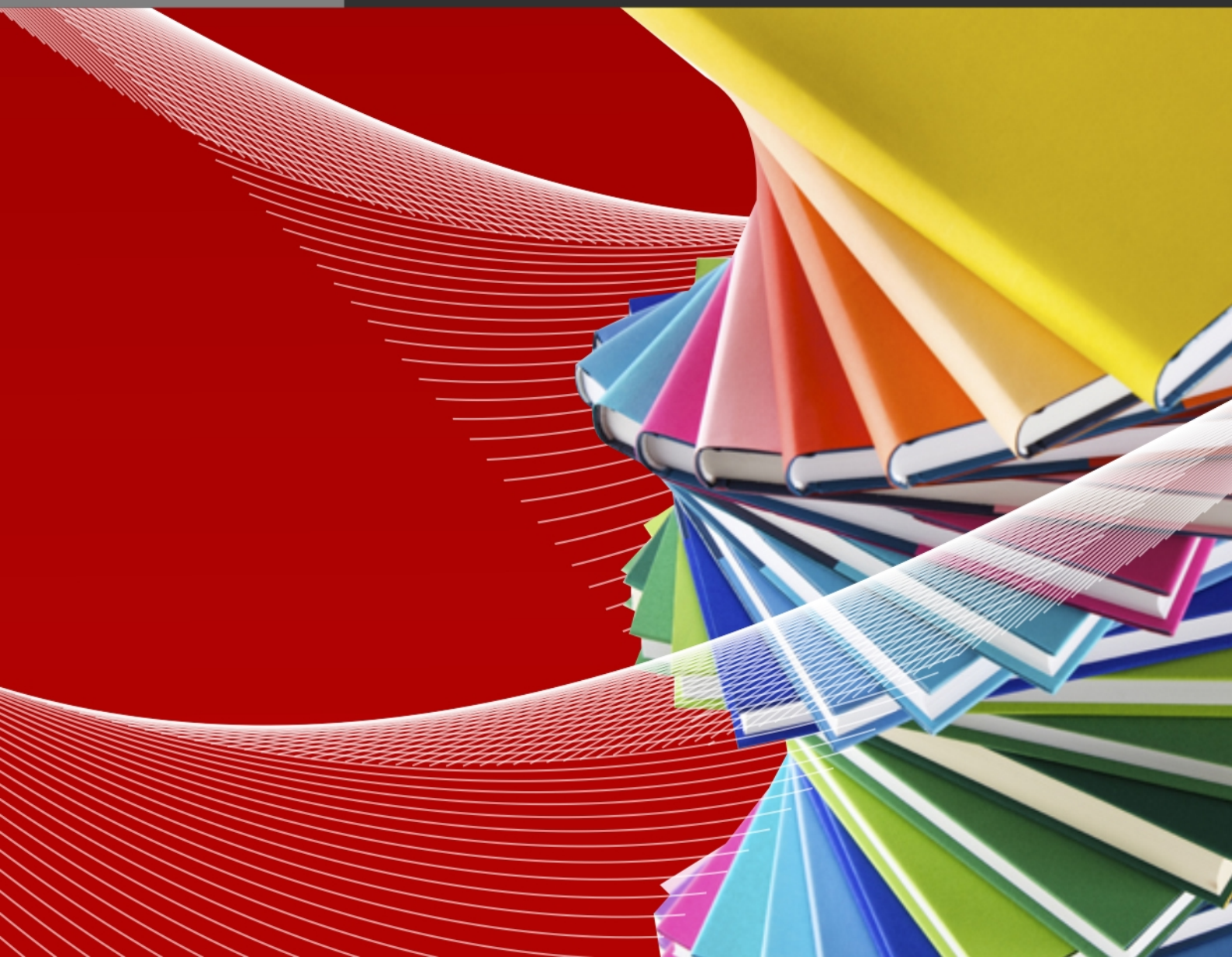
澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

Outstanding Academic Papers by Students
學生優秀作品



University of Macau

Faculty of Science and Technology

澳門大學
UNIVERSIDADE DE MACAU
UNIVERSITY OF MACAU

Indoor Navigation by Image Recognition

by

Leong Chi Chong, Student No: DB325023

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

Project Supervisor

Prof. Pun Chi Man

26 May 2017

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 : Leong Chi Chong

Student No. : DB325023

Date : 26 May 2017

ACKNOWLEDGEMENTS

The author would like to express his 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, the author was very fortunate to receive the guidance and encouragement from his supervisor.

ABSTRACT

We present a prototyping mobile application for indoor navigation using image recognition to locate user's position in the indoor environment and calculate a appropriate path from the current location of user to the desired location. The user can directly follow the direction of arrow until arrived the target location, the application can able to dynamic generate a new path if the user out of path. We also make use of the sensors in the Smartphone, such as accelerometer, gyroscope and magnetometer to increase the accuracy. Our application is implemented on Android platform, but not limit on Android platform, since our navigation system is not required special features only for Android. We test our prototyping application in a shopping mall and found that our application provides better and clearer location information the paper map.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1. Introduction

1.1 Overview

Navigation system always requested for people to find the direction of destination. Even if in the past, people use compass to point the south direction of Earth in order to find the desired path. Navigation system has evolved in many centuries and still is the most useful system in the world. It was request by many industries such as airport, ships and car. But the original purpose that guides users to their desire location has not changed. It is crucial and helpful for users when they are experiencing in an unfamiliar environment. For example, everyone has experiences that go to a large airport, there are 100 gates to aboard, and you maybe get lost in the airport. Moreover, with the invention and popularization of mobile devices, navigation system has become handier and mobilized. The user can install navigation application on their mobile phone. As a result, navigation system is always a good assistant for tourists to use when travelling. Currently, the main type of navigation system is the outdoor navigation system. It is low cost than other outdoor navigation system, easy to use and works fire under all weather. The working principle of outdoor navigation system is to make use of the Global Positioning System (GPS) to achieve high accuracy of positioning and navigation. It is the best solution for outdoor navigation system. Nevertheless, due to the interference of building, the signals of GPS will not accuracy pointing direction and the current position. Thus, it is not the case in the indoor environment.

The indoor navigation system will be acting a great role in guiding travellers inside airport, mall, museum, etc. It will be extremely useful and helpful for travellers to find the right path with short time in an unfamiliar indoor environment. Yet, GPS does not work effectively and precisely in indoor environment as GPS relies on microwaves from the satellite to locate a user's location. Microwaves are attenuated and scattered by obstacles and hence affects the performance of the GPS. Therefore, we have to use other method to implement indoor navigation system.

Many researchers proposed various methods as an alternative to be used in indoor environment. These methods are Wi-Fi fingerprinting, dead reckoning, infrared technology and marker image recognition, etc[[1, [8]. However, some of them require pre-install equipment and maintenance. In addition, network signals can also be easily affected in indoor environment with the presence of a great amount of obstacles in a tide area. Hence, an offline-based system without the assistance of microwaves is a better solution. It is also the objective of implementing an indoor navigation system using image features recognition in offline environment.

Since there are many indoor navigation system are implemented but most of them are not extensively used in indoor environment. Moreover, we realized many people are not familiar to use paper map and then get lost all the time. For example, the first time I go to new campus of University of Macau, it was harder to find the right classroom in E11 building since there are many classrooms. We want to implement an indoor navigation system can be widely used in indoor environment. It can help the user can find their desired location within shortest time and quick familiar the environment.

1.2 Objectives

We want to implement a mobile phone application that is a stable, easy-to-use offline indoor navigation system which helps user to reach the desired destination. The system uses image recognition technique to locate the user. The user just needs to use their camera of mobile phone to point to environment. Thereafter, it generates a route with the stored on-device map data. Finally, it shows the proper direction with an arrow in a combination with the real world environment leading users to their ideal destination. It can be install in any Android mobile phone without can precondition except the version of Android. We want to help the people who not familiar with paper to find the location they want.

Our application is divided into three parts, positioning, navigation and direction rendering as show in Figure 1. The positioning mainly focus on recognize user's position. The navigation mainly focus on generate a proper path for user. The directions mainly focus on how to display a direction of path properly.

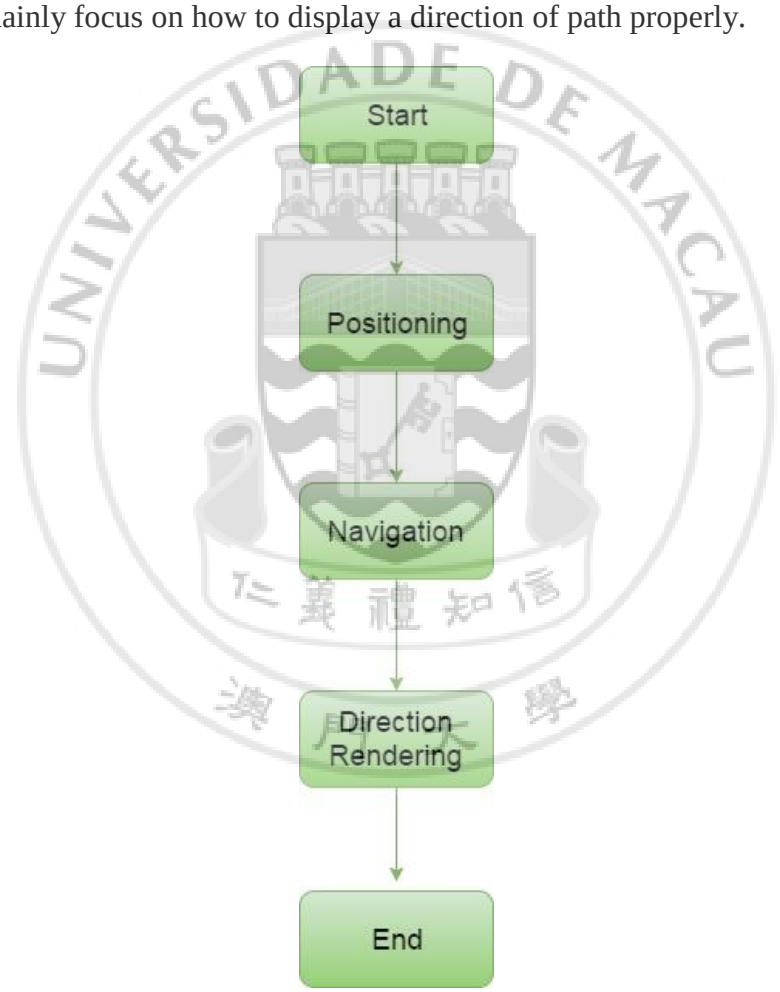


Figure 1: Flowchart of our application

In order to accomplish the final product, there are several stages need to achieve. The first stage is data collection which is most difficult part in our project. Because of our application use image to locate user's position, the image data will be larger. The map data is also required for our application. As our application needs to calculate the path from current position to user desired destination, we have to currently and precisely implement the map on Android device. Thus, we need to formulate the map and make

it suitable for our application. The second stage is database construction which converts the data into database in our system. The third stage is user interface design and implementation which design the proper user interface of our application. We want the user do not have confusing during using our application and provide a clear direction to user. The fourth stage is route search and rendering which calculate the proper path and display the direction of destination. The fifth stage is evaluation and optimization which we testing our prototype application and discover some issue in our system and solve the problem with compass function in mobile phone. I am responsible for design and develop the user interface, map construction and compass function implementation.

CHAPTER 2. Related Work

The past literatures have proposed various approaches to achieve indoor navigation system without using GPS technology. Those the indoor navigation system can be divided into following parts: user's location detection and route representation.

The user's location detection mainly focuses on recognizing the user's location and information of current environment. Many literatures have proposed different methods to detect the user's location. These methods can also be categorized into visual method or non-visual method[[4]. The non-visual methods usually use signal to locate user's position such as infrared technology, Wireless technology and radio fingerprint system[[1], etc. Infrared and Wireless technology can provide non-visual tracking for the user, but the equipment are expensive and they require pre-installed special hardware equipment to emit signal[[1, [8, [9]. Davide et al.[[2] proposed another non-visual method using Dead Reckoning (DR) technology to estimate the user's movement and direction. Even though DR technology relies on the sensors of mobile phone, the accuracy is quite low and hence requires periodically recalibrating[[2] to enhance the accuracy.

The visual method uses camera to detect user's location. Therefore, visual method commonly uses marker-base system to achieve the detection. Comparing to the non-visual method, visual method has a relatively lower cost. It does not require any additional device but a mobile phone with camera and it is more suitable for showing directions in user's perception. Konrad et al.[[10] and Jongbae Kim, Heesung Jun[[12] also proposed an alternative approach to track the user's position by taking photos of the current location. Then, the captured photos are compared to the photos in online large-image database, to retrieve location information. Obviously, network connection is required in such approach. However, network signal is weak or even not available in some indoor environments.

In marker-based system, Fiducial markers are usually used[[4] because it can achieve detection at a lower cost than non-visual method and easier to install. Manfred et al.[[15] successfully achieve continuous tracking the user's location with large amount of Fiducial markers. The location information can be retrieved when the corresponding Fiducial marker is scanned with a camera. However, such implementation requires pre-marked Fiducial markers, i.e. the markers need to be printed and pasted on wall in the indoor environment. Furthermore, calibration is harder to maintain when the area of indoor environment is larger. Hence, in the implementation of this project, feature-based recognition provided by CraftAR and compass function is used to improve efficiency and reliability of the implementation.

The route representation mainly focuses on presenting the direction and route to the destination. Many literatures had proposed various representations for the navigation path. Christian et al.[[3] discovered that 2D or 3D map is not necessary for user using indoor navigation system. Since the sketch of direction is enough for the user. Alessandro et al.[[13] and A.J.Bernheim et al.[[14] using a sequence of activities to guide the user to destination, such as "walk 10 steps north", "walk 2 floors down" etc. This method could provide good and concrete instructions for short distance destination, which make user clearly know the distance of target location. However, if

the distance between starting position and destination is longer, the number of activity will also be larger; user may be confused with those numerous instructions. In addition, Hans et al.[[11] use a different approach that directly placed the route on the map through AR. The only requirement of the user is to scan the map, and then the phone will show the path above the map image with Augmented layer in phone. Martin et al, Rehman and Andrew[[4, [5, [7] also find the AR is a good and straightforward representation to guide the user to destination. Andrew Zhong[[7] use AR to generate an arrow to guide the user, which is a more straightforward method to show the direction.

Therefore, our project group summarized the result of those literatures and carried out an integrated approach to achieve the indoor navigation task. We have decided to implement a mobile phone application which detects user's current location by using image features recognition and guide user to destination through displaying guidelines on the camera view of the device, just like AR technique but not actual AR.

CHAPTER 3. Design

3.1 User Interface

User interface of application is always a crucial factor to decide good or bad usability and user experience of an application. A terrible user interface cause huge problem for the user to operate the application even the application has excellent functionality. User interface is also the most important part for our application since our application is map application to offer location and path information. It provided a clearly and consistent viewing to the user and the user can easy to intact with the application without unnecessary confusion.

Figure 2 shows the operation flowchart for our application interface. The user interface of our application is divided into 2 parts, destination selection screen and navigation screen, which based on our information architecture. With the user open our application in mobile phone; the user can see welcome screen with our designed logo for our application. Then, the user have to select a desired destination, which can be done by select a location directly or type the key words in the search bar we provided. If the destination was decided, the screen will open camera of mobile phone, and the application is ready to recognize the position of user.

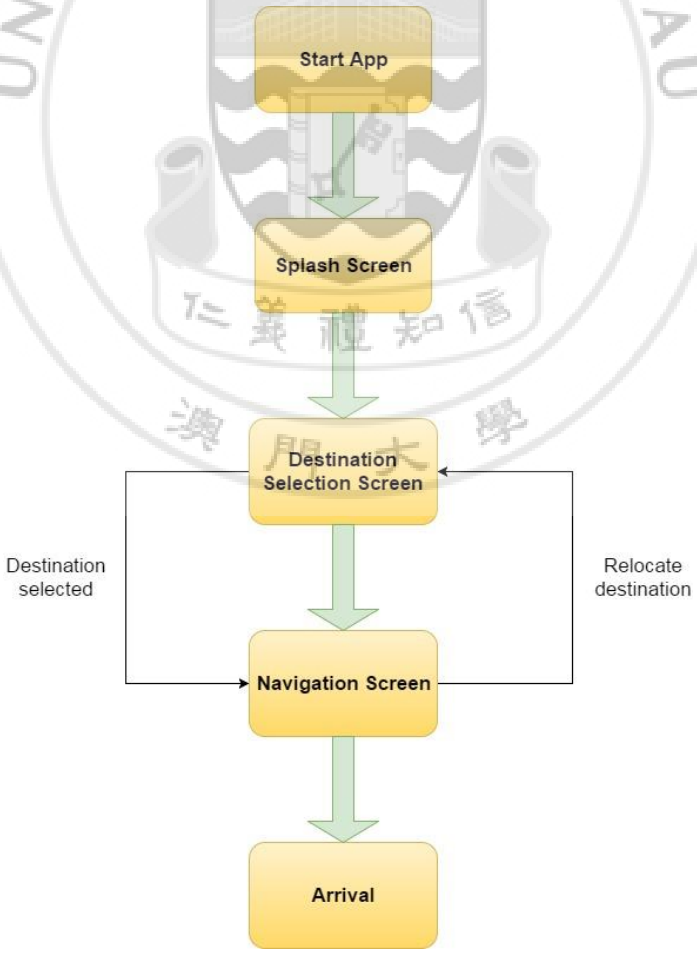


Figure 2: Flowchart of user interface

Figure 4 shows the destination selection screen of our application. We collected all the shop names in the testing shopping mall, and stored into the database. This screen shows list of shop names for the user to choose a desired location. But when we test our prototyping application, we noticed that the list could be very long since there are many shops in the list, searching whole list is a terrible idea for the user to select the destination. In order to improve the efficiency of the destination selection, we add a search function to the selection system. The user can type the key words about the destination and the function will provide some possible location for the user.

Figure 3: Splash Screen

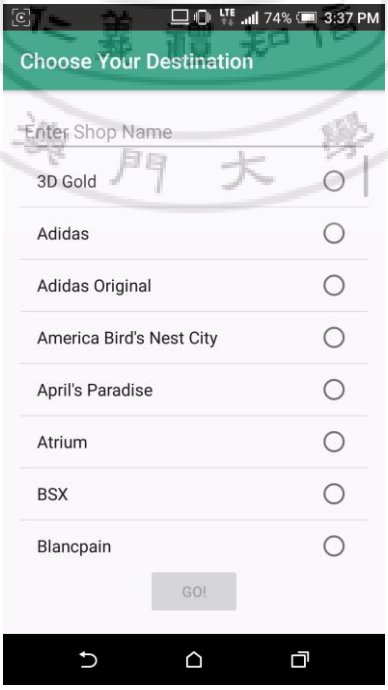


Figure 4: Destination Selection Screen

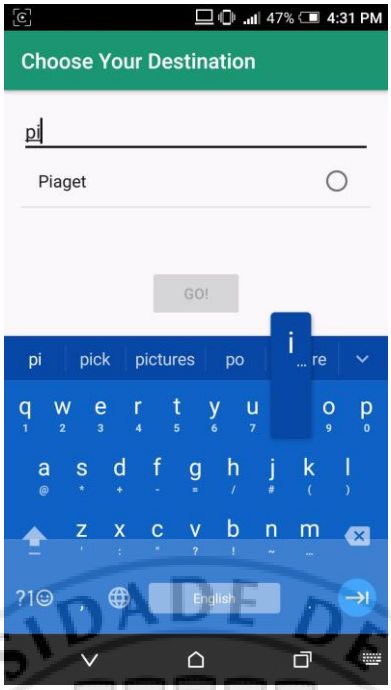


Figure 5: Destination Selection Screen - Search Function

After the destination is confirmed, navigation screen will show a virtual arrow on the screen when the position of the user is known (as Figure 6 (b)). The green circle at the right down corner of Figure 6 (a) is a destination selection button, which can open a selection screen for the user to relocate the destination. The user can change destination whatever he/she want. With the destination changed, the application will regenerate the path for the new destination. The navigation also pops up a arrival message when the user reach the destination.

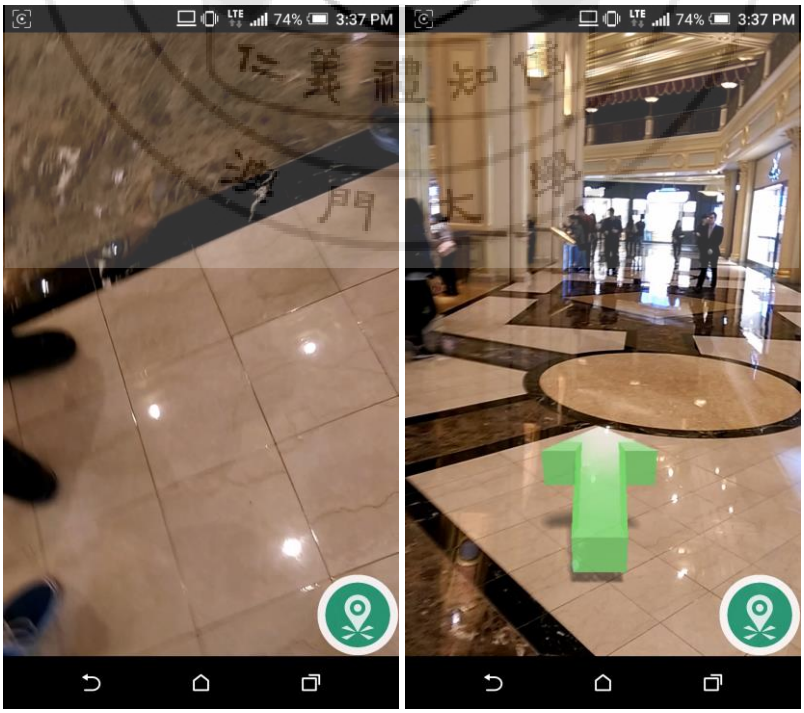


Figure 6: Navigation Screen – (a), (b)

3.2 Map

The outdoor navigation system usually provided maps for user to view their current location and show the direction of destination. Some of them provide a normal map just like the paper map, and some of them provide a 3D map for viewing. However, we decided not to display digital maps for the user in our application since we believed using an arrow to point the direction is more directly and some people may have terrible reading maps. Therefore, we only use the map for path searching and the user has not to read maps.

The most challenge of creating map is how to collect the data from the environment. In our testing location, they have provided a handbook which included paper maps and shops information to tourists. We only use one in four of the map as our map in application because of the size of testing environment is larger.

Figure 7: Divided map

Figure 7 shows the map of our experimental location, all the paths are called as street, such as Opera street and Lion street in the mall. Moreover, we divided the map into many small streets so that can create graph stored in application. We tried using a street as block and shops in this street were linked to this block. But we realized the some street was very long and many shops in there, causing the accuracy of map lower, and also taking into account the user needs to do the only thing is to follow the direction of arrow shown. Thus, we decided to separate all the streets in the shopping mall into many smaller substreets.

Figure 8: Each shop is assigned to a corresponding substreet

Nevertheless, the user's destination is always a shop, not any substreet. The shop information is not stored in our digital map but in our database. However, we discovered that we only need to know the corresponding substreet to the user's destination shop. More information stored in our map will only make the map more complex. Hence, we decided to store the shop information outside the map. The digital map structure remains the same, but shop and substreet assignment is recorded. Each shop is assigned to a corresponding substreet. Once the user destination shop is known, the corresponding substreet can be derived and the route calculation will be started.

The map has to store into our application for searching the shortest path. We use a graph to store the map, but there are many type of graph such as directed or undirected and has weighted or not. So we choose weighted undirected graph to store the map data. In order to generate the graph from the map, we created a database to store the relation of substreet and shops information.

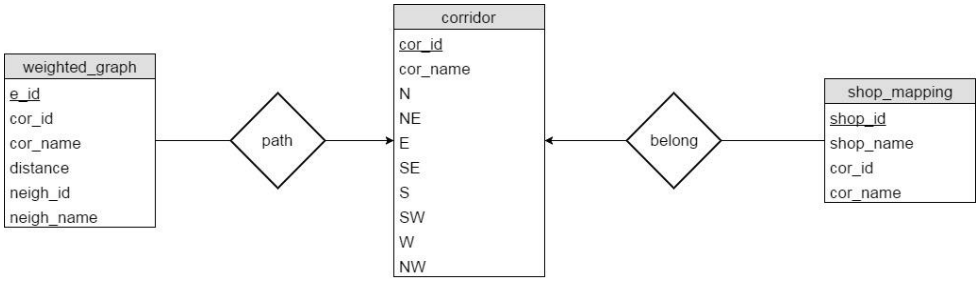


Figure 9: Database Structure Chart - ER diagram

Figure 9 shows the structure of our database. There are 3 main tables in our database. The corridor table store the corridor information and their 8 orientations neighbours. The 8 orientations neighbours are used in image recognition and the orientation is referring to orientation of shopping mall. The shop_mappping table store the shops in each substreet, used in getting corresponding substreet for user's destination shop. All the shops only assign to one of substreet. But unfortunately some of shops are not exist anymore since our shop data are came from the handbook of shopping mall. We have to recheck these shops and correct the information. The weighted_graph table store relationship between each substreet (corridor). The distance of each substreet is default to 1. Some distance is 3 if there is a bridge between the substreet. Because the bridge has stairs and will take more time to go through. This table and corridor table are mainly tables for generating the weighted undirected graph.

3.3 Compass

The android platform provided many sensors for application development to measure various environmental conditions such as motion, orientation and gestures. We can utilize these sensors to enhance our application. In our first prototype testing, we found the gesture of holding mobile phone is significant factor for image recognition. The gesture will affect the image recognition result, if the mobile phone is not holding in front of the user; the user's environment cannot be recognizing by our application. Since the user's current position not able to update, then the direction of destination also not able to update for a while. It caused the navigation system not functioning unless the user change the direction of mobile phone or change the position standing right now. We can solve this problem with the help of compass function in mobile phone.

Therefore, we can use the last direction of destination as marker and mark on the compass. Because of the direction of destination and compass are refer to the same coordinate system, the direction of destination and compass pointing direction are the same. The direction of compass can be the temporary direction for the user to follow. The arrow also will change base on the user facing direction. The direction will be update when the current position of user is known, which means the image of current position is recognized. This solution is used in case the recognition system is not working, and the compass can help to guide the user for short time.

In order to build a compass function in our application, we have to understand the principle of compass on mobile phone. For Android platform, there are 3 sensors that can use to determine the direction of the mobile phone: magnetometer, gyroscope and accelerometer. The magnetometer is to measure the Earth's geomagnetic field on three physical x axis, y axis and z axis. The gyroscope is to measure the rate of rotation of mobile phone on three physical x axis, y axis and z axis. The accelerometer is to measure the acceleration force include gravity on three physical x axis, y axis and z axis. The compass function is combined with the information from these sensors and calculated the orientation of north. The accelerometer determined the position of axis base on mobile phone position, and magnetometer measure the Earth's geomagnetic field around the mobile phone; combined with the rotation degree on horizontal direction by gyroscope we can know other directions. We also can utilize the accelerometer to determine compass value when mobile phone is tilted.

Figure 10: 3 Axis of Android mobile phone

Figure 11 shows the angle distribution of compass in 8 directions N, E, S, W, NE, SE, SW, NW. These directions are distributed on average. Because of the image reorganization phase we have to divide the compass into 8 directions. In order to match the image recognizes result.

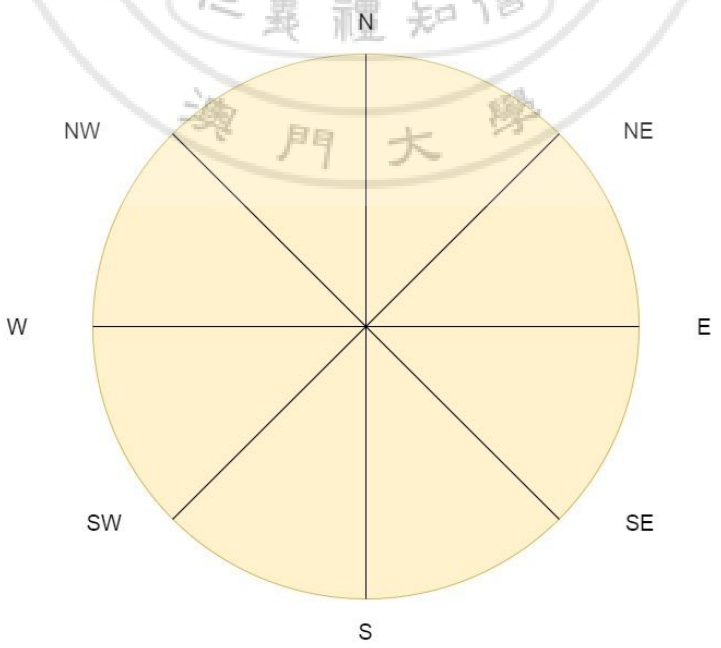


Figure 11: Angle Distribution of Compass

CHAPTER 4. Implementation

Our application is developed on Android platform with Android Studio. We use different tools to achieve these functionalities.

4.1 User Interface

There are 3 user interfaces we have to implement for our application, splash screen, destination selection screen and navigation screen.

4.1.1 Splash Screen

In order to provide better user experience of our application to the user, the first impression of our application is important. We had designed a custom logo for our application showing in the splash screen. The implementation of splash screen can be divided into 3 parts.

First part, we created a activity named "SplashScreenActivity.java", which is the first activity active in Android application. It is responsible for setup all the setting and CraftarAR environment, include the layout setting. We can use onCreate() function to setup the layout design by call setcontentView() to specify the layout.

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.splash_screen);

        CraftARSDK.Instance().init(getApplicationContext());

        //Initialize the Collection Manager
        mCollectionManager = CraftAROnDeviceCollectionManager.Instance();

        //Initialize the Offline IR Module
        mCraftAROnDeviceIR = CraftAROnDeviceIR.Instance();


        ...
}
```

Second part, we have to create a design layout file for layout. This file is written by xml and put in res\layout\ folder. We add a ImageView widget to display the logo.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@android:color/white"
    android:id="@+id/splash_screen"
    tools:context=".SplashScreenActivity" >

    <ImageView
        android:layout_width="220dp"
        android:layout_height="fill_parent"
        android:src="@mipmap/logo"
        android:id="@+id/logo"
        android:layout_centerHorizontal="true" />
</RelativeLayout>
```

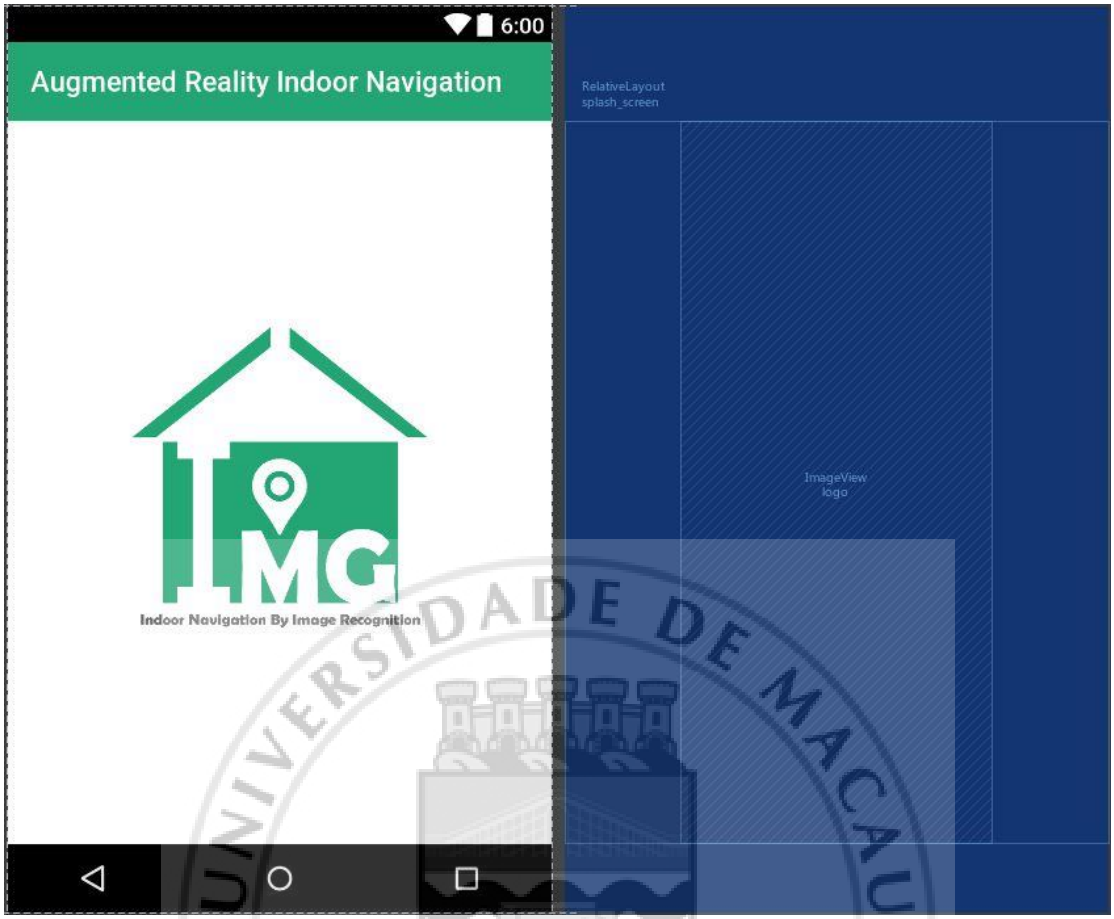


Figure 12: Splash Screen Design

Third part, we register this activity in AndroidManifest.xml. Since we need to let this activity start first, we also have to add intent-filter to indicate this activity start first.

```
<activity android:name=".SplashScreenActivity">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

4.1.2 Destination Selection Screen

The process of adding destination selection screen is same as splash screen. But we have to get all the shop names from the database.

First part, we created a activity named "ChoosingActivity.java". It is responsible for listing all the shop names and get the shop name as destination pass to "MainActivity.java". We can use onCreate() function to setup the layout design and setup data for display.

```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_choosing);
        setTitle("Choose Your Destination");

        ...
}
```

We can setup database and get all shop names by calling getShopName() function

```
final DatabaseAccess databaseAccess = DatabaseAccess.getInstance(this);
databaseAccess.open();
List<String> ShopName = databaseAccess.getShopName();
```

Then, we can add to listView widget to display all the names.

```
adapter = new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_single_choice, ShopName);
this.listView.setAdapter(adapter);
```

We also add search function to listView by using filter.

```
searchtext.addTextChangedListener(new TextWatcher() {
    @Override
    public void onTextChanged(CharSequence cs, int arg1, int arg2, int arg3) {
        // When user change the text
        adapter.getFilter().filter(cs);
            }
        ....
});
```

Second part, we have to create a design layout file for layout. This file is written by xml and put in res\layout\ folder. We add a EditText, ListView and Button widget to construct the interface.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_choosing"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:descendantFocusability="beforeDescendants"
    android:focusableInTouchMode="true"
    tools:context="com.catchoom.test.ChoosingActivity">

    <Button
        android:text="Go!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/button"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:enabled="false" />

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="textPersonName"
        android:ems="10"
        android:id="@+id/editText"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:hint="Enter Shop Name" />

    <ListView
        android:layout_width="match_parent"
        android:id="@+id/shop_list"
        android:layout_height="match_parent"
        android:choiceMode="singleChoice"
        android:clickable="true"
        android:listSelector="#bcf3c3"
        android:saveEnabled="false"
        android:layout_below="@+id/editText"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:layout_above="@+id/button" />
</RelativeLayout>
```

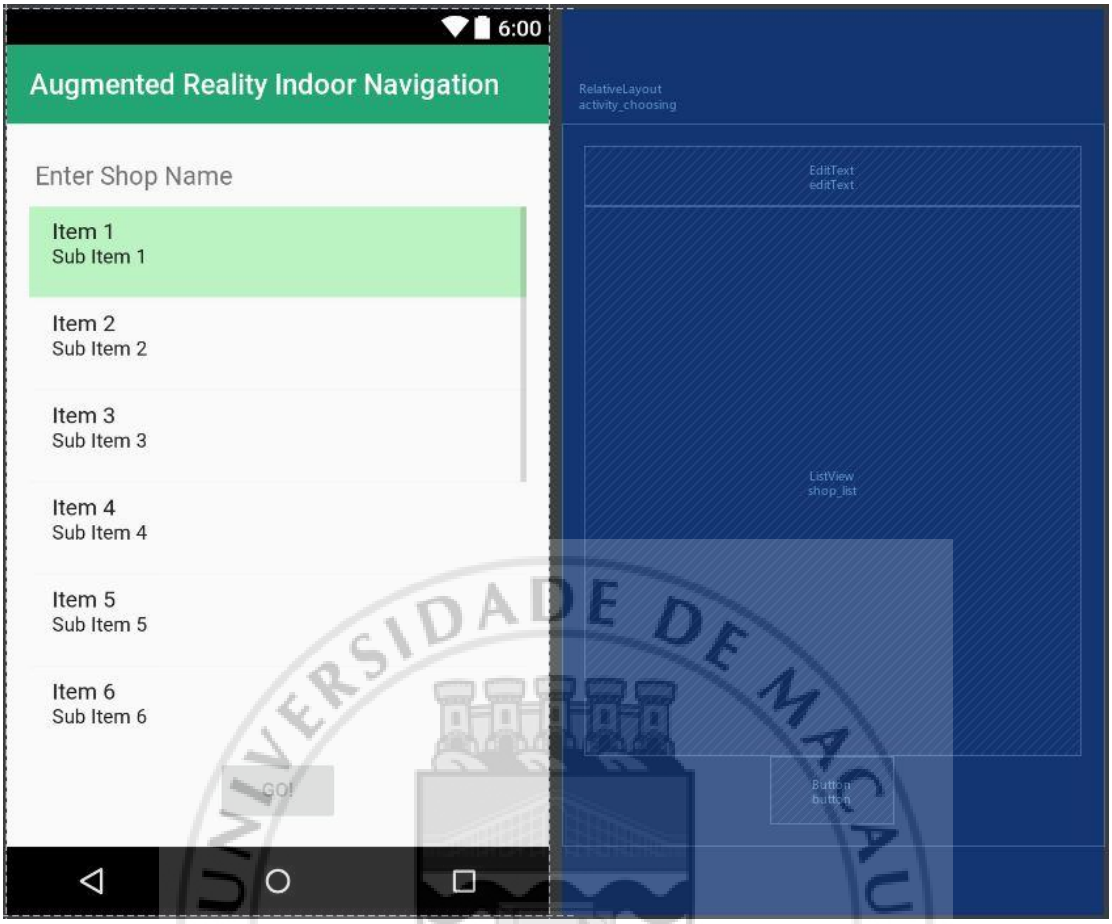


Figure 13: Destination Selection Screen Design

Third part, we register this activity in AndroidManifest.xml.

```
<activity android:name=".ChoosingActivity"></activity>
```

4.1.3 Navigation Screen

The navigation screen is mainly interface for our application. The interface design is simple and clear for the user to see the direction of destination.

First part, we created a activity named "MainActivity.java", which is the main activity in our application. It is responsible for path searching, image recogniation and generating destination direction. We can use onCreate() function to setup the layout design by call setcontentView() to specify the layout in onPostCreate() function.

```
public void onPostCreate() {
    View mainLayout = getLayoutInflater().inflate(R.layout.activity_main, null);
    setContentView(mainLayout);
    ...
}
```

Second part, we have to create a design layout file for layout. This file is written by xml and put in res\layout\ folder. We add a EditText, ListView and Button widget to construct the interface.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/activity_main"
```

```
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MainActivity">

        <com.craftar.CraftARCameraView
            android:id="@+id/camera_preview"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:layout_alignParentTop="true"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true">

        </com.craftar.CraftARCameraView>

        <TextView
            android:id="@+id/pole"
            android:layout_width="60dp"
            android:layout_height="20dp" />

        <TextView
            android:layout_width="60dp"
            android:id="@+id/degree"
            android:layout_height="20dp"
            android:layout_marginTop="18dp"
            android:layout_below="@+id/pole"
            android:layout_alignParentLeft="true"
            android:layout_alignParentStart="true" />

        <TextView
            android:layout_width="20dp"
            android:layout_height="20dp"
            android:id="@+id/result"
            android:layout_alignParentRight="true" />

        <Button
            android:layout_width="75dp"
            android:layout_height="75dp"
            android:id="@+id/button2"
            android:layout_alignParentRight="true"
            android:layout_centerInParent="true"
            android:background="@mipmap/button" />

        <ImageView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:src="@mipmap/empty"
            android:id="@+id/imageView"
            android:adjustViewBounds="false"
            android:layout_below="@+id/degree"
            android:layout_centerHorizontal="true"
            android:layout_marginTop="199dp" />

    </RelativeLayout>
```

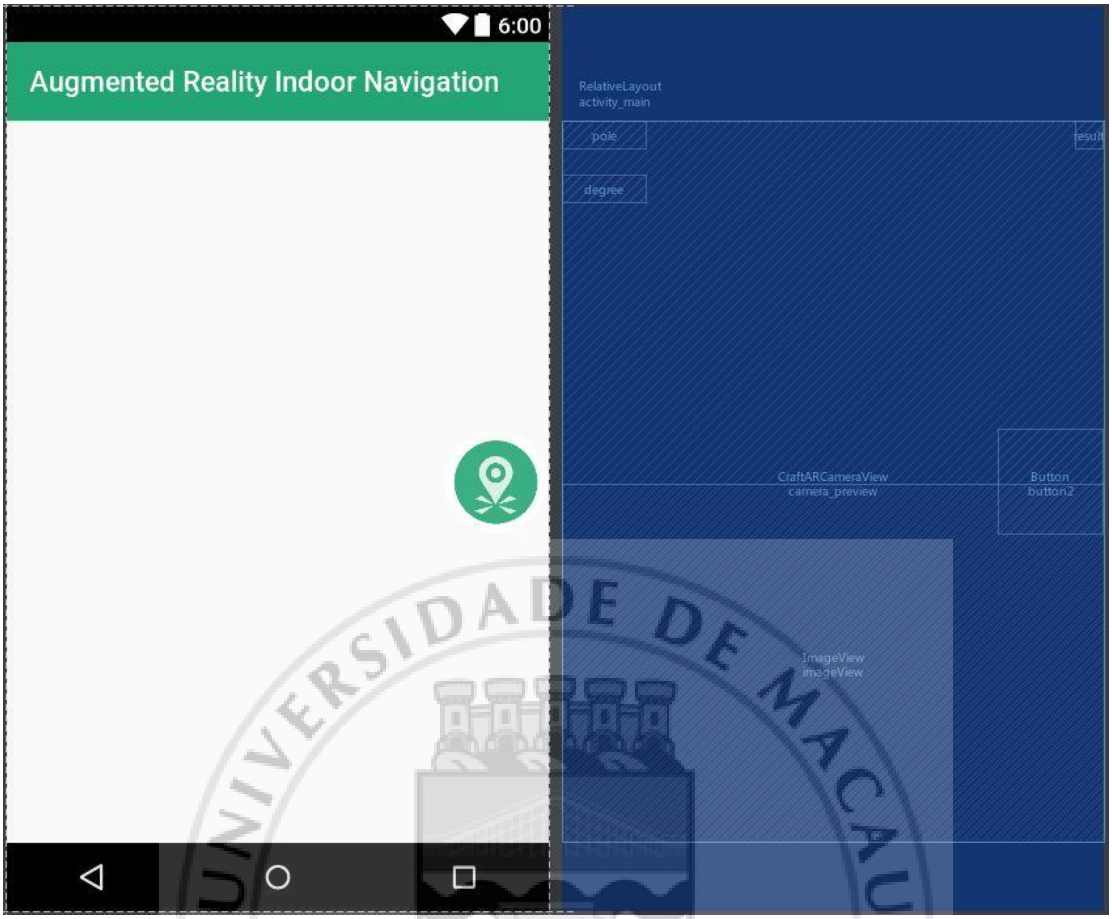


Figure 14: Navigation Screen Design

Third part, we register this activity in AndroidManifest.xml.

```
<activity
    android:name=".MainActivity"
    android:screenOrientation="portrait"></activity>
```

4.2 Map

In our application, we use a graph as map to search the shortest path for the user. But we have to build a database and then using database to construct a graph. In design part, we mentioned the map is divided into many small substreet and shops are linked to them.

4.2.1 Database

We use Android built-in database, SQLite to implement our designs. Our database is mainly constructed by three tables: corridor table, shop_mapping table and weighted_graph table. The corridor table stores all the substreets information to represent as a digital map. Each record contains a substreet ID, name and the corresponding neighbours in 8 directions. Null is marked when there is no neighbour in that direction. Moreover, shop_mapping table stores the shop information. Each record represents a shop and indicates the corresponding substreet. The weighted_graph store the distance and relation between each corridor.

We used 8 directions (North, North Eastern, East, etc.) to state the connection between all the substreets. For example, “Lion street 1” is connected on the south of “Lion street 2” and “Lion street 2” is connected on the north of “Lion street 1”. When all the substreetes are completely connected, a digital map of the entire indoor environment is also completed.

The structure of three tables is show below:

Corridor

```
CREATE TABLE "corridor" ( `cor_id` INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
`cor_name` TEXT NOT NULL, `N` TEXT, `NE` TEXT, `E` TEXT, `SE` TEXT, `S` TEXT,
`SW` TEXT, `W` TEXT, `NW` TEXT )
```

Shop_mapping

```
CREATE TABLE `shop_mapping` ( `shop_id` TEXT, `shop_name` TEXT, `cor_id`
INTEGER, `cor_name` TEXT )
```

Weighted_graph

```
CREATE TABLE "weighted_graph" ( `e_id` INTEGER PRIMARY KEY AUTOINCREMENT,
`cor_id` INTEGER, `cor_name` TEXT, `distance` INTEGER DEFAULT 1, `neigh_id`
INTEGER, `neigh_name` TEXT )
```

4.2.2 Graph

After we build up a database, we use a java graph library JGraphT to generate the graph for our application. We found few java graph library like JGraphT but the JGraphT are better and we thought this library meet our requirements. We created a application to generate the graph.

First we need to open database for access, we created DatabaseAccess.java to handle the database querying. The getAllCorridor() get all the corridor name from corridor table, then store into list.

```
public List<String> getAllCorridor() {
    List<String> list = new ArrayList<>();
    Cursor cursor = database.rawQuery("SELECT cor_name FROM corridor", null);
        cursor.moveToFirst();
        while (!cursor.isAfterLast()) {
            list.add(cursor.getString(0));
            cursor.moveToNext();
        }
        cursor.close();
        return list;
}
```

The getAllEdge() get all the relation between each corridor from weighted_graph table.

```
public Cursor getAllEdge() {
    Cursor cursor = database.rawQuery("SELECT cor_name, neigh_name, distance
FROM weighted_graph", null);
    return cursor;
}
```

The mainly process for generating graph is in createStringGraph(). The SimpleWeightedGraph object is provided by JGraphT, it contain several methods to operate such as addVertex(), addEdge() and setEdgeWeight(). We read all the corridor from database, add to graph as vertex and read all the relation from weighted_graph to graph as edge, then set weight for each edge.

```
private Graph<String, DefaultWeightedEdge> createStringGraph() {
    SimpleWeightedGraph<String,
    DefaultWeightedEdge> g = new SimpleWeightedGraph<String,
DefaultWeightedEdge>(DefaultWeightedEdge.class);

    //open the database
    DatabaseAccess databaseAccess = DatabaseAccess.getInstance(this);
    databaseAccess.open();

    //add vertices
    List<String> allcorridor = databaseAccess.getAllCorridor();
    for (String name : allcorridor) {
        g.addVertex(name);
    }

    // add edges
    String from;
    String to;
    Integer weight;
    Cursor cursor = databaseAccess.getAllEdge();
    cursor.moveToFirst();
    while (!cursor.isAfterLast()) {
        from = cursor.getString(0);
        to = cursor.getString(1);
        weight = cursor.getInt(2);
        g.setEdgeWeight(g.addEdge(from, to), weight);
        cursor.moveToNext();
    }

    //close the database
    databaseAccess.close();

    return g;
}
```

We write the graph object to file for later implementation.

```
private void SaveGraph() {
    //Intent intent = new Intent(Intent.ACTION_OPEN_DOCUMENT_TREE);
    //startActivityForResult(intent, 42);
    try {
        File file = getAlbumStorageDir("Graph");
        System.out.println(file.toString());
        File filename = new File(file, "graph_FYP.amr");
        FileOutputStream fos = new FileOutputStream(filename);
        ObjectOutputStream out = new ObjectOutputStream(fos);
        out.writeObject(Graph);
        out.close();
    } catch (IOException ex) {
        ex.printStackTrace();
    }
}
```

4.3 Compass

The design for the compass function is mentioned at Design section. We will utilize the sensors in mobile phone to accomplish the work. However, the Android SDK also provided another sensor to archive the orientation of mobile phone but this method is deprecated in current SDK edition. Android still provided two method to archive the orientation except the above one method. First method is using SensorManager.getOrientation() to archive the orientation. Second method is using TYPE_ORIENTATION to archive the orientation. The first method is more precision and is recommended by Android. Thus, we use the first method to implement our compass function.

The data return from getOrientation() is a float type array include values on 3 axis. As show in Figure 13. The data structure is show as follow:

- values[0] – azimuth, rotation around Z axis. We use this value to estimate the compass orientation. We have to normalize the value before using because of the value range is -180~180 degree. It means the north is 0 degree, the east is 90 degree, the south is 180 or -180 degree, the west is -90 degree.

- values[1] – pitch, rotation around X axis. We not care about this value.

- values[2] – roll, rotation around Y axis. We not care about this value.

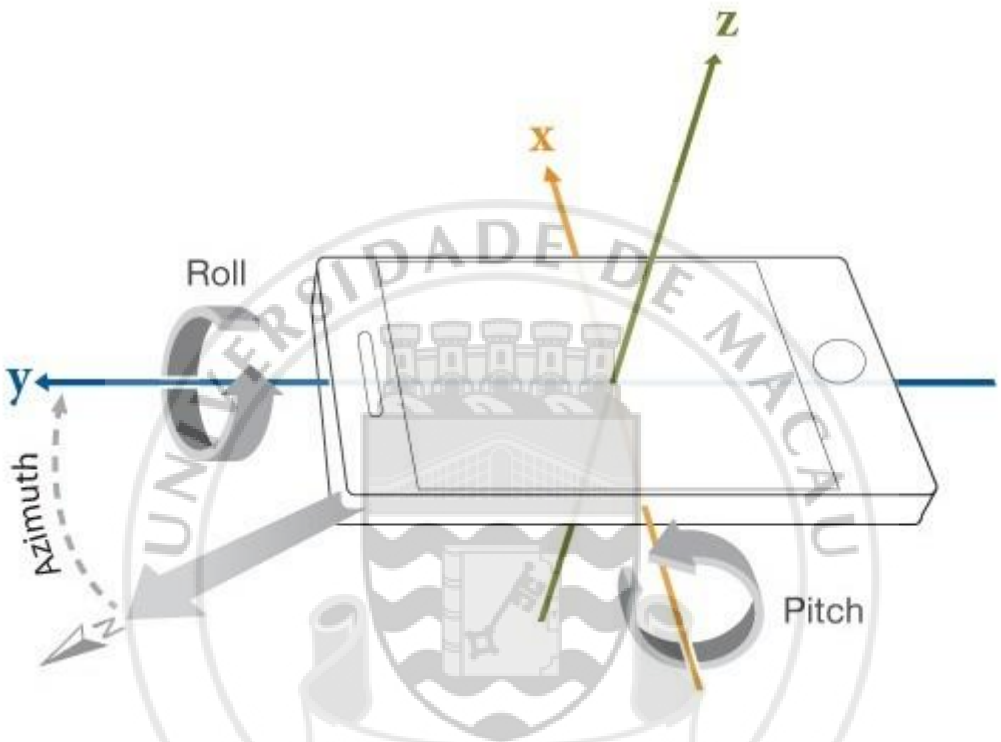


Figure 15: 3 axes on mobile phone

The interface of getOrientation() is show as follow:

```
public static float[] getOrientation (float[] R, float[] values) {...}
```

There are two parameters in this function, R is rotation matrix and values are the result of this function. We just want to get the values in parameter but the return values from the function. We cannot get the rotation matrix directly, so we use getRotationMatrix() to retrieve the rotation matrix R.

The interface of getRotationMatrix () is show as follow:

```
public static boolean getRotationMatrix (float[] R, float[] I, float[] gravity,
float[] geomagnetic) {...}
```

There are four parameters in this function. The R is what we want to retrieve in this function. The gravity is data which get from the accelerometer. The geomagnetic is data which get from the magnetometer. Now we have rotation matrix to get the orientation of mobile phone through getOrientation().

There are many noises in the data since the data are directly retrieved from the sensors. These noises will cause the compass result unstable and inaccurate. We can apply

appropriate technique to solve this problem. We use a low-pass filter to filter out the noise with proper alpha value.

```
private float[] low_pass_filter(float[] input, float[] output, float alpha) {
    if (output == null)
        return input;
    for (int i = 0; i < input.length; i++) {
        output[i] = output[i] + alpha * (input[i] - output[i]);
    }
    return output;
}
```

After testing, we had chosen 0.4 as alpha value for geomagnetic, 0.2 for accelerometer. We also use remapCoordinateSystem() to rotate the rotation matrix when the mobile phone is tilted.

CHAPTER 5. Testing

We have chosen a shopping mall as our testing environment. There are several testing stages in our application. First stage is to test the image recognition system, this system is the mainly part of our application. We built a simple application on mobile phone and then test it accuracy of image recognition. This application will simply display the name of location after it recognizes the environment. The second stage is test the JGraphT library for creating graph and finding shortest path. The third stage is testing the system on mobile phone after these parts are integrated together.

We test our final application on shopping mall. The type of testing mainly divided into two parts, environment recognition and navigation. The environment recognition testing is important for our application since we need the user's current position to calculate the path. We will randomly select shop and corridor to test the system able to recognize or not. With successful recognize the user's position, we will test the path is correct or not for navigation. We collect the errors of system during the navigation. We also prepare different destination to test the system from different position to see the direction of destination is correct or not.

CHAPTER 6. Evaluation

We have chosen a shopping mall as testing environment. In order to calculate the shortest path between current position and destination for the user. We have to recognize the user current position before we calculate the shortest path to the destination. This section shows the rate of success recognition of our application. We have selected 22 locations for recognition evaluation.

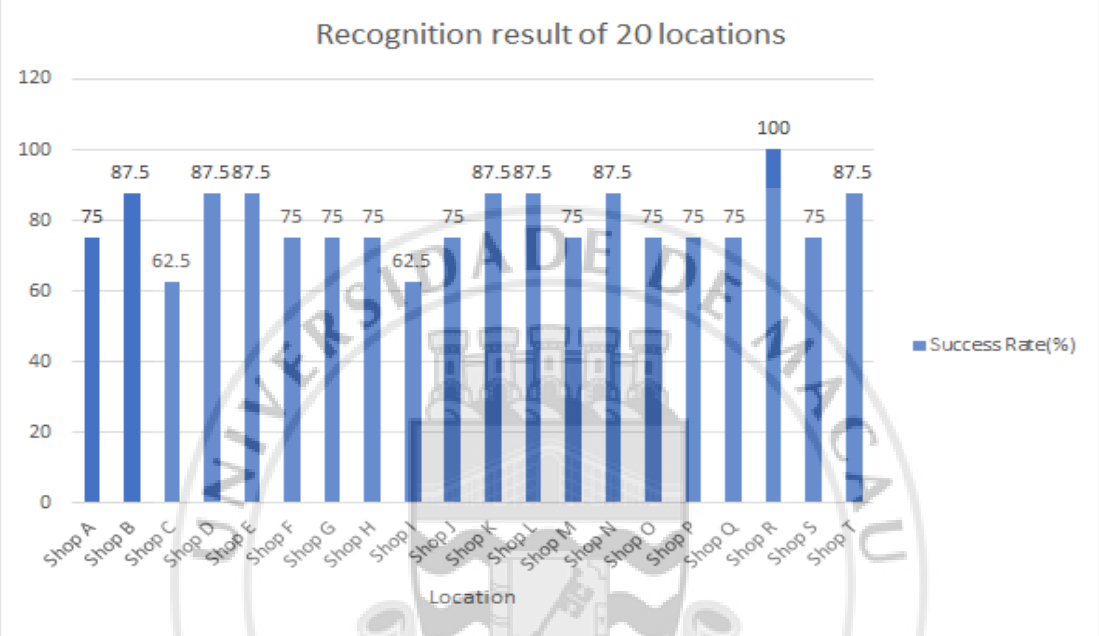


Figure 16: Recognition test result

Figure 16 show the recognition result of 20 locations. The 20 locations are selected randomly for testing. As a result, the recognition performance is lowered and the average success rate of recognition is 79.37%. There are two mainly reasons of the recognition failures. First reason is that the user standing on the edge of the street and the captured image is extremely different from the collected image data. Second reason is that a lot of people enter in the view of the camera and cover up the feature of environment. It will affect the image recognition system to locate the user current position. From this testing result, we can improve the success rate of image recognition by increasing the quantity of image data of environment in different angle and position.

Table 1: Navigation test

From	Shop A	Shop Q	Shop R	Shop L	Shop D	Shop M	Shop S	Shop C	Shop F	Shop B	Shop H
To	Shop T	Shop B	Shop H	Shop E	Shop Q	Shop A	Shop E	Shop S	Shop O	Shop I	Shop T
Average success rate	80%	90%	100%	90%	100%	100%	100%	80%	100%	100%	100%
Total Average Success Rate									95%		

Table 1 shows the testing result of navigation in 11 different destinations and starting location. The starting location and destination is randomly selected from the shopping mall. We repeated the testing 10 times for each location. The testing result shows that the performance of our system is satisfy and the error rate is low. The application completed the navigation without fatal error. Table 2 shows the comparison of our application with other indoor navigation methods. Our application can continuous locate the user's position during the user moving to destination. Our application also not requires network connection and pre-install equipment or image maker on the environment. Image marker method requires the user to scan the image marker with mobile phone [[7]. Wi-Fi Fingerprinting method use electromagnetic signal which are easily affected in indoor environment. Our purposed method does not have such constraints and result in a low error rate according to our test.

Table 2: Compare results with existing indoor navigation methods

	Image recognition	Wi-Fi Fingerprinting[[*6*]	Image Marker[[*7*]
Continuous Localization	✓	✓	✗
Network Requirement	✗	✓	✗
Positioning Constraint	✗	✓	✓
Require Pre-installed equipment	✗	✓	✓
Error Rate	5%	10%	2%

CHAPTER 7. Discussion

The test result is satisfied for me to use the application. I through this is a great navigation application which can give clear direction about the destination. It is very useful for me to use in unfamiliar environment such as airport since there are many signs on the environment. I will get lost in this environment or take some to understand the direction and relation about signs. The operation of our application is simple and easy to use. The result of recognition is high and a lot of place can be recognized. However, there are several places we can do better for our application. First place is the collection of image is take picture one by one manually. We can use other technique to capture image from the video and then generate the image feature data automatically. Second place is automatically generating the image feature data. We only can generate image feature data using the website service provided by CraftAR to get the feature. It is necessary for generating larger image feature data in larger environment. Third place is improving the direction rendering. The direction arrow is not actual AR technique but a simple image. If we implement a real AR image to display the arrow will be more attractive and interactive. We also can add some AR object on user navigation screen and let the user can interact with the virtual object such as small game of shop to promote their product. But overall of our application is acceptable for me to use this application.

CHAPTER 8. Ethics and Professional Conduct

8.1 Image Data

Since our application development will take image data from environment such as shopping mall. The face of people will also capture in the image. But the actual image will not use in our application. We only use these images to generate the image features and delete these images after feature generation. However, these images also can be taken by other people. The face of people is not useful in our application development. Thus, we will blur each people face in the image for personal privacy.

8.2 Library Tools

Since we use an open source library to create graph for our application. This library is JGraphT and its library license is LGPL. Therefore, we can use this library to create the graph and find the shortest path in the graph.

CHAPTER 9. Conclusions

The barrier in the building or building itself affected the signal of GPS and other navigation system base on signal and lower down the effectiveness of GPS and these systems. As a result of this reason, to achieve a stable and accurate indoor navigation system is huge challenge in these years. In this paper, we have proposed an image-based indoor navigation application to overcome this challenge. The core of our application can be mainly divided into two parts: positioning and navigation. The indoor positioning method is accomplished by image recognition. The image of the user's current position will be captured from camera of mobile phone and compare with the pre-collected image data in our application database. The image recognition result is achieved by CraftAR SDK, the most similar image can be derived out from the collection. We can identify the user's current position and direction by matching the index of current result image. The first recognized direction is marked as reference marker for compass function to provide continuous navigation effect. In addition, the navigation is accomplished based on the identified current position, direction and destination location id to generate the proper path for the user. We have chosen Dijkstra 's algorithm to perform the shortest path search on the map graph of the indoor environment. The algorithm returns a list containing a path from the user's current position to destination. The positioning processes are performed continually during the navigation. The Dijkstra's algorithm searching will be executed again once the user's current position index is out of path list. Also, direction arrow is integrated with the real environment on screen to provide user-friendly experience and increase acceptability of the application. It make the user can follow the direction more straightway.

We show the accuracy of the proposed positioning method with amounts of testing base on randomly selecting location from the tested indoor environment. The testing result showed that the image recognition method and collected image data can achieve high accuracy of positioning. Failure results are usually caused in image feature cover up by many people, position that are extremely different from the collected image data or the capture image provide not enough information of the environment. On the other hand, improvement of accuracy performance can be easily increased by increasing the quantity of image data in collection. The advantage of our proposed indoor navigation method is totally free of internet connection. The accuracy of our proposed method will also not be affected by attenuation of electromagnetic waves caused by obstacles. Our future improvement could focus on collecting more image data to increase the success rate, decreasing the time of image recognition and include additional information of the indoor environment in the process of navigation. It could be make the collection of image and generation of image data more automatically.

CHAPTER 10. References

[1] R. Harle. "A Survey of Indoor Inertial Positioning Systems for Pedestrians", IEEE Communications Surveys & Tutorials, vol.15, no.3, pp. 1281 - 1293, Jan 2013.

[2] D. Merico and R. Bisiani, "Indoor Navigation with Minimal Infrastructure", in Positioning, Navigation and Communication: WPNC '07. 4th Workshop on, Mar 22, 2007. IEEE, 2007

[3] C. Kray, C. Elting, K Laakso and V. Coors, "Presenting Route Instructions on Mobile Devices", in Proceedings of the 2003 International Conference on Intelligent User Interfaces, Jan 12-15, 2003, Miami, FL, USA. ACM, 2003

[4] M. Smit and R.J. Barnett, "A comparison of augmented reality indoor navigation systems with traditional techniques" in Proceedings of 2012 Annual research conference of the South African Institute for computer Scientists and information Technologists. 2010.

[5] U. Rehman,"Augmented Reality for Indoor Navigation and Task Guidance". Diss. University of Waterloo, 2016.

[6] A.C. Mohan, "VirtualTag: Enabling Indoor Augmented Reality with Marker-Facilitated Localization", 2014.

[7] A. Zhong, "Improving User Experiences in Indoor Navigation with Augmented Reality", 2014.

[8] H. Liu, H. Darabi and P. Banerjee, "Survey of wireless indoor positioning techniques and systems" in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2007. pp. 1067-1080.

[9] C. Randell and H. Muller, "Low cost indoor positioning system" in International Conference on Ubiquitous Computing. Springer Berlin Heidelberg, 2001.

[10] K. Tollmar, T. Yeh, and T. Darrell, "IDeixis–searching the web with mobile images for location-based information" in International Conference on Mobile Human-Computer Interaction. Springer Berlin Heidelberg, 2004. pp 288-299

[11] H.J. Müller, J. Schöning and A. Krüger, "Mobile Map Interaction-Evaluation in an indoor scenario" in Informatik 2006: Informatikfür Menschen, 2006, Dresden. pp. 403-410.

[12] J. Kim and H. Jun. "Vision-based location positioning using augmented reality for indoor navigation" IEEE Transactions on Consumer Electronic, vol. 54, no.3, pp. 954-962, 2008..

[13] A. Mulloni, H. Seichter and D. Schmalstieg, "Handheld augmented reality indoor navigation with activity-based instructions" in Proceedings of the 13th international conference on human computer interaction with mobile devices and services: Mobile HCI 2011, Aug 30 - Sep 2, 2011, Stockholm, Sweden. ACM, 2011. pp. 211-220.

[14] A.J Bernheim Brush, A.K. Karlson, J. Scott, R. Sarin, A. Jacobs, B. Bond, ... S. Levi, "User experiences with activity-based navigation on mobile devices" in Proceedings of the 12th international conference on Human computer interaction with mobile devices and services: Mobile HCI 2010, Sep 7-10, 2010, Lisbon, Portugal. ACM, 2010. pp. 73-82.

[15] M. Klopschitz and D. Schmalstieg, "Automatic reconstruction of wide-area fiducial marker models" in Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality. IEEE Computer Society, 2007. pp. 1-4.

CHAPTER 11. Appendix

11.1 Project Planning

We had completed early preparation stage of our application. In order to accomplish the final product, there are several stages need to achieve in the future implementation.

1. Data Collection

   a. Image Data Collection

   Since our application use image recognition to locate user's position, we have to enhance the detection accuracy by using different image collection standards.

   b.Map Data Formulation

   As our application need to calculate the path from current position to user desired destination, we have to correctly and precisely implement the map on Android device. Thus, we need to formulate the map and make it suitable for our application.

2. Database Construction

   a.Image Database Construction

   An image database which stores position information and the corresponding image features is needed. The application can then quickly retrieve current location information by comparing features captured with features in the database. However, we need to minimize the size of database by filtering less important features, otherwise the application size will be horribly large.

   b.Map Database Construction

   A map database will be implemented using the formulated map data from the previous stage.

3. User Interface Design and Implementation

Due to the limited screen size, the information provided for the user should be well managed and displayed. An intuitive interface should be designed, and the information should not overlap or contradict with each other.

4. Route Search Algorithm Implementation

A searching algorithm generates the shortest path from user's current location to the destination should be implemented. Existing searching algorithm will be applied in order to confirm the application works correctly. Customization or improvement will be made at last stage.

5. Direction Implementation

User should follow the direction indicated by the generated arrow. The arrow points differently according to the different values of device sensor. Hence, the direction algorithm should be implemented at this stage.

6. Experimentation and Evaluation

Intensive testing will be conducted at this stage. We will improve the accuracy by analyzing the incorrect navigation cases and fix the bugs.

7. Optimization

When the application works properly as we have expected, we will try to optimize it. For instance, narrowing the range of features need to be compared or improving the searching algorithm, etc.

11.2 Hardware and software requirements

We use Android studio 2.2.3 to implement the application. The target Android version is 6.0 Marshmallow.