling Student Academic Dishonesty and the Regulations of the Student Discipline of the University of Macau.

Signature : _________________________

Name : WONG KA CHON

Student No. : D-B2-2761-2

Date : 21 July 2016
ACKNOWLEDGEMENTS

I would like to express my utmost gratitude to UM for providing the opportunity to carry out a project as a partial fulfillment of the requirement for the degree of Bachelor of Science.

Throughout this project, I was very fortunate to receive the guidance and encouragement from Prof. Leong Hou U, Ryan, also the effort from my partner Fan Wai Lon, Alan.
ABSTRACT

This project is about an android application development. It is an Augmented Reality application for users to explore bus stations and points of interest around them. The information of real time public transportation is provided. Direction suggestion function is also added into this application for users to find the shortest route to the destination, and the quickest route generated by a time-dependent network of public transport.
# TABLE OF CONTENTS

CHAPTER 1. INTRODUCTION ................................................................. 9

CHAPTER 2. RELATED WORK .............................................................. 13
2.1 Augmented Reality ................................................................. 13
2.2 OpenStreetMap (OSM) ............................................................ 14
2.3 Similar Mobile APPs .............................................................. 15

CHAPTER 3. DESIGN .................................................................. 16
3.1 Data Crawling ........................................................................ 16
3.2 User Interface and Front End Architecture ......................... 17
3.3 Augmented Reality .............................................................. 18
3.4 Front End of “Direction” ....................................................... 23

CHAPTER 4. IMPLEMENTATION ..................................................... 25
4.1 Data Crawling ........................................................................ 25
4.2 Data Organizing .................................................................... 26
4.3 Front End Architecture ......................................................... 29
4.4 User Interface ....................................................................... 30
4.5 Augmented Reality .............................................................. 32
   4.5.1 Data Preparing .............................................................. 32
   4.5.2 Position Calculation ..................................................... 35
   4.5.3 Maximum Coverage Algorithm .................................. 38
   4.5.4 From DirectionActivity ............................................... 39
4.6 Front End of DirectionActivity .............................................. 40
4.7 Offline Map ........................................................................... 42

CHAPTER 5. CASE STUDY ............................................................... 43

CHAPTER 6. DISCUSSION ............................................................... 46

CHAPTER 7. ETHICS AND PROFESSIONAL CONDUCT .................. 48

CHAPTER 8. CONCLUSIONS ........................................................... 49
CHAPTER 9.  REFERENCES........................................................................................................ 50
LIST OF FIGURES

Figure 1.1 overview of the project ................................................................. 10
Figure 1.2 overview of this application .......................................................... 12
Figure 2.1 Reality–virtuality continuum .......................................................... 13
Figure 2.2 AR for solar system using Wikitude SDK (JavaScript API) .......... 13
Figure 2.3 Macau on OSM .......................................................................... 14
Figure 2.4 One of similar mobile app ............................................................. 15
Figure 3.1 Life cycle of activities .................................................................. 18
Figure 3.2 values of orientation sensor (azimuth, pitch, roll) ...................... 19
Figure 3.3 Horizontal position calculation .................................................... 20
Figure 3.4 Rotated position calculation I ....................................................... 20
Figure 3.5 Rotate device in the clockwise direction ...................................... 21
Figure 3.6 Rotated position calculation II ..................................................... 21
Figure 3.7 Maximum Coverage Algorithm ................................................ 22
Figure 3.8 Character suggestion of Chinese input Method .......................... 23
Figure 4.1 Structure of OSM data ................................................................. 26
Figure 4.2 Organizing Points of Interest ....................................................... 27
Figure 4.3 Example of near bus station for large scale POI ......................... 27
Figure 4.4 Structure of class BusStop and BusRoute ................................. 28
Figure 4.5 Java program for parsing and organizing xml data .................... 28
Figure 4.6 Flowchart for the Front End architecture ................................. 29
Figure 4.7 Layout structure for “AR explore” ............................................. 30
Figure 4.8 Layout structure for “Direction” ............................................... 30
Figure 4.9 UI for MainActivity ................................................................... 30
Figure 4.10 Icons for bus stations and POIs .............................................. 31
Figure 4.11 Initial UI of AR .................................................................
Figure 4.12 Transition from inputs layout to results layout ..........
Figure 4.13 Preparing data for placing icons of BusStops and POIs ...
Figure 4.14 CPU usage with filter (top) and without filter (bottom)
Figure 4.15 Adjustment for roll value .............................................
Figure 4.16 levels of vertical position ..............................................
Figure 4.17 Result of rotating position calculation .........................
Figure 4.18 Display real-time situation ...........................................
Figure 4.19 Ranking of POIs (Higher is better) ..............................
Figure 4.20 Showing POIs with Max Cover (left) and without it (right)
Figure 4.21 Information for Direction in ARActivity .....................
Figure 4.22 Flow chart of keyword search .....................................
Figure 4.23 AsyncTask not applied (right) and applied (left) .........
Figure 4.24 Downloading map tiles of Macau in MOBAC ............
Figure 5.1 Case 1: Exploring .........................................................
Figure 5.2 POIs that route 33 can reach ........................................
Figure 5.3 Case 2: Find a Route .....................................................
Figure 5.4 Finding bus stations for route ....................................
Figure 6.1 Rotating the phone vertically ......................................
CHAPTER 1. INTRODUCTION

The objective of Big Data for Smart Transport is to collect users’ geolocation data, to analyze the traffic situation for public transportation in Macau. Also estimates real-time information by organizing geolocation data from users. It provides an application for users to check real-time information that using the location of other users or him/herself to generate. This becomes a cycle for the whole system. The more people who share their data, the more powerful service this application provides. Furthermore, Augmented Reality feature is included in the app. It helps users to locate the bus stations and points of interest in a visual way. Use the time dependent network to suggest a better path for user to arrive anywhere in Macau.

Our project is about the Augmented Reality and public transportation application development on android platform. The user groups of this application are local residents and travelers in Macau. This application is divided to two main parts – “AR explore” and “Direction”. As such, “AR explore” is for exploring things around user using Augmented Reality. Another is suggesting paths for users to go anywhere they want in Macau. Those suggestions are analyzed with a simulated time-dependent network of public transportation.

This application is developed by several special reasons as the following.

1. Many public transportation applications describe information to users just on the 2D map. It may be too difficult for some kind of users who are weak at reading map. Especially for travelers because they might not be familiar with the map of Macau even some residents do not know where the place corresponds to the position on map neither. Using Augmented Reality to show public transport information might be more visual and interesting. User could locate a bus station or place more efficient. However, relevant application for Macau is not found on app market.

2. Everywhere in Macau is traffic jam in recent years. Number of private cars in Macau is keeping increase. The private vehicle density of Macau is in the front rank of the world with an incredible rate. The reason why people would like to drive rather than taking bus is that it’s seriously worst for Macau public Transportation. We want to develop an application which is not just provide static routing suggestion, but also to consider the time for getting destination also with real-time traffic situation. It may help residents or any users to avoid traffic jam as possible. Hope smart transportation could conduce to resolve dead-loop of traffic problem in Macau.

3. Macau is a city for tourist. It would help for travelers if there is an application that combines tourist attraction and public transportation. In this application, travelers are able to explore which tourist attraction they can go to by taking bus routes nearby.
The brighter part is our application architecture for the whole project. The Augmented Reality feature has to rely on devices sensors and map data that stored in R* tree with a time dependent network. There will have connection between two projects for exchanging data to each other to maintain the cycle. To provide the real-time data, tracking user’s locations is necessary. The location will be transmitted to the server. Later on, user will receive a better result or service for his/her contribution.
Here will briefly introduce different part of this application. First for all, the data of all Macau bus stations, bus routes, buildings, roads, places and attractions in this application are all from the OpenStreetMap (OSM). They are not totally up-to-date and the authenticity is not ensured because the OSM is a free editable map of the world for public. Mining these data using Overpass API queries on a web based data mining tool, which calls Overpass Turbo, and then export and organize these data. After that, put these organized data into SQLite database. Geolocation data of which are specially stored in R-tree table for finding them more efficient.

In the “AR explore”, users can explore no more than 10 bus stations and points of interest (POI) around whom using Augmented Reality. By keeping track of geolocation that provided by the device’s GPS/network location provider, and listen the device’s orientation sensor to determine which azimuth the device face to, finding bus stations and POI that are nearby user from database. And then show their icon to the correct position on the device’s screen. Those icons on screen will keep moving in each 200ms according to the orientation value, and the size of which is decided by the distance to device’s current location, the icons are also resized in each location changing. Once the user click the bus route’s icon, the real time public transport situation will be showed. It briefly describe where the next bus is, and whether there has traffic jam or not by showing figure. Also show several attractions the bus route will arrive to. Those real time data are simulated in current status since those real time data cannot be collected from bus companies.

In the “Direction”, users are allowed to search paths in different several ways. Basically it provide keyword search for all bus stops, buildings, attractions and roads in database as the place of departure or destination. Search by category is also provided. Bus stops can be selected in different categorized area of Macau. For some users who do not know where he/she is, they can just click a location button and then it will request current geolocation once to the GPS/network provider as the place of departure. Offline map is provided for users picking place as the destination. After departure and destination are selected, system analyses which paths are the shortest, the quickest and the most convenient paths which would arrive to the destination in current time according to the time-dependent network. Walking to the destination or walking to a bus stop for transition may be considered as a way. Furthermore, the result can be showed using Augmented Reality. This way might be more clearly for user who is not familiar with Macau such as travelers. In this Augmented Reality page that is opened from “Direction”, only show the selected suggestion’s information, such as where the departure bus stop is, POI that can arrive to by that suggestion and so on.

This application can be used in offline without real-time transportation situation because the database is stored in local storage of the device. However, as the principle mentioned in this chapter before, the device has to be able to receive mobile signal or GPS signal at least if user want to use Augmented Reality. So it can also be used for some travelers who are not able to connect to the Internet with the phone, to find the most optimize direction to the destination.
In this project, I have worked on

- Crawling and organizing most of the data from OSM
  
  Using a web based data mining tool to crawl data. Organize them by using java code. Finally insert into database.

- Designing user interface and architecture for the whole application

  Design the icon of this application, and user interface of each page. Some of those are designed with animation by using xml and java.

- Augment Reality (“AR explore”)

  Monitoring the orientation sensor and tracking geolocation, find the bus stops and POI nearby in the database and then place them to correct position on the camera.

- “Direction” front end

  I mainly work on the front end which provides several ways such as keyword-search, search on map and category for users to input. And then passing the user’s input to my partner for the result.

More detail of my contributions will be show in the following chapters below.
CHAPTER 2. RELATED WORK

2.1 Augmented Reality

Augmented Reality is a technology which augments real world environment through computer calculating with inputs of hardware or sensors such as image, video, location data. Computer generates virtual elements via recognizing images of GPS location data from the real world, and then put it onto the specified position of device’s screen. The virtual elements could be generated with some interaction for users.

![Figure 2.1 Reality–virtuality continuum](This figure is from Wikipedia)

Augmented Reality has some difference from the Virtual Reality, which becomes trend in recent years. Virtual Reality could totally base on a virtual world whose elements are all simulated. Augmented Reality only puts virtual elements on the real world which cannot be replaced. It provides information that cannot be seen by human eyes in the real world. With that information, the real world will be looked more diverse in human eyes. That’s why it calls “Augmented” Reality.

Wikitude SDK

There is an Augmented Reality technology provider calls Wikitude, which founded in 2008. Wikitude focused on location-based augmented reality through the web app at the beginning. After few years, they launched Wikitude SDK which utilizing geolocation, image recognition, tracking, 3D model rendering and video overlay technologies to redefine their proposition. It is available for Android and IOS platform and smart glasses. [1]

![Figure 2.2 AR for solar system using Wikitude SDK (JavaScript API)](This figure is from Wikitude’s documents)
In our case, Wikitude SDK might be unavailable because our application is native Android and the location-based service of Wikitude is not provided for native Android platform. However, it provides completed service for web app (JavaScript). One of the examples in documentation webpage for Wikitude SDK (JavaScript API) is a demo for showing solar system in AR. It scans the solar system picture on a paper, and then recognizes the 2d image on the paper, to render a 3d model and show to the screen with tracking. Moreover, to use basic features of Wikitude SDK, several hundred in EUR is required to pay. Also, it provides a trial version with watermark for free. [1]

2.2 OpenStreetMap (OSM)

OpenStreetMap is a collaborative project for an online map which can be edited by public. The creator was inspired by Wikipeida. [2] It has more than two million users. Every registered user is able to contribute for the OSM. The mechanism for editing is similar as Wikipeidia, it records every modification. There is provided a web-based editor on official OSM website calls id which programed by Mapbox for registered users to contribute GPS, vector data or knowledge of user’s experience to the place. Others tools or programs could also be able to use for contributions such as JOSM which is programed in Java. OSM uses Open Database Licence (ODbL) as its licencing terms of its map data since Sep. 2012. [3]

![Figure 2.3 Macau on OSM](image)

As the map in Figure.2.3 for Macau, the sea area that confirmed to be under Macau in Dec 2015 is updated to OSM, even the adjustment for line 4 bus route that is adjusted few days ago can also be seen on OSM. This shows its data is updated enough for our data source in application.
2.3 Similar Mobile APPs

All application for Macau public transportation on Google Play provides routing suggestion and show suggestions in map. The suggestion they provided is generated without time-dependent network. Probability of getting traffic is high even through it could lead user reach the destination. Moreover, any real-time information about public transportation cannot be found in relevant application. We realize that getting real-time data of public transport is not an easy thing to implement without support from government or any public transportation company, but getting support from them is not the only way to collect real-time data. User data is useful for real-time data. We should try to provide real-time information by using and analyzing user data.

Furthermore, as the Figure 2.4, those applications only show information on a 2D map. However, not all people are familiar with reading map by considering people in higher ages who using smart phone. So we should try to provide augmented Reality to showed information on camera.

![Figure 2.4 One of similar mobile app](image)
CHAPTER 3. DESIGN

This application’s name is “MACAR”. Combines “Macau” and “AR”. Users are able to explore around Macau in Augmented Reality. Even though native app can only be installed into the specific platform, and the programming languages are totally different for developing two main smart-phone platforms, device’s hardware and sensors are highly used for AR. Considering the complex and user experience, we decide to develop android native app instead of hybrid and web app. So this application can only be installed into android platform.

3.1 Data Crawling

To implement this application, the data of all Macau bus stations, bus routes, buildings, roads, places and attractions should be stored as much as possible. We need to find the data source which provides:

- Bus stations
- Bus routes
- Buildings, Roads, Attractions, Place

For the data of public transportation, there are much enough and the most authentic in DSAT’s website since it is the government department. Some of those data are showed as images, as same as the paper of route’s detail that are placed at the bus stop. To extract the data from the image, it might spend a lot of time. This source is not suitable. But some of the text data can be used.

Google map is a powerful map. It has different kinds of API for developer. And those API are concise and easy-to-understand. Without considering the price of API usage, Google maps API might be suitable for mining data from Google. However, the data of Macau in Google map are outdated somehow. The new campus of university of Macau is not added into Google map until now. So we find other source for the data.

Openstreetmap (OSM) is a free editable map for public. For the data of Macau, this map is more update than Google map but the authenticity is not ensured because everyone can edit the data in OSM. There is Editing API for fetching and saving raw data from/to the OpenStreetMap database, and Overpass API for read-only. In this case, Overpass API is suitable for us and it is also easy-to-understand. The OSM data structure has four core elements: Node, Way, Relation, and Tag. Node is point in the map, it contains latitude and longitude of a point in map. Way is a list of Nodes. Relation is a list of Nodes, Ways and other Relations. Tag is key-value pair, it stores data of object. According to the keys list, there are some useful keys-values in our case. As the table 3.1.1 below, key and value are needed to searching all bus station and bus routes in Macau. For buildings, roads, attractions and places, it could use key to crawling all relevant data without any specified values in our case.
While all necessary data is collected, use Java to organize those data as the structure below:

- **Bus stations** - ID, name, location, bus routes, district
- **Bus routes** – route number, carriers, stations
- **Buildings, Roads, Attractions, Place** – name, location, type

Information in DSAT’s website might be taken as reference for district’s category of bus stations by parsing its HTML file [6], since OSM seems does not have district’s category for Macau.

After that, create three normal tables in a SQLite database, there are “BusStop” for bus stations, “BusRoutes” for bus routes and “POI” for all buildings, roads, places, and attractions. And then insert the data into specified column in relevant table. This database can be packed in the apk file. It will be installed with the application together once the apk file is executed. On the other hand, there are some other special tables with R-tree feature for locations. The relevant work of R-tree tables is credited to my partner.

### 3.2 User Interface and Front End Architecture

There are two main parts in this application, “AR explore” and “Direction”. Including the main page, there are three activities basically (In Android, activity is similar with page, it has five state of life cycle in basically – onCreate ➔ onResume ➔ onPause ➔ onStop ➔ onDestroy. The activity controls the screen when calling onResume, and onStop to leave from the screen. For more detail please visit the Android developer website[7]). For users picking place as the destination on the map, a new activity for the map should be added. Moreover, users are allowed to check the direction suggestion using AR. It needs an activity to show the suggestion in AR. There is an activity for “AR explore” already, so this activity could be shared. Totally there are four activities.
In the MainActivity, it provides two buttons for opening “AR explore” and “Direction”. Corresponding activity will be created after a button was clicked.

There are basically three layouts in ARActivity. The bottom is a layout for the camera preview. The camera of the device will be occupied until this activity was paused. Above the camera preview layout is a layout for the icons of bus stations or POIs. Parameters of all elements on this layout will be modified dynamically according to the values of orientation sensors and device’s geolocation. The top layout contains header and footer to display information of the selected bus station or POI. These three layouts are showed together to the screen. For the DirectionARActivity, it is almost as same as the UI of ARActivity. The difference is showing things that are related to the suggestion from DirectionActivity.

Divide two groups for layouts in DirectionActivity. One for users entering inputs, one for system showing the direction suggestion to the users. It provides several ways to selecting places as departure and destination such as keyword search, location, category and map. For the map, offline map has high priority to be considered since we expect this app can be used as possible by who are not able to connect to the internet. The layout group of suggestion is hided at first. After search button was click, exchange the visibility between two groups. Animations are added during the transition between two groups to make it smooth. And then show the suggestions using list view.

3.3 Augmented Reality

Most of Augmented Reality application might need image recognition. In this case, image recognition is unnecessary because this is supposed to show things that are not able to see at the user’s location. It means that we do not need to recognize any image from the camera preview. However, some information about the specification of camera is needed such as view angle. This section requires permissions to control camera. On the other head, all data of bus stations and POIs that stored in local database are 2D data. This section will perform some calculation to make that look like 3D for showing them in AR.
To find things nearby, the center point and radius are necessary. So the GPS and network location provider will be needed for getting the device’s current geolocation as a center point. Set the radius to the 500m. And then it has enough condition to find things nearby. Considering users might be moving when they are using this application, the location will be requested for each few or dozens of seconds. Use different types of icon represent bus stops or POIs will be showed. When the icons were clicked, the name and distance will be displayed. If the clicked icon is a bus stop, it will display bus routes icons of that bus stop, and they are also clickable. After those icons were clicked, it will provide a real-time situation to show how long the next bus’s distance to the selected station is, and the traffic situation. Also, it will show which attraction the bus will arrive to. However, the real-time information is simulated since we cannot get any accurately real-time data from anywhere in current development of our project.

Orientation sensor will be used in this section. The orientation sensor is software-based. It combines data from two hardware-based sensors – accelerometer and magnetic field sensor. The value from magnetic field sensor will has significant error when there are nearby magnet, electronic device or anything affect magnetic field from the device. Orientation sensor provides values of azimuth pitch and roll of the device. The value of azimuth is 0 when the device faces to north, 90 for east, 180 for south and 270 for west. The value of pitch is positive when rotates toward and negative when rotates backward. The value of roll is positive when rotates to counterclockwise direction and negative when rotates to clockwise direction.

The next step, determine which place is out of view angle using azimuth value. Get the azimuth value from the orientation sensor and get view angle (A) of device’s camera. Also the resolution of screen (W for width, H for height) is needed for the following horizontal position calculation of each icon. First, calculate the angle (θ) from azimuth of the device to the bearing of each bus station or POI nearby. Compare θ with camera view angle. It should be considered as out of view angle if θ is larger than half of A. When the things nearby is in the camera view, calculate how many horizontal pixels per degree of camera view angle (p=W/A), and then multiply p by θ, and perform an addition with horizontal center pixel, hence we got the horizontal pixel (x) that should be placed to. Here is the formula for horizontal position calculation.

\[
\text{Horizontal Pixel (x)} = p \times \theta + \text{Horizontal Center Pixel}
\]
The horizontal position is now calculated. Vertical position \((y)\) and the size of icon will be decided by the distance from device’s location to corresponding place. The maximum radius is 500m. And divide it to ten levels by each 50m. The nearest level will be showed on the vertical center of the screen. More farther will be more upper and smaller. This could make it more stereoscopic. For the place that was considered as out of view, there is a pointer icon placed to the edge on screen to indicate it is out of view.

The above calculation is only suitable when the device is not rotated like the phone in Figure 3.2. Let’s define it as the normal state. When the device is rotated (roll) and user is standing normally on the ground, everything in the device is rotated in user’s view for sure if any adjustment is not applied. To make it still look like normal state in user’s view, it should be rotated to opposite angle in device’s view. The trigonometric function is used for calculating which position should be placed to when the device is rotated. Base on the previous calculation we got \((x, y)\). And we know roll value \((\alpha)\) from the orientation sensor. Conditions are enough for the calculation now.
Rotate things around the center of screen. The center of the default coordinate system is at the left-top on screen and positive x-axis is on right side, positive y-axis is on downside. So define the center point of a new coordinate system to the center of screen \((x = x - \frac{w}{2})\) and use the same directions from the default. The distance \((d)\) from new point \((x', y')\) to the center point will be equals to the distance from origin point \((x, y)\) to center point. When the device is rotated in the clockwise direction like Figure 3.5 (\(\alpha\) is negative), \(y'\) should be negative if \(x\) is positive and positive if \(x\) is negative. Vice versa. Here is the formula for \((x', y')\).

\[
x' = d \cos \alpha \quad y' = d \sin \alpha
\]

And then convert them back to the default coordinate system of the device \((x' = x' + \frac{w}{2}, y' = y' + \frac{H}{2})\).

![Figure 3.5 Rotate device in the clockwise direction](image)

However, this rotation calculation is only suitable for the origin position point on x-axis \((y=0)\). For the origin point at anywhere on the screen, there is more complex algorithm to calculate the rotated point. As same as before, we know origin point \((x, y)\) and rotate angle \(\theta\).

![Figure 3.6 Rotated position calculation II](image)
Likewise, convert the origin point to the new coordinate system which was defined before \((x = x - \frac{W}{2}, y = y - \frac{H}{2})\). From the equations of counterclockwise rotation of axes \(^9\), we have

\[
x' = x \cos \theta + y \sin \theta \\
y' = -x \sin \theta + y \cos \theta.
\]

This equation is calculating the new point after rotated axes in counterclockwise direction with a positive angle. It is the same as rotating points clockwise. However, the positive y-axis of the defined coordinate system is point to bottom. We need to reverse the y-axis before calculating then turn it back. Hence, there are

\[
x' = x \cos \theta - y \sin \theta \\
y' = -(x \sin \theta - y \cos \theta) = x \sin \theta + y \cos \theta.
\]

Finally, convert the new point to the default coordinate system of the device \((x' = x' + \frac{W}{2}, y' = y' + \frac{H}{2})\). Then all icons on screen will always display horizontally in user’s view.

**Figure 3.7 Maximum Coverage Algorithm**

When an attraction icon was covered by another attraction icon, select the more famous one to display. In current situation, the standard of famous is the amount of how many bus routes pass through. First, calculate how many horizontal pixels between their center points \((d)\). If \(d\) is shorter than the half of the sum of their width, compare their ranking of famous.
3.4 Front End of “Direction”

The “Direction” suggests several options of route to users based on inputting place as departure and destination. It provides different way for input. They are keyword search, category search, locate and pick on map. Locate is only provided in place departure and pick on map is only for destination selection.

![Figure 3.8 Character suggestion of Chinese input Method](image)

The result of keyword search will be displayed when input each character. Keep listening the input field to detect whether the content is change or not. It will select a full name of items in BusStop and POI table which match or contain words of the content of input field. However, the preview or suggested character of the Chinese input method in smartphone will be recognized as a part of the content in input field. It means when user is typing for one Chinese character, the content will change one to five times. Each change will also access the database once. To reduce CPU usage, that meaningless input should be filter. So 800ms is set to a countdown-timer for the time interval of typing one character. Each time the content changed, cancel the previous countdown-timer and restart it. Searching items in database after the countdown was finished.

For the category search, bus stops will be classified by districts. Macau has three main areas – Macau Island, Taipa and Colona. And each of them could be divided by several districts. Users are allowed to select bus stop in district category. The UI for this part is designed to tab view. There are three tabs for Macau Island, Taipa and Colona. A relevant selectable list for districts will be displayed when clicking a tab. Each item in that selectable list contains a sub-list for bus stops which are in corresponding district.

Set a location button for getting current location of user when it was clicked. It will request geolocation to Network geolocation provider only once. The location of the device is unnecessary to be tracked. On the other hand, the map for picking place as destination could be used when offline, so the map tiles should be pre-download to local storage. We decided to use “Run Bike and Hike” map style from MapBox as our map tiles source. Also, different map source has different zoom level (as tiles levels), the scale from level 0 is the whole world, and higher levels zoom to smaller area and have more tiles. Likewise, a tile contains smaller area with higher levels. For Macau, having level 15, 16, 17 and 18 are enough. Mobile Atlas Creator is a program which creates offline map package. Download the map tiles of Macau in level 15-18 from MapBox via this program into a zip file. This package can be place into a server for users to download if we have a server. It current state, just places it to the local
storage of the device. Also, it is necessary to limit the available zoom level and restrict the extent of map views in this application. Otherwise it might cause out-of-bounds errors due to the limited offline map source.

Finally, system can now generate paths from user’s input with the time dependent network. After the system returns the result, put them into a list and provide a button that opens an ARActivity for specified result in each item.
CHAPTER 4. IMPLEMENTATION

4.1 Data Crawling

For the data crawling, use overpass API which is a read-only API to send a query and gets the OSM data that relevant to the query. Overpass API provides two different query languages, overpass XML and overpass QL [4], which can always be converted to each other. It runs on the third party service. On the official website, it recommends a web-based data mining tools – overpass turbo which is front-end to run the query and display the response on a map.

As the Table.3.1.1, the key-value pair for the bus routes is “route-bus”. Let’s try to query a bus route data using overpass QL query at first.

```xml
//query for line 71
rel[route=bus][ref=71][network=Macau];
out geom;
```

[ref=71] is to specify the line number for a bus route to query. The data format of response is xml, its information is much enough, there has a role as forward_stop or backward_stop for each stop of route. To query all bus routes of Macau, only need to remove the tag ref in the query. Next, let’s try to query bus stops in overpass XML with a key-value pair “highway-bus_stop”.

```xml
...<query type="node">
  <has-kv k="highway" v="bus_stop"/>
  <area-query from="area"/>
</query>
...
```

The data format of the response is XML too. The basic information for bus stops is provided except the zone which the bus stop is in. Those zones for bus stops are necessary since category search is provided for user picking bus stop as the departure and destination in direction feature. DSAT’s website provides bus stops information with zones, it can be classified later. There is one last large data set is left, that is all places, buildings, roads and so on of Macau. The query is similar as last query, use the seven keys for them according to Table.3.1.1, value is not necessary in this query. And the query is not only for nodes but all nodes, ways and relations. Also, most of the buildings or places are polygon on the map so request the center point of each rather than calculated by ourselves.

```xml
...<query type="node">
  <has-kv k="building"/>
  <area-query from="area"/>
</query>
...<query type="way">
  ...
  <query type="relation">
    ...
    <print mode="meta" geometry="center"/>
  ...
```
Since all nodes, ways and relations are requested, the results include data in these three types. It only has the location value if a place is node. When the location is way, it contains a list of nodes and provides the nodeID for each node. All contained nodes are placed in the bottom of the file with its location value. All nodes’ location value in the nodes list of a way it has to be found in the contained nodes list in the bottom of file via the NodeID. Likewise to the ways list of a relation, each way is needed to be found in the bottom by a WayID, and then find each node of that way by NodeID.

Result:

```
<relation id="4497465" ...>
  <center lat="22.1634629" lon="113.5588602"/>
</center>
<!-list of ways-->
  <member type="way" ref="322859393" role="outer"/>
  <member type="way" ref="322859399" role="inner"/>
    <tag k="name" v="研發大樓 Research and Development Building"/>
    ...
</member>
</relation>
...

<!-bottom of file-->
<node id="3296514374" lat="22.1632536" lon="113.5587114" "..."/>
...
<way id="322859393" ...>
  <!-list of nodes-->
    <nd ref="3296514374"/>
    <nd ref="3296514369"/>
    ...
</nd>
</way>
```

### 4.2 Data Organizing

First of all, parse all nodes and ways in the list of bottom of file. Store the location and nodeID into the class OsmNode and a list of OsmNode into class OsmWay also with the wayID. We can now parse all necessary information of the Points of Interest. Concurrently, calculate the maximum distance (length) from a node on its boundary to other nodes if the POI is way or relation (polygon). It will be useful for some calculation after. Also, it needs to recognize whether the each item of data is place for tourism or not. Define it is for tourism if there is “tourism” key or the value is “mall” or “place_of_worship”. And the data types of interests also include Node, Way and Relations, so they need to be divided in to three classes – NodeOfInterest, WayOfInterest and RelationOfInterest. Basic structure of them is similar with OsmNode and OsmNode but the different is that they contain information of interests in the class.
After that, the necessary information of bus stations and bus routes in XML file could be parsed. Find the nearby bus station for each POI. The POI’s maximum length that has been calculated just now can be used now. The reason is that some of the POI’s cover considerable area of land. They should be considered as points. If the POI’s maximum length is larger than 400m, select the bus stations which within 500m from any point of the POI. And select the bus stations near by the center point of POI. Also store that POI itself into the list of nearPOIs in BusStop class.

Find nearby bus stations for POIs or nearby POIs for bus stations in order to prepare the data for routes suggestion. Also calculate the ranking of attraction by how many nearby bus stations of POIs. Likewise, find nearby bus stations for each bus station in the same way due to the bus interchange, but maximum length calculating is not necessary because all bus stations cannot be considered as points in Macau. Afterwards, download the HTML file of bus stations list from DSAT’s website. The file has busStopID in corresponding district table. Bus stations were classified into different district category after parsing the HTML file.

In the class BusRoute, it contains basic information and one or two list of bus stops which the route will arrive to. Some of the bus route have two terminal station and some of them only have one. Hence, two list of BusStop should be prepared in order to store forward_stop and backward_stop.
Finally all xml data of bus stations, bus route and POIs have been organized. When the class contains other classes, insert a number which is index in the table relevant to the classes since SQLite is not able to insert class. According to the statistic from the program, there are 423 bus stations and 75 bus routes. The sum of all buildings, places, roads and so on is just around five thousand in Macau. I believe that many places are left. However, these data are much enough for the normal usage.

In this program, I have worked on crawling, organizing and inserting the data of bus stations, bus routes, buildings, places, roads and so on. For the data of traffic sign, generating the time dependent network and creating R* tree table in this program are credited to my partner.
4.3 Front End Architecture

According to the design, this application has four activities at least. They are MainActivity, ARActivity, DirectionActivity and MapsActivity. The ARActivity will be called from MainActivity for “AR explore”, and DirectionActivity with passing values for specified bus stations, the things it shows will be different by calling from different activity. Hence, only need to determine whether the activity has value from DirectionActivity or not. And then perform the function for the relevant features. On the other hand, MapsActivity is made for user to pick place on the map. This activity returns a location value picked by user back to the DirectionActivity.

![Flowchart for the Front End architecture](image)

There are two buttons in MainActivity for the corresponding activity. Nothing will be passed to any activity form MainActivity. This activity was made as the root of this application, and user cannot do nothing except clicking button in which. In the ARActivity, there are basically three layouts in different levels. Camera preview is placed on the bottom layout. It requires the permission to control camera of the device. The camera will be occupied after onResume() is called. When ARActivity is paused, it will release the camera. The middle layout contains the icon of bus stations and POIs within the view angle of device’s camera. Those icons will always being change programmatically based on the orientation sensors value and geolocation of the device. The top layout is for the header which displays hints and information of selected bus station and POIs. In the footer which shows a list of route. When selected a bus station, and displays real-time situation of the next bus when clicked a route in the list. The real-time information will be keep refresh in each few seconds. In the case of called by DirectionActivity, the structure for the activity do not change as well. The difference is the data which to be showed.
DirectionActivity has two main layouts at the same level. One is for the inputs and another one is for the result. Hence, these two main layouts should not be visible at the same time. Their visibility must be the opposite of each other. MapsActivity can be called for result from the layout of inputs. And results layout could call the ARActivity with values to specify the data that will be showed in AR.

In the MapsActivity, only the layout which contains the map in full screen is showed. The main and only feature of this activity is picking place as the destination for the direction. Return the location values to DirectionActivity before destroying this activity, or return nothing if there is no any place was picked. When the results layout is showed, each item in the list of result has a button for calling ARActivity. It passes values about the corresponding route suggestion to ARActivity.

4.4 User Interface
Since the MainActivity is made for opening different activity, place two buttons on a background image and the name of this application on the top. It will look concise.
Prepare a FrameLayout for camera preview in the ARActivity, this layout is blank in the initial state because the CameraPreview will be called when this activity is in `onResume()` of lift cycle. And then, prepare one more FrameLayout for icons which will be added programmatically. This layout is also blank in the UI designing phase.

The icon will not be showed when the corresponding location of bus station or POI is out of view angle from the camera except which is selected. The Figure 4.10 indicates the different state for the icon. However, when nothing is selected, the overflow icon will be showed in a small arrow like “>” and “<”.LinearLayout is used for the header and footer as the top layout. This layout type is different from FrameLayout and RelativeLayout which is used in MainActivity. Views could not be added by a coordinate value and any spaces between views are not allowed. All views in LinearLayout must be added one by one in order, and the position of views depends on the order they are added. In the addition, display a small TextView for accuracy of GPS/Network location above the footer.

As the Figure 4.11, only header and footer can be seen in the initial state. The background color of layout for is transparent for two layouts in the bottom, so the background color (white) from the device’s screen was seen. Camera preview will be show immediately in general once the ARActivity is opened. And the icons will be added to the after located the device.

The first layout is showed when opened DirectionActivity, is the layout for inputs. It has two blocks to contain EditableTextView in the middle. Click a button to exchange the layouts. Animation could be added during the transition to make it smooth.

```java
protected void moveUp_anim(final View view) {
    float translateY = view.getHeight() + view.getTop();
    Animation animation = new TranslateAnimation(0f, 0f, 0f, -translateY);
    animation.setDuration(200 + ((long) translateY / 5));
    animation.setStartOffset(100 + ((long) translateY / 5)); //delay
    view.startAnimation(animation);
    ...
}
```

This piece of code is setting animation which moves the blocks on inputs layout upward to leave the screen. All views will not be move upward concurrently, but
delay few hundred milliseconds depend on the block’s position. The delay time for lower block will be longer than the upper block as well as the duration of animation. After all views from inputs layout have been out of screen, apply a animation fadeIn() to results layout.

![Figure 4.12 Transition from inputs layout to results layout](image)

The effect of delay can be seen obviously in the middle of Figure 4.12. The right side in the figure is results layout. A list of result will be showed after the calculation. Each item in the list has a button which is designed for calling ARActivity.

4.5 Augmented Reality

4.5.1 Data Preparing

Camera will be occupied immediately when ARActivity was opened. CameraPreview will be added to the bottom layout of this Activity. On the other hand, request location updates via GPS and network provider, and register a sensor listener for orientation sensor. The change of values from both could affect what the screen shows and the position of which.

Depends on the hardware of device and physical environment where the device is, the time interval of location updated is between one to thirty seconds in general. It might be longer if the mobile single is weak. onLocationChange() provides class Location which contains Lat/Lng and accuracy value of this location update. In each update, prepareData() will be called once. It removes all previous views on the middle layout if the update is not the first time, because the new data may be change a bit from previous. That’s one of the ways to avoid null pointer error. And then check null for the bundle, to determine whether the activity is called from MainActivity or not (The purpose of bundle is to pass values between activities). It is opened from MainActivity when bundle is null. In this case, find five nearby bus stations and important POIs surrounding the location within 500m separately.

```java
nearBusStops = mdh.getNearBusStop(location, 5, 500, false);
```
Calling this method to find the five nearest bus stations within 500m from given location. The fourth parameter is a Boolean value for sorting from near the far. The reason why we want to sort them from far to near is that the view added of front position will be covered by back position in the order. In this method, after selected all necessary information from database, the distance and bearing from device’s location to bus station’s location is calculated. These two values are important for placing its icon to the screen. After that, it returns a sorted list which contains specified number (in this case is 5) of bus stations.

![Diagram of data preparation process](image)

**Figure 4.13 Preparing data for placing icons of BusStops and POIs**

Create a LinearLayout to contain icon of bus stations and TextView for distance, and add it into a list of LinearLayout which for all views that prepared to be display. And then set an onClickListener to each LinearLayout in the list in order to enable the clickability of which. Data and icons of POIs could be prepared likewise the bus stations by the method shown below.

```java
public void prepareNearPOIs(Location location){
    nearPOIs = mdh.getNearPOI(location, 5, 500);
    ...
}
```
Concurrently, according to the Android official developer document, `onSensorChange()` that updates to orientation values, will be run for each 200ms because the default time interval `SensorManager.SENSOR_DELAY_NORMAL` is set before. As same as the location provider, the value from orientation sensor depends on the hardware’s capability and the physical environment. Nearby electronic device, magnet or anything affecting magnetic field will cause significant error to the sensor. The reason is that this software-based orientation sensor combines values of hardware-based accelerometer and geomagnetic field sensor. The second sensor is easily being affected in case the magnetic field around the device is interfered. Unfortunately, this kind of error is difficult to be resolve due to the principle of magnetic.

The update interval of SensorManager is not allowed to be modified with custom value. So we create a looping Handler with custom interval 200ms to access orientation value. Even if the default interval of SensorManager is 200ms as well as the interval of Handler we set, sometimes we don’t want it update too fast when the CPU usage is high. The Handler calls `locateViews()` for each loop. It places icons in the list of LinearLayout which contains bus stations and POIs icons to the middle layout. Since there are considerable amount of calculations in `locateViews()`, it should reduce CPU usage of unnecessary calculations. We store each update of orientation value, and each time before calling `locateViews()`, the azimuth in new orientation value will be compared with the previous. Ignore the update if the difference of azimuth is less than ±2 degrees. This will also stabilize all things on screen because the camera (phone) may be shaking a little in human hands.

![Figure 4.14 CPU usage with filter (top) and without filter (bottom)](image)

As shown in Figure 4.14, before applying filter, CPU usage is always close to 20 percent. The filter is effective obviously. After applying filter, CPU usage can be reduced to 0 when the azimuth is not change.
4.5.2 Position Calculation

Rotation value (roll) is necessary for the calculation later. As the range of rotation value shown in Figure 3.2, it is not able to indicate the device’s rotation state only by one roll value. Therefore, we need to convert the range of roll value from -90→0→90 to -180→0→180, by determining the pitch value. When pitch value if positive (top edge of device is lower than bottom edge), calculate a value for addition (d\(\theta\)) to the roll value. Thus, we have

\[
d\theta = \pm 180 - 2 \cdot \theta
\]

![Figure 4.15 Adjustment for roll value](image)

After adjustment of orientation value, we can now begin to calculate the coordinate for each bus station and POI icon. First of all, calculate x as horizontal position of icon for normally state of device (without any rotation). According the first formula which was defined in chapter 3.3, bearing value from current azimuth (\(\theta\)) of device and to target bus station or POI is the only variable, and the screen with (W) camera view angle (A) and pixels per degree of view angle (p) are constant value. So the parameters of this calculation are current azimuth and bearing value from north to target for calculating \(\theta\).

```java
public int calcViewX(int newAzimuth, int newBearing){
    int angle=0, busStopX=0;
    //calculate the angle from current azimuth to given bearing
    angle = calcAngleDiff(newBearing, newAzimuth);
    //calculate x-axis value according to the angle
    if( Math.abs(angle)<cameraAngle/2){
        viewX = (layoutWidth/2) + angle*pxPerDegree;
    }else if (angle>0) {
        viewX = 0;
    }else if (angle<0){
        viewX = layoutWidth;
    }
    return viewX;
}
```

The position is on the boundary of the screen for those bus stations or POIs which are out of view angle. The reason is that the icon will be display as an arrow when nothing is selected by user or it is selected. For the vertical position (y), it depends on the distance from device’s location to the target place. y is separated to ten levels for each 50m of geo-distance and 50 pixels between each level. As well as the size of icons, there are ten levels by each 50m and 10 pixels shorter for each edge.
Figure 4.16 levels of vertical position

The proportion in the Figure 4.16 is scaled for showing the concept clearly. Level 1 is containing nearest place from device as well as level 10 contains the farthest. Coordinates of all icons are calculated for the normal state of device. Use the rotated position formulas in chapter 3.3. In the case that y is on the vertical middle (level 1), call the function which performs the first formula that only needs x value and rotate angle for new coordinate calculation.

```java
public int[] calcViewCoor2(int viewX, int rotate){
    int[] coordinate = new int[2];
    float po = viewX - layoutWidth/2; //hypotenuse
    double newAngle = rotate + plusRotateAngle;
    coordinate[0]= layoutWidth/2 + (int)(po*Math.cos(Math.toRadians(newAngle))); //x-axis value
    coordinate[1]= layoutHeight/2 + (int)(po*Math.sin(Math.toRadians(newAngle))); //y-axis value
    return coordinate;
}
```

Even through the second formula can also calculate it, we also call that function in order to avoid unnecessary steps in calculation. When the point is not on the vertical middle of screen, the second formula can be used in the case. Pass the coordinate of origin point and rotate angle to the function which performs second formula.

```java
public int[] calcViewCoor1(int viewX, int viewY, int rotate){
    int[] coordinate = new int[2];
    float oldX = viewX - layoutWidth/2;
    float oldY = viewY - layoutHeight/2;
    double newAngle = rotate + plusRotateAngle;
    return coordinate;
}
```

In the both function, `plusRotateAngle` is the addition of adjustment which was calculated just now, and `rotate` is raw roll value from orientation sensor. After performs calculation according to the formula, the functions return a new coordinate for the point to be placed to. Also the LinearLayout which contains icon should be
rotated itself to the opposite direction of device’s rotation. Finally, if the location is updated, remove all view on middle layout and add new view for the new bus stations and POIs which is found by updated location. Else if the location has not updated yet, apply animations for moving icons to the new coordinate.

The icons are shown in an acceptable position when rotating the device as the Figure 4.17. When bus station icon was clicked, all route of that station is display into the footer. Those routes are shown as icons that are clickable for real-time situation of the next bus about the selected route. Concurrently, it shows a list of POIs which the selected route will arrive to, and five POIs which is the front position in the list to be shown using AR.

Since we do not have any real-time information, a looping Handler data is created to that provides fake real-time for each five seconds, to demonstrate show how the real-time data will be show. When this Handler was created, the interval time of accessing orientation value is increased to 300ms in order to reduce CPU usage.
4.5.3 Maximum Coverage Algorithm

Furthermore, Maximum Coverage algorithm is applied for showing POIs of routes. Considering some icons of important POIs are covered by another, we have counted the famous ranking before. The standard of this ranking is to determine how many bus stations surrounding the POIs.

![Figure 4.19 Ranking of POIs (Higher is better)](image)

However, the ranking does not rank POIs well in my expectation. As the ranking list in Figure 4.19, the most “famous” position is the place where actually is not that famous than the St.Paul’s and other famous attraction. Use it as the standard for ranking temporary since we don’t have any user data in current stage of our project.

```java
//add POI views with MaxCoverage Algorithm
analyzeCoverage(originPOIs) {
    for each originPOIs {
        boolean shouldBeAdd=true;
        //if POI covers bus stop, show bus stop
        int PBangleDiff = calcAngleDiff(busStop, originPOI[i]);
        if (Math.abs(PBangleDiff)*pxPerDegree < (originPOIs[i].width+ busStop.width)/2) continue;
        for each tempPOIs {
            int angleDiff = calcAngleDiff(tempPOIs[j], originPOIs[i]);
            //if POIs are cover each other, compare with the ranking
            if(Math.abs(angleDiff)*pxPerDegree < (originPOIs[i].width+ tempPOIs[j].width)/2) {
                //compare ranking
                if(originPOIs[i].getRank()>tempPOIs[j].getRank()){
                    //replace higher ranking
                    tempPOIs.remove(j);
                    tempPOIs.add(originPOIs[i]);
                    shouldBeAdd=false;
                    break;
                };
            }
        }
        //POI does not cover anything
        if(shouldBeAdd) {
            tempPOIs.add(originPOIs[i]);
        }
    }
    return tempPOIs;
}
```

This function returns a new list which filtered origin list by Maximum Coverage algorithm. To get the distance between two icons, we need to know the difference of bearing values between two place, and then multiply it with the pixels per degree. At the first in this function, check whether the POI icon covers the bus station or not,
filter the POI icon if yes. And then check POIs each other, if an icon doesn’t cover anything, add it in to the temp list. Otherwse compare the ranking between two overlapping icons, higher ranking will be add into templist which is return value for this function. After got the return from this function, hide all nearby bus stops and POIs icons except the selected bus stop, then show filtered POIs icons of the selected route.

![Figure 4.20 Showing POIs with Max Cover (left) and without it (right)](image)

4.5.4 From DirectionActivity

When the Bundle in this activity is not null, it means that there are some values from DirectionActivity since that is the only one activity which will pass values to ARActivity. The values from DirAct contain several index of bus stations corresponds to the column number in table of database. Prepare data of bus stations which will be showed in AR, and set the order number to the bottom of icon. All nearby bus stations and POIs are no longer to be showed. After that, all process of hardware data and all features are the same as before.

![Figure 4.21 Information for Direction in ARActivity](image)
4.6 Front End of DirectionActivity

Text input field is provided on inputs layout of DirectionActivity. Implement the input field using class AutoCompletedTextView. This class can a dropdown list for keyword. We want to set 800ms delay of Chinese input method as the gap between inputting each Chinese character. Find relevant place according to the input keyword.

To implement this, add a TextChangedListener to each AutoCompletedTextView. When user is typing a character into the TextView, beforeTextChanged() is called before the character is added in to TextView by system. Cancel countDownTimer during this stage. And then restart it in onTextChanged(). If there is no any new input within 800ms, function keywordSearch() will be run once the countdown is finished. This function performs two simple to search item in database,

```
sql1 = "SELECT ID, STOPID, NAME FROM " + _BusStopTable +  " WHERE NAME LIKE '%%"+keyword+"%' OR STOPID LIKE '%%"+keyword+"%';
sql2 = "SELECT ID, NAME FROM " + _POITable +  " WHERE NAME LIKE '%%"+keyword+"%' OR OTHERNAME LIKE '%%"+keyword+"%';
```

Users are also allowed to input keyword as the STOPID of bus stop or the old name or other name of a place. After selected items from database, combine the STOPID and NAME to a string if the item is from BusStopTable. Define a Char that represent the type for the result, and the ID which corresponds to the column in table of the result. These three elements are added into an ArrayList to be returned. Moreover, request single update of location when user click the locate button to set current geolocation as the place of departure. The location does not need to be updated as
ARActivity for saving power. For the destination, we provide a map for user to picking place. MapsActivity will be called once the map button was clicked via `startActivityForResult(mapsAct)`, to get the location where is picked by user.

When all necessary blanks were filled, users are allowed to click the button to request suggestions for the route. Consider the calculation of suggestions with time-dependent network will spend some time, it is necessary to use AsyncTask for the calculation. Otherwise, the application will freeze in user’s view even through the calculation is performing normally.

![AsyncTask not applied (right) and applied (left)](image)

*Figure 4.23 AsyncTask not applied (right) and applied (left)*

On the left side of *Figure 4.23*, everything in the thread freezes including the UI while the calculation is performing. Even through a loading icon has been added, it freezes either because it is a part of UI. On the right side of that figure, AsyncTask is applied. Any frozen in that thread cannot affect the thread of the UI. Everything in thread one will have feedback during the calculation. Once it finish, update the result list in the UI. Also, place a button in each item of list to pass values and open ARActivity for check result in AR.
4.7 Offline Map

Use Osmdroid as the API to place a map on MapsActivity. We decided to use offline map instead of online map since we supposed our application can also be used in offline. All materials in this application should be offline as possible. The map tile source we used is “Run Bike and Hike” from Mapbox. According to the official document of Osmdroid, they suggest a program Mobile Atlas Creator as a way to collect offline map tile source. For Macau, level 15 to 18 for the zoom level can contains all places in Macau. After downloaded all necessary map tiles of Macau into zip file, place it to local storage in the phone. Finally it can now be used.

In the MapsActivity, the only thing needs to be shown the map. Use class XYTileSource as the type of tile. And then set a minimum level (zoom out) and limit the scrollable area for the map because we don’t want user pick a location where is out of Macau and we only provided the tiles for Macau. Each tap from user is stored via calling singleTapConfirmedHelper() and point a marker to where user picked. A “OK” button is placed in the corner of screen, once user click “OK” it return the picked location to DirectionActivity as destination or return location(0,0) if nowhere is picked.
CHAPTER 5. CASE STUDY

Design some case to test the quality of this application. These cases check “AR explore” and “Direction”. In the case for checking “Direction”, the ARActivity called by Direction is counted in to “Direction” instead of “AR explore”.

First case, it is about a traveler who just visited Red Market, and he did not plan the next place he will go to. The traveler can use our application to explore around. He open our app and click into “AR explore”.

Once the “AR Explore” was opened and updated location, nearby bus stations and POIs were showed. In this case, even though the traveler can find nearby attractions, he is not interest in these shown POIs. However, checking POIs of specified route is also a part of features in our application.
He clicked bus route randomly. Finally, he clicked into route 33 from a bus station. And then he decided NewYaohan which displayed as POI in “AR explore” to be the next place he will go to. Furthermore, he got the real-time situation of the next bus which will arrive to there. It showed the next bus had two stops left, but he is unnecessary to run to the bus station for not missing that bus. Because we also display the distance from user’s current location to the bus station for route 33. He could evaluate himself whether he can arrive to the bus station on time for not missing any bus by real-time situation and distance to the bus station.

For the case two, there is a student who wants to go to University of Macau by taking bus, but he did not know where he is. Our application solved his problem. First, he clicked into “Direction”, two input text field were showed for departure and destination. Since the place as departure was in known, he click the locate button to search his current location as the departure. And then search University of Macau by keyword. After he indicated the places of departure and destination, he clicked “GO” button to inquired routes for that.

![Figure 5.3 Case 2: Find a Route](image)

Our application suggested several route for his case. We showed all stations for departure, interchange and destination. However, he did not know where those stations are. He did not worry about that since he was using our application which provides Augmented Reality to check every place. He clicked “AR” button which is on the right side in corresponding group header, to found the location of those bus stations using Augmented Reality.
Location of all bus station which relevant to that route was showed using Augment Reality. The order of the bus station which will be arrived to is shown under each bus station icon. Also, the basic information for that route was displayed on the header. Likewise the “AR explore”, we provided the real-time situation for the next bus. Finally, he had enough information to go to the University of Macau since he knew the location of the bus station for departure, interchange and destination of that route via Augmented Reality. And also the line number which he was going to take.

In both two cases, our application is able to find the solution for users. The results are completely solved uses’ problem in specified situation. Even through it helped a little for uses, we observed out application could be better. More information and services could be provided in this app, to make users have better experience.
CHAPTER 6. DISCUSSION

As the first in previous chapter, the Augmented Reality feature of our application is able to assist users to solve their problem completely. Every icons on the camera move fine in different the orientation state. The delay of moving icons is exist but it is in the acceptable range in our expectation. However, the icons cannot place exactly on the corresponding position on the screen to indicate relevant bus station or POI location accurately. This problem might be cause by error value of location provider and orientation sensor. It is difficult to solve this kind or error due to some physical reason. So we recommend user to use Augmented Reality feature in suitable environment as possible to, reduce probability and range of error.

![Figure 6.1 Rotating the phone vertically](image)

Moreover, the icons moving are only considered on horizontal orientation without vertical orientation in Augment Reality feature of our application. It means the icons don’t follow the vertical rotation to move. This makes the Augmented Reality do not “Real”, because the icons will show to the wrong position corresponds to the real location if the device is facing to the ground as shown in Figure 6.1. To solve this problem, we could calculate the icon position with pitch value from orientation sensor in the future.

On the other hand, the information we showed is clear enough for indicating what/where the place is. However some content is lack such as the routes of bus station. Even through the number of lack is less, there are only few dozen bus routes in Macau, any lack information of bus routes might affect the route suggestion with time-dependent network. We did not fill this lack because our project was mostly on the final stage when we found that. We could find more sources for the data in the future instead of OpenStreetMap only. Moreover, stores could be added in to our data as POIs. All stores information can be found on Yellow Page Macau. Some of them provide location data, so it is possible to be showed in our Augmented Reality feature with category filter.

Regarding the category filter for POIs to be shown in Augmented Reality, it has not been provided in our application. We only show attraction as important POIs for now because the screen will be mess, CPU and memory usage will also be high if everything is showed to the screen. This filter could be a feature in our app in future.
to be the solution for this issue. Users could choose any thing or place they want to explore such as stores, restaurants, ATM machine but not only attractions.

Finally, for the maximum coverage algorithm, we only calculated the overlapping with horizontal overlapping pixels. If the distance of icons center is less than their width, they are considered as overlapping each other for now. However, they are not overlapping actually if the vertical distance is larger than their height. So we could consider with vertical overlapping together in the maximum coverage algorithm.
CHAPTER 7. ETHICS AND PROFESSIONAL CONDUCT

Since some features in the application request location data of user, and even trace user’s location. We realize that location data is a highly privacy for everyone. Our responsibility is to assure the safety of all information from every user. Every valuable contribution about location data from users would be highly appreciated. We promise all location from users must be protected in anyway. Any location provider of user’s device will be unregistered immediately once the relevant features were closed, and all location value will be destroyed with the activity together. To protect user’s privacy in advance, location data we collected could be anonymous. K-anonymous which makes protection to the data by re-identification could be used in future.

We realize that battery power is precious for smart phone. For saving power of the phone of user, we try our best to reduce CPU and memory usage by avoiding unnecessary calculation. Also, close and release all hardware sensors which are used in this application once they finish the task, such as camera and location provider. We would like to provide a good user experience as possible.

The IDE this application developed on is Android Studio which released by Google and it is free. API we used for the map in our application is credited to osmdroid. All data except distinct category, is collected from OpenStreetMap, and for the distinct category information which is from DSAT, we strictly follow the term of use that is indicated on DSAT website. Those HTML files which are copied from DSAT must be used as non-commercial usage. Moreover, the figures shown in the report are drawn using PhotoShop, web-based drawing tools – draw.io and Geogebra. Some of the figures are the screen captured of our application and other software.
CHAPTER 8. CONCLUSIONS

As the objective of Big Data for Smart Transport, we really hope our application can resolve the traffic problem of Macau, since the traffic circumstance is getting worst in these recent years. Every one want to reach their destination as fast as possible, but it may be difficult to be achieved for current traffic situation. So, smart transportation may assist every citizen who will use public transportation to reach anywhere they want in Macau as fast as possible by big data.

Augmented Reality is one of the interesting features of our application. Information about public transportation is showed to user more visually than 2D map. For the feeling of users, this kind of technique may be technological. It may attract user to experience our application more. User’s experience should be highly regarded in every application. Moreover, this feature could really help the people who are not familiar with Macau or the map of Macau. Hence, our application may be useful for that people group.

In short, the main features in our applications are:

- Augmented Reality based explore
- Route suggestion with time dependent network
- Check suggestion in Augmented Reality

Regarding the problem that was mentioned in pervious chapter, they are necessary to be solved in the future. We want our application to be implemented and sustainable in Macau. This project could totally be implemented by combining the other project of Big Data for Smart Transport.
CHAPTER 9. REFERENCES


