**AUTHORSHIP**

*”I hereby declare that the work contained in this thesis is of my own and has not been previously submitted for a degree or diploma at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no materials previously published or written by another person except where due reference or acknowledgement is made.”*

*Signature:………………………………………..*

**SUPERVISOR’S APPROVAL**

*“I hereby approve that the thesis in its current form is ready for committee examination as a requirement for the Bachelor of Electronics and Communications Engineering degree at the University of Engineering and Technology.”*

*Signature:………………………………………..*

**ACKNOWLEDGMENT**

I would like to express my sincere gratitude to Assoc. Prof. Dr Tran Duc-Tan who always guides me, points out the mistakes, and gives me the instructions and comments during the time to realize this work. Without his supervising, I would have many difficulties to finish this thesis.

I am grateful to all members of group of project who always facilitate me to do this thesis, answer my questions in a familiar way and share their experience for me as well as make me feel comfortable and better in studying.

I would like to also thank the faculty members and staffs of the Faculty of Electronics and Telecommunication, VNU-UET for their enthusiasm to guide me to for the background of knowledge.

I greatly appreciate the following organization the Micro-electromechanical Systems and Microsystems Department where you did your thesis work, the University of Engineering and Technology.

Finally, I would like to thank my family for staying close to me even in the most difficult days and for always been of encouragement to me.

Sincerely,

Do Van Lam

# **ABSTRACT**

Nowadays, traffic accidents are a big problem. Thousands of people die every year in traffic accidents, causing damage to people and assets. One of the causes of traffic accidents it is using the phone while participating in traffic. Drivers tend to turn on the phone when an incoming call or notification of message, email, etc. So, it will make them lose focus while driving, as it would endanger for this person and others are participating traffic and accidents can happen any time. Another cause leads to traffic accident this is driving with high speed. In this situation, the driver cannot handle when unexpected situations occur, it will lead to traffic accidents. So, we can see that the dangers of using phone while participating in traffic.

In order to solve this problem, my thesis “Profile Management for people in motorcycle traffic” will develop an application “Motor Safe” which installed on smartphone android can help people more focus on while driving, notification to drivers when they drive with high speed and automatic sending a message to their relatives when accident occur. By using this application, the number of traffic accidents will decrease significantly.

**Table of Contents**

[**ABSTRACT** iv](#_Toc450584026)

[**Table of Contents** v](#_Toc450584027)

[**List of Figures** vii](#_Toc450584028)

[**List of Tables** viii](#_Toc450584029)

[**ABBVERATION** ix](#_Toc450584030)

[Chapter **1** 1](#_Toc450584031)

[**INTRODUCTION** 1](#_Toc450584032)

[**1.1.** **Motivation** 1](#_Toc450584033)

[**1.2. Contribution and overview thesis** 2](#_Toc450584034)

[Chapter 2 4](#_Toc450584035)

[**SYSTEM INTEGRATION** 4](#_Toc450584036)

[**2.1. Hardware** 4](#_Toc450584037)

[**2.1.1. Device** 4](#_Toc450584038)

[**2.1.2. Computer** 5](#_Toc450584039)

[**2.2. Software** 5](#_Toc450584040)

[**2.2.1. Android studio** 5](#_Toc450584041)

[**2.2.2. Language programing** 8](#_Toc450584042)

[**2.2.3. Overview android** 8](#_Toc450584043)

[Chapter 3 13](#_Toc450584044)

[**THE PROPOSE METHOD** 13](#_Toc450584045)

[**3.1. System block diagram** 13](#_Toc450584046)

[**3.2. Detecting user’s status** 14](#_Toc450584047)

[**3.3. Determine the velocity** 17](#_Toc450584048)

[**3.3.1. Get speed from accelerometer** 18](#_Toc450584049)

[**3.3.2. Get speed by using GPS** 19](#_Toc450584050)

[**3.4. Detecting accidents** 22](#_Toc450584051)

[Chapter 4 25](#_Toc450584052)

[**SETUP PROJECT** 25](#_Toc450584053)

[**4.1. Setup environment** 25](#_Toc450584054)

[**4.1.1. Install Java JDK** 25](#_Toc450584055)

[**4.1.2. Install Android Studio** 25](#_Toc450584056)

[**4.1.3. Setup project** 26](#_Toc450584057)

[**4.2. Implementation** 27](#_Toc450584058)

[**4.2.1. Detecting user’s status** 27](#_Toc450584059)

[**4.2.2. Determine the velocity** 30](#_Toc450584060)

[**4.2.3. Detecting accidents** 32](#_Toc450584061)

[Chapter 5 34](#_Toc450584062)

[**RESULTS AND DISCUSSIONS** 34](#_Toc450584063)

[**5.1. Results** 34](#_Toc450584064)

[**5.1.1. Detecting user’s status** 34](#_Toc450584065)

[**5.1.2. Determine the velocity** 37](#_Toc450584066)

[**5.1.3. Detecting accidents** 38](#_Toc450584067)

[**5.1.4. Application** 40](#_Toc450584068)

[**5.2. Discussions** 47](#_Toc450584069)

[Chapter 6 49](#_Toc450584070)

[**CONCLUSIONS** 49](#_Toc450584071)

[**6.1. Conclusions** 49](#_Toc450584072)

[**6.2. Future works** 50](#_Toc450584073)

[**Reference** 51](#_Toc450584074)

# **List of Figures**

[**Figure 1-1 - (a) Using mobile phone while driving; (b) Driving high speed** 1](#_Toc450584079)

[**Figure 2-1 - Samsung Galaxy Trend Plus** 4](#_Toc450584080)

[**Figure 2-2 - The project file in Android Studio view** 7](#_Toc450584081)

[**Figure 2-3 - Android architecture** 9](#_Toc450584082)

[**Figure 3-1 - System block diagram** 13](#_Toc450584085)

[**Figure 3-2 - Diagram for Detecting Activity** 17](#_Toc450584087)

[**Figure 3-3 - Three segments of GPS** 19](#_Toc450584088)

[**Figure 3-4 - Space segment** 20](#_Toc450584089)

[**Figure 3-5 - Get position from satellite** 21](#_Toc450584090)

[**Figure 3-6 - System coordinate** 22](#_Toc450584091)

[**Figure 3-7 - Processing of accident** 23](#_Toc450584092)

[**Figure 3-8 - Sending message after accident** 24](#_Toc450584093)

[**Figure 4-1 - Setup Android Studio** 26](#_Toc450584094)

[**Figure 4-2 - Create a new project** 27](#_Toc450584095)

[**Figure 4-3 - Setup class and resource for project** 27](#_Toc450584096)

[**Figure 5-1 - The percentage of detecting “Walking” in training “Walking”** 35](#_Toc450584097)

[**Figure 5-2 - The percentage of detecting “On vehicle” in training “Walking”** 35](#_Toc450584098)

[**Figure 5-3 - The percentage of detecting “Walking” in training “On vehicle”** 36](#_Toc450584099)

[**Figure 5-4 - The percentage of detecting “On vehicle” in training “On vehicle”** 36](#_Toc450584100)

[**Figure 5-5 - The percentage of detecting “On vehicle” in training “On vehicle” with stable speed** 37](#_Toc450584101)

[**Figure 5-6 - Result training: (a) Speed 33 km/h (± 2 km/h); (b) Speed 37 km/h (± 2 km/h)** 37](#_Toc450584102)

[**Figure 5-7 - Accelerometer data for training walking** 38](#_Toc450584103)

[**Figure 5-8 - Accelerometer data for training on vehicle (No accident)** 39](#_Toc450584104)

[**Figure 5-9 - Accelerometer data from demo accident** 39](#_Toc450584105)

[**Figure 5-10 - Total accelerometer (Acc) and mean (win-acc) by windowSize = 5** 40](#_Toc450584106)

[**Figure 5-11 - User Interface** 41](#_Toc450584107)

[**Figure 5-12 - Menu setting** 42](#_Toc450584108)

[**Figure 5-13 - Setting initial values** 43](#_Toc450584109)

[**Figure 5-14 - (a) Setting the VIP contacts; (b) Setting the maximum speed; (c) Setting message** 44](#_Toc450584110)

[**Figure 5-15 - (a) Tutorial; (b) About; (c) Confirm to exit application** 45](#_Toc450584111)

[**Figure 5-16 - Change to silent mode while on vehicle** 46](#_Toc450584112)

[**Figure 5-17 - Warning high speed** 46](#_Toc450584113)

[**Figure 5-18 - Message notification and address of accident** 47](#_Toc450584114)

[**Figure 5-19 – Notification when accident or free falling** 48](#_Toc450584115)

# **List of Tables**

[**Table 2.1** - **Components in Android** 10](#_Toc450584141)

[**Table 2.2** - **Additional Components** 11](#_Toc450584142)

[**Table 3.1 - The constant in DetectedActivity** 15](#_Toc450584144)

# **ABBVERATION**

|  |  |
| --- | --- |
| API | Application Programing Interface |
| GPS | Global Positioning System |
| OS | Operating System |
| APK | Android Application Package |
| SDK | Software Development Kit |
| JDK | Java Development Kit |

# Chapter **1**

# **INTRODUCTION**

## **Motivation**

In 2015, According to the Traffic Police Department, Viet Nam has 22827 cases traffic accidents, in which died 8727 person and 21069 persons were injured. One of the causes of traffic accidents it is using the phone while participating in traffic and driver with too high speed.



(a) (b)

**Figure 1-1 - (a) Using mobile phone while driving; (b) Driving high speed**

Mobile phone becomes a familiar object for many people. They use the phone to listen to, call, photography, messaging, online, facebook,... anytime, anywhere, even when you're participating in traffic. Both controller motor and using mobile phone to call, texting is easy image caught on the roads every day. However, when using the phone in conjunction with the drivers also bring the potential risks. Losing a hand to hold the phone will make you lose focus while driving, and they cannot react when they meet unexpected situations. According to the analysis of the relevant authorities, the number of traffic accidents caused by phone use while participating in traffic equivalent to the number of accidents caused by drivers using alcohol. Using your phone while participating in traffic will cause distractions, limited visibility, making the driver of vehicles distraction; the ability to manipulate and control the speed to meet unexpected situations will be embarrassing, not timely handling accidents are inevitable.

Many people tend to drive with high speed when they are in a hurry or drink alcohol. So, they do not handle when unexpected situations occur, it will lead to traffic accident.

So, in order to answer the question: How to reduce traffic accidents? Road Traffic Law 2008 provides: "Banning person is driving motorcycles, motorcycle behaving mobile phone use". Recently, Decree No. 71/2012 / ND-CP of the Government on administrative sanctions in the road transport sector also stipulates: "Sanctions at 60,000 - 80,000 Vietnamese Dong for violations of drivers using cell , mobile phones, audio equipment, except for hearing aids. "Besides, there are law regulation on the speed limit while driving. And, in this thesis, I will present another way to reduce traffic accidents, this is an application which installed on smartphone of driver. It prevents any incoming call, warning speed of motor and notification when accident occur. So, this application will help driver to focus while driving and reduce traffic accidents.

## **1.2. Contribution and overview thesis**

In this thesis, my application which is call “Motor Safe” has built for android smartphone using Android Studio. My application will determine the status of user (walking, driving) change mode of mobile for suitable cases, beside that notification when user drivers with high speed and accidents occur. There are three main functions in this application. Firstly, in order to determine the status of user, I use the Google Play Service Activity Recognition API. Secondly, I use GPS to calculate speed of vehicle, then notification to user when their speed is bigger than maximum speed. Thirdly, I user both GPS and accelerometer sensor on android smartphone to detect accidents.

This thesis is divided into six following chapters:

**Chapter 1:** Introduction

The motivation is showed to explain why I choose this topic and develop application. Some brief overview of the thesis is also described in this chapter.

**Chapter 2:** System integration

In this chapter, I show the hardware and software which is used to develop application.

**Chapter 3:** The propose method

I show that the algorithm for each functions of application.

**Chapter 4:** Setup project

I show that the steps to develop project and how to implement algorithm for each functions.

**Chapter 5:** Results and discussions

I show the results after this application is completed and discussion.

**Chapter 6:** Conclusion

# Chapter 2

# **SYSTEM INTEGRATION**

## **2.1. Hardware**

### **2.1.1. Device**

In this thesis, I use smartphone Samsung Galaxy Trend Plus GT S7580.



**Figure 2-1 - Samsung Galaxy Trend Plus**

Detail specification:

* CPU: Dual core 1.2Ghz
* RAM: 768 MB
* Android OS: Android 4.2.2 (Jelly Bean)
* ChipSet: Broadcom BCM21664
* MicroUSB 2.0
* Sensor: Accelerometer (BOSCH BMC150), Proximity (GP2A), Magnetic (BOSCH BMC150).

### **2.1.2. Computer**

The system develop and builds application:

* Windows: Microsoft® Windows® 10, 64 bit
* CPU: Intel, Core i5 Haswell, 4200U, 1.60 GHz
* RAM: 8GB
* GPU: Nvidia Geforce 720M.

## **2.2. Software**

### **2.2.1. Android studio**

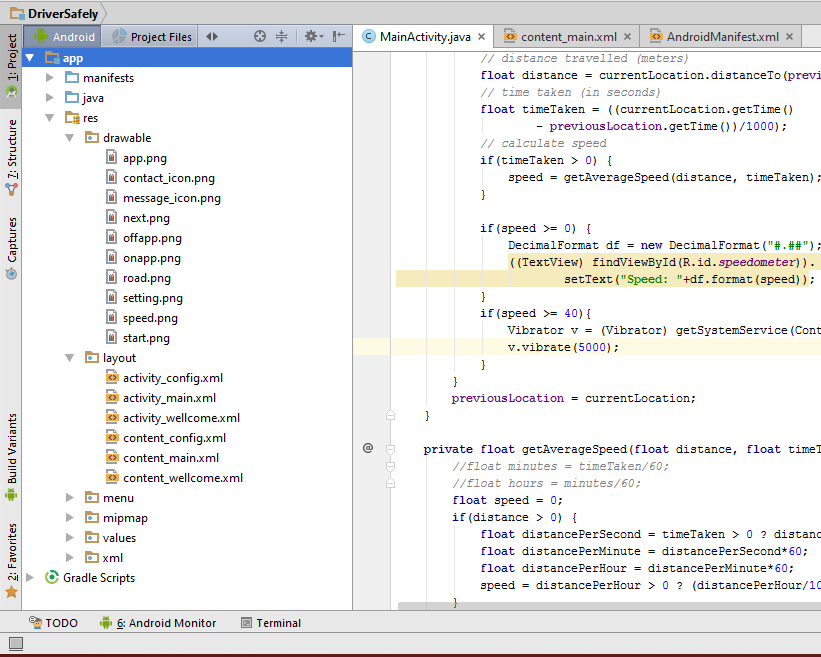
Android is a mobile operating system (OS) based on the Linux kernel and currently developed by Google. With a user interface based on direct manipulation, Android is designed primarily for touchscreen mobile devices such as smartphones and tablet computers, with specialized user interfaces for televisions (Android TV), cars (Android Auto), and wrist watches (Android Wear). The OS uses touch inputs that loosely correspond to real-world actions, like swiping, tapping, pinching, and reverse pinching to manipulate on-screen objects, and a virtual keyboard. Despite being primarily designed for touchscreen input, it also has been used in game consoles, digital cameras, regular PCs and other electronics.

Android's source code is released by Google under open source licenses, although most Android devices ultimately ship with a combination of open source and proprietary software, including proprietary software developed and licensed by Google. Initially developed by Android, which Google backed financially and later bought in 2005, Android was unveiled in 2007 along with the founding of the Open Handset Alliance-a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices.

Because of its open source nature, Android really attracted a large community of developers. Therefore, Android always led in the number of mobile application and remains the most mobile platform popular till now.

Android Studio is the official IDE for Android app development, based on IntelliJ IDEA. On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

* A flexible Gradle-based build system
* Build variants and multiple APK file generation
* Code templates to help you build common app features
* A rich layout editor with support for drag and drop theme editing
* Lint tools to catch performance, usability, version compatibility, and other problems
* Code shrinking with ProGuard and resource shrinking with Gradle
* Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine



**Figure 2-2 - The project file in Android Studio view**

Each project in Android Studio contains one or more modules with source code files and resource files. Different types of modules include:

* Android app modules
* Test modules
* Library modules
* App Engine modules

By default, Android Studio displays your project files in the Android project view, as shown in figure 2.2. This view is organized by modules to provide quick access to the key source files of your project.

All the build files are visible at the top level under Gradle Scripts and each app module contains the following three elements:

* manifests: Manifest files.
* java: Source code files.
* res: Resource files.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select Project from the Project drop-down (in figure 1, it's showing as Android).

Android Studio uses Gradle as the foundation of the build system, with more Android-specific capabilities provided by the Android Plugin for Gradle. This build system runs as an integrated tool from the Android Studio menu and independently from the command line. You can use the features of the build system to:

* Customize, configure, and extend the build process.
* Create multiple APKs for your app with different features using the same project and modules.
* Reuse code and resources across source sets.

The flexibility of Gradle enables to achieve all of this without modifying app's core source files.

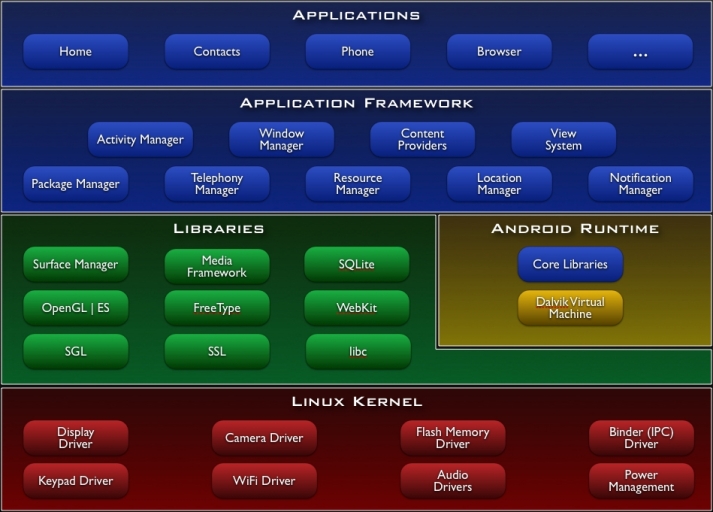
### **2.2.2. Language programing**

Applications are usually developed in Java programming language using the Android software development kit (SDK).

### **2.2.3. Overview android**

* **Android architecture[7]**

Android operating system is a stack of software components which roughly divided into five sections and four main layers as shown below in the architecture diagram.



**Figure 2-3 - Android architecture [9]**

* **Linux kernel**: At the bottom of the layers is Linux - Linux 2.6 with approximately 115 patches. This provides basic system functionality like process management, memory management, device management like camera, keypad, display etc. Also, the kernel handles all the things that Linux is really good at such as networking and a vast array of device drivers, which take the pain out of interfacing to peripheral hardware.
* **Libraries:** On top of Linux kernel there is a set of libraries including open-source Web browser engine WebKit, well known library libc, SQLite database which is a useful repository for storage and sharing of application data, libraries to play and record audio and video, SSL libraries responsible for Internet security etc.
* **Android Runtime:** This is the third section of the architecture and available on the second layer from the bottom. This section provides a key component called Dalvik Virtual Machine which is a kind of Java Virtual Machine specially designed and optimized for Android. The Dalvik VM makes use of Linux core features like memory management and multi-threading, which is intrinsic in the Java language. The Dalvik VM enables every Android application to run in its own process, with its own instance of the Dalvik virtual machine. The Android runtime also provides a set of core libraries which enable Android application developers to write Android applications using standard Java programming language.
* **Application Framework:** The Application Framework layer provides many higher-level services to applications in the form of Java classes. Application developers are allowed to make use of these services in their applications.
* **Applications:** all the Android application at the top layer. Writing application to be installed on this layer only. Examples of such applications are Contacts Books, Browser, Games, etc.
* **Android components [7]**

Application components are the essential building blocks of an Android application. These components are loosely coupled by the application manifest file AndroidManifest.xml that describes each component of the application and how they interact. There are following four main components that can be used within an Android application:

**Table 2.1** - **Components in Android**

|  |  |
| --- | --- |
| **Components** | **Description** |
| Activities | They dictate the UI and handle the user interaction to the smartphone screen |
| Services | They handle background processing associated with an application. |
| Broadcast Receiver | They handle communication between Android OS and applications. |
| Content Provider | They handle data and database management issues. |

* **Activities**: An activity represents a single screen with a user interface. For example, an email application might have one activity that shows a list emails, another activity to compose an email, and another activity reading emails. If an application has more than one activity, then one of them should be marked as the activity that is presented when the application is launched. An activities is implemented as a subclass of Activity class as follow:

public class MainActivity extends Activity{}

* **Services**: A service is a component that runs in the background to perform long-running operations. For example, a service might play music in the background while the user is in a different application, it might fetch data over the network without blocking user interaction with activity. A service is implemented as a subclass of Service class as follows:

public MyService extend Service{}

* **Broadcast Receivers**: Broadcast Receivers simply respond to broadcast messages from other applications or from the system. For example applications can also initiate broadcasts to let other applications know that some data has been downloaded to the device and is available for them to use, so this is broadcast receiver who will intercept this communication and will initiate appropriate action. A broadcast receiver is implemented as a subclass of BroadcastReceiver class and each message is broadcasted as an Intent object.

public class MyReceiver extend BroadcastReceiver{}

* **Content Providers**: A content provider component supplies data from one application to others on request. Such requests are handled by the methods of the ContentResolver class. The data may be stored in the file system, the database or somewhere else entirely. A content provider is implemented as a subclass of ContentProvider class and must implement a standard set of APIs that enable other applications to perform transactions.

public class MyContentProvider extends ContentProvider{}

* **Additional Components**: There are additional components which will be used in the construction of above mentioned entities, their logic, and wiring between them. These components are:

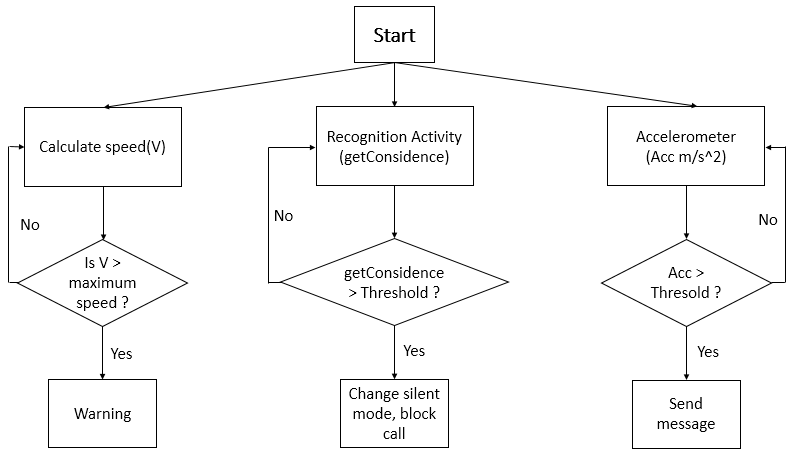
**Table 2.2** - **Additional Components**

|  |  |
| --- | --- |
| **Components** | **Description** |
| Fragments | Represents a behavior or a portion of user interface in an Activity |
| Views | UI elements that are drawn onscreen including buttons, lists forms, etc… |
| Layouts | View hierarchies that control screen format and appearance of the views |
| Intents | Messages wiring components together |
| Resources | External elements, such as strings, constants and drawables pictures |
| Manifest | Configuration file for the application |

# Chapter 3

# **THE PROPOSE METHOD**

## **3.1. System block diagram**



**Figure 3-1 - System block diagram**

When user starts application, there are three threads which run parallel:

* In the first thread, application will calculate the speed of user and motorcycle, if this speed is bigger than the maximum speed which is sat by user (40 Km/h, 50 Km/h, etc…), the application will send notification to user (vibration).
* In the second thread, application detects the activity of user. In my project, I focus on to detect on motorcycle or not. If the value obtain from device is over than the threshold, the application will control device to change the mode to silent mode. In this mode, nobody can call user except from contacts in list VIP contacts.
* In the last thread, application will detect accident by using accelerometer sensor on mobile phone. If values obtain from accelerometer is bigger than threshold, application will auto send message to VIP contacts.

## **3.2. Detecting user’s status**

In order to detect the status of user, I use Google Play Services Activity Recognition API. Activity Recognition gives Android device the ability to detect a number of our physical activities like walking, riding a bicycle, driving a car, motorcycle or standing idle. All that can be detected by simply using an API to access Google Play Service, an increasingly crucial piece of software available to all Android versions.

There two public methods in ActivityRecognitionApi:

* **Public abstract PendingResult<Status> removeActivityUpdates(GoogleApiClient client, PendingIntent callBackIntent)**: Remove all activity updates for the specified PendingIntent.

Parameters: client: An existing GoogleApiClient; callbackIntent: the PendingIntent that was used in request activity update.

* **Public abstract PendingResult<Status> requestActivityUpdates(Google client, long detectionIntervalMilis, PendingIntent callbackIntent)**: Register for activity recognition updates.

The activities are detected by periodically waking up the device and reading short bursts of sensor data. It only makes use of low power sensors in order to keep the power usage to a minimum. For example, it can detect if the user is currently on foot, in a car, on a bicycle or still.

The activity detection update interval can be controlled with the detectionIntervalMillis parameter. Larger values will result in fewer activity detections while improving battery life. Smaller values will result in more frequent activity detections but will consume more power since the device must be woken up more frequently. Long.MAX\_VALUE means it only monitors the results requested by other clients without consuming additional power.

Activities may be received more frequently than the detectionIntervalMillis parameter if another application has also requested activity updates at a faster rate. It may also receive updates faster when the activity detection service receives a signal that the current activity may change, such as if the device has been still for a long period of time and is then unplugged from a phone charger.

Activities may arrive several seconds after the requested detectionIntervalMillis if the activity detection service requires more samples to make a more accurate prediction.

To conserve battery, activity reporting may stop when the device is 'STILL' for an extended period of time. It will resume once the device moves again. This only happens on devices that support the Sensor.TYPE\_SIGNIFICANT\_MOTION hardware.

Beginning in API 21, activities may be received less frequently than the detectionIntervalMillis parameter if the device is in power save mode and the screen is off.

The detected activity of the device with an associated confidence. DetectedActivity was obtained from ActivityRecognitionApi. Some constant will show in table below:

**Table 3.1 - The constant in DetectedActivity**

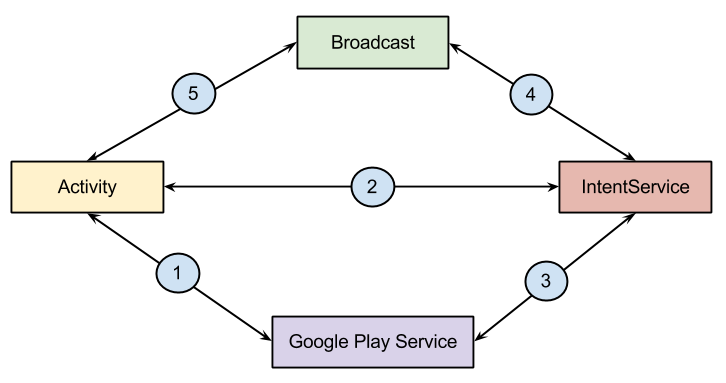
|  |  |  |
| --- | --- | --- |
| **Constants** | **Describe** | **Constant value** |
| IN\_VEHICLE | The device is in a vehicle, such as a car. | 0 |
| ON\_BICYCLE | The device is on a bicycle. | 1 |
| ON\_FOOT | The device is on a user who is walking or running. | 2 |
| RUNNING | The device is on a user who is running. This is a sub-activity of ON\_FOOT. | 3 |
| STILL | The device is still (not moving). | 4 |
| TITLING | The device angle relative to gravity changed significantly. This often occurs when a device is picked up from a desk or a user who is sitting stands up. | 5 |
| UNKNOWN | Unable to detect the current activity. | 6 |
| WALKING | The device is on a user who is walking. This is a sub-activity of ON\_FOOT. | 7 |

The public methods:

* **public int describeContents ()**
* **public int getConfidence ()**: Returns a value from 0 to 100 indicating the likelihood that the user is performing this activity. The larger the value, the more consistent the data used to perform the classification is with the detected activity. This value will be <= 100. It means that larger values indicate that it's likely that the detected activity is correct, while a value of <= 50 indicates that there may be another activity that is just as or more likely.
* **public int getType()**: return the status of driver.
* **public String toString()**
* **public void writeToParcel(Parcel out, int flags)**.

For this project, I just care to detect the status of user when is driving motorcycle, so in order to determine threshold, I base on the method getConfidence() with getType() return “on vehicle”.

The process to detect activity of user will implement as diagram following:



**Figure 3-2 - Diagram for Detecting Activity**

The application need to done following operations to return the expect result:

* Check availability of Google Play Service using "isGooglePlayServiceAvailable" function.
* If Google Play Service is available then do following steps:
* Implements ConnectCallback and OnConnectionFailedListener
* Create Object of ActivityRecognitionClient
* Call connect() method using object of ActivityRecognitionClient
* onConnect method will call after connection is established between app and Google Play service. In this method we should start request for Activity Recognition.
* Create and register BroadcastReceiver.
* In destroy method: remove activity updates, disconnect with Google Play Service and remove BroadcastReceiver.

## **3.3. Determine the velocity**

In order to obtain the speed of the device, there are some ways to do it. In this thesis, I will present two methods to get speed from smartphone.

### **3.3.1. Get speed from accelerometer**

To get speed of the device, I need integrate acceleration:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

where: a (m/s2) is the instantaneous acceleration at time t (s).

Assumption that the value accelerator is read with a space δt (s), I have:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |

where v(t) is velocity at time t.

Because, I get accelerometer by three dimensional, so I need integrate in each of the three dimensions x, y, z separated.

|  |  |  |
| --- | --- | --- |
|  | *vx = ax\*dt*  *vy = ay\*dt*  *vz = az\*dt* | (3) |

where: vx, vy, vz are velocity in three dimensional and ax, ay, az are the instantaneous accelerometer which is measured by sensor on mobile phone.

Total speed |v|, give by (4):

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

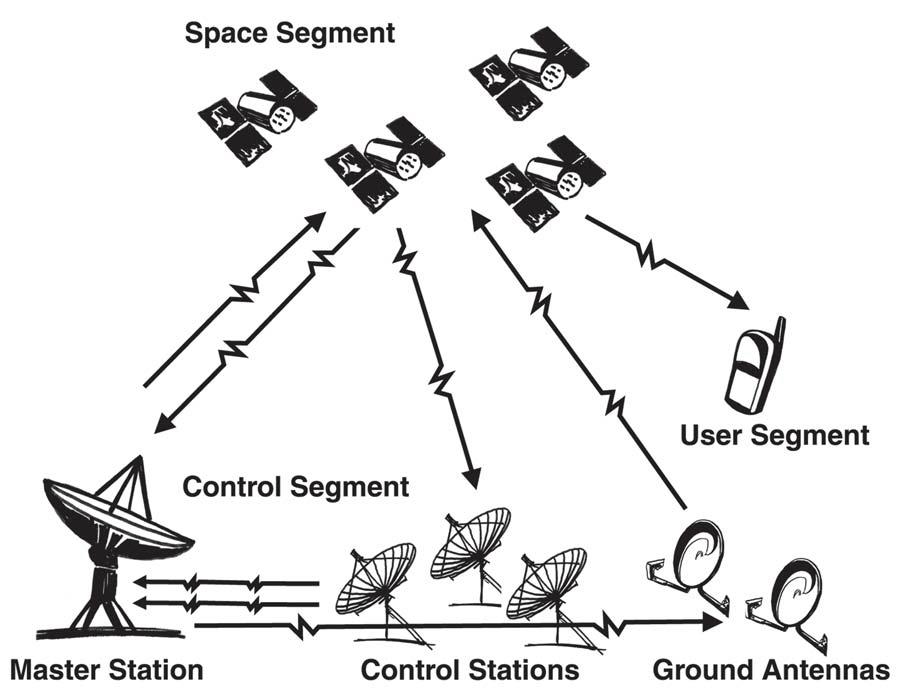
In practice, I face many issues trying to implement it, namely:

* Integrating acceleration and integration error will grow indefinitely. The errors get out of control really quick.
* Determine the initial speed based on the same referential as the acceleration. This would require some measurements and a lot of trigonometry, as I would get both values from different sensors at different rates.
* The motorbike will not move in a straight line, so acceleration referential will be constantly moving (a lot more of trigonometry and calculus).
* If the device is in the user hand, the device movements in relation to the car will increase even more the calculations (and accumulated errors).

So, in this thesis, I use other method to get speed from smartphone. This is GPS.

### **3.3.2. Get speed by using GPS**

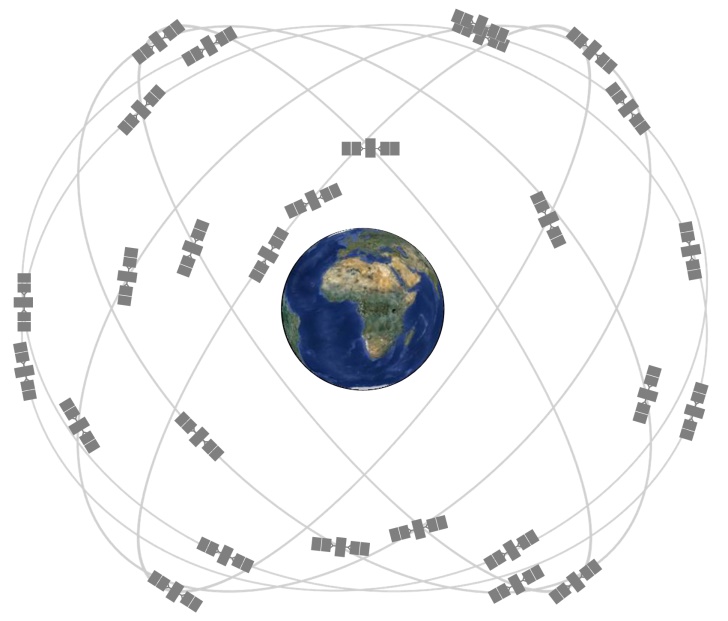
GPS stands for Global Positioning System, and refers to a group of U.S. Department of Defense satellites constantly circling the earth. The satellites transmit very low power radio signals allowing anyone with a GPS receiver to determine their location on Earth [6].



**Figure 3-3 - Three segments of GPS**

Three segments of GPS: space segment, control segment, user segment.

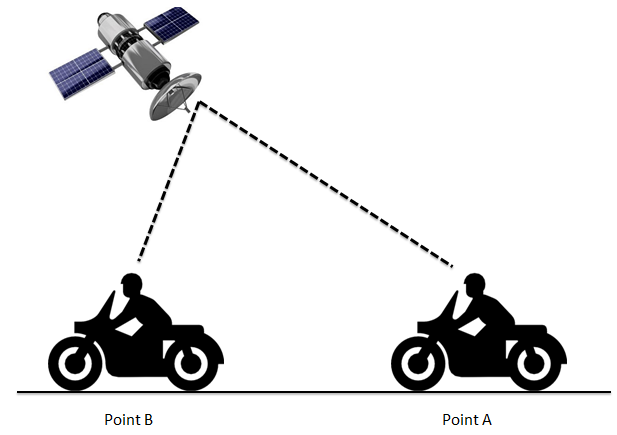
* Control segment: The U.S. Department of Defense maintains a master control station at Falcon Air Force Base in Colorado Springs, CO. There are four other monitor stations located in Hawaii, Ascension Island, Diego Garcia and Kwajalein. The DoD stations measure the satellite orbits precisely. Any discrepancies between predicted orbits and actual orbits are transmitted back to the satellites. The satellites can then broadcast these corrections, along with the other position and timing data, so that a GPS receiver on the earth can precisely establish the location of each satellite it is tracking.
* Space segment:



**Figure 3-4 - Space segment**

* The number of satellite vehicles: minimum is 24 and maximum is 32.
* Six orbital planes: inclined 55o with respect to equator.
* 22200 km elevation above earth
* Orbital period of 11h55min
* 5 to 8 visible from any point on Earth
* Broadcast three signals on two frequencies: coarse/acquisition signal for civilian use and precision signal for military use.
* User segment

Nowadays, any smartphone is also integrated GPS. So, smartphone with GPS receivers communicate with units from among the 30 global positioning satellites in the GPS system. The built-in receiver trilaterates position using data from at least three GPS satellites and the receiver. GPS can determine the location by performing a calculation based on the intersection point of overlapping spheres determined by the satellites and your phone's GPS receiver.



**Figure 3-5 - Get position from satellite**

Speed of vehicle equals distance covered divided by the time taken:

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

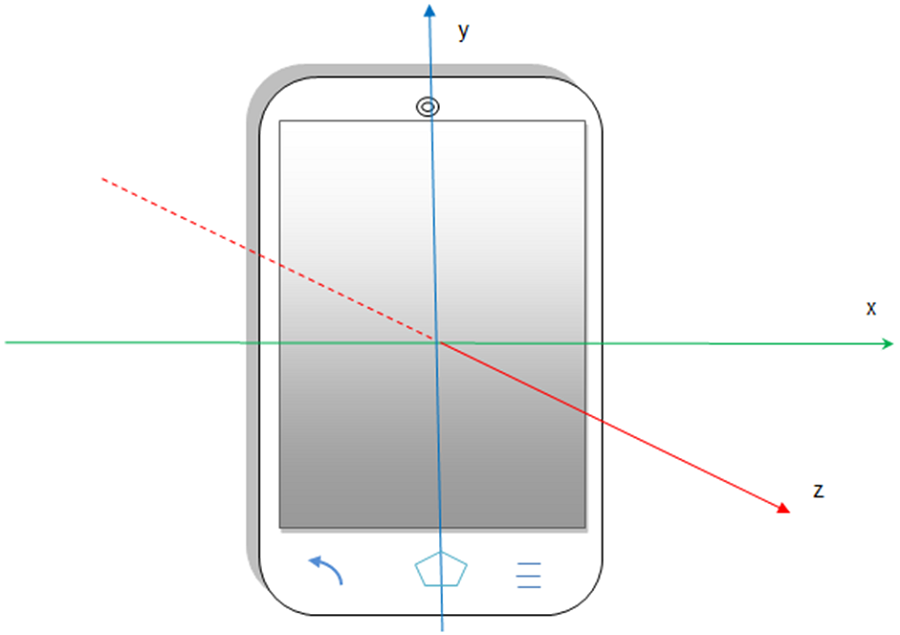
where d (m) is the distance between point A and point B; t (s) is the interval time to user moving from point A to point B.

GPS satellites send their positions to receivers on the ground every second. In a split second the GPS receiver will generally perform the following tasks to determine speed:

* Convert the difference between the two latitudinal/longitudinal positions into a unit of measurement
* Determine the difference between the two timestamps to calculate how long it took to get from Point A to Point B.
* Calculate the average speed based on these results.

GPS devices are positional speedometers, based on how far the device has moved since the last measurement. The algorithm also uses the Doppler shift in the pseudo range signals from the satellites. It should also be noted that the speed reading is normalized, and is not an instant speed. Speeds are updated at short intervals to maintain accuracy at all times. It uses frequent calculations to determine the vehicle’s speed.

## **3.4. Detecting accidents**



**Figure 3-6 - System coordinate**

The accelerometer sensor uses a standard 3-axis coordinate system to express value. The default orientation of device will show in figure 3.9. When a device is held in its default orientation, the X axis is horizontal and point to right, the Y axis is vertical and points up, and the Z axis points toward the outside of the screen face.

Android system allows to specify sampling frequency using one of four sensor delay:

* SENSOR\_DELAY\_FASTEST: 0 microseconds delay
* SENSOR\_DELAY\_GAME: 20000 microseconds delay
* SENSOR\_DELAY\_NORMAL: 200000 microseconds delay
* SENSOR\_DELAY\_UI: 60000 microseconds delay

An acceleration sensor measures the acceleration applied to the device, including the force of gravity.

Conceptually, an acceleration sensor determines the acceleration that is applied to a device (Ad) by measuring the forces that are applied to the sensor itself (Fs) using the following relationship:

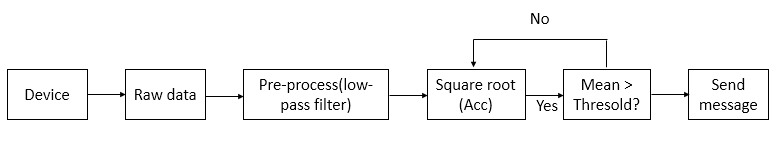
|  |  |  |
| --- | --- | --- |
|  |  | (6) |

However, the force of gravity is always influencing the measured acceleration according to the following relationship:

|  |  |  |
| --- | --- | --- |
|  |  | (7) |

For this reason, when the device is sitting on a table (and not accelerating), the accelerometer reads a magnitude of g = 9.81 m/s2. Similarly, when the device is in free fall and therefore rapidly accelerating toward the ground at 9.81 m/s2, its accelerometer reads a magnitude of g = 0 m/s2. In this project, I use a low-pass filter can be used to isolate the force of gravity.

The following figure will express the processing of acceleration to determine accident:



**Figure 3-7 - Processing of accident**

After pre-process by using low-pass filter, the square root (Acc) is calculated:

|  |  |  |
| --- | --- | --- |
|  |  | (8) |

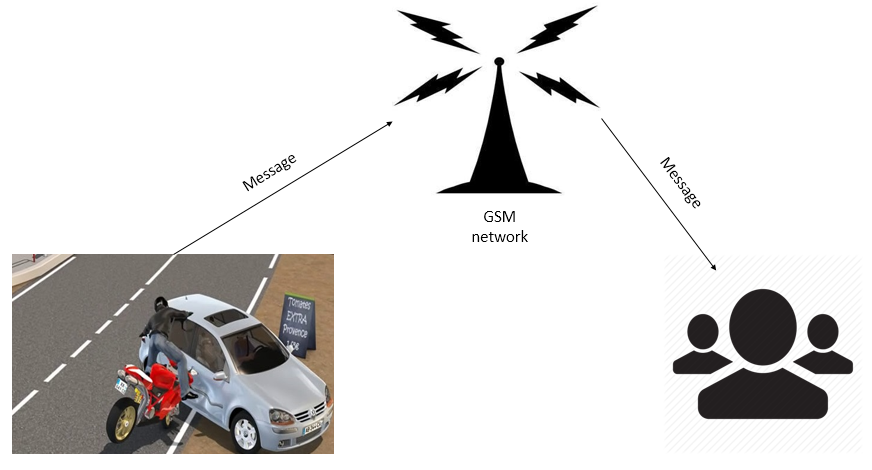
where: ax, ay, az is acceleration value by each axis.

Mean is obtained by using a moving-average filter, this is represented by the following difference equation:

|  |  |  |
| --- | --- | --- |
|  |  | (9) |

where: y(n) is “mean”; windowSize in this application, I choose windowSize = 5.

By obtaining “mean”, I compare it with threshold 4\*g (g = 9.81 m/s2) [8] to detect accident. For the case, accident occurs, an automatic message will send to parent’s contact. This process is simulated by following figure:



**Figure 3-8 - Sending message after accident**

in which, message includes notification and the address of accident(link google map).

# Chapter 4

# **SETUP PROJECT**

## **4.1. Setup environment**

### **4.1.1. Install Java JDK**

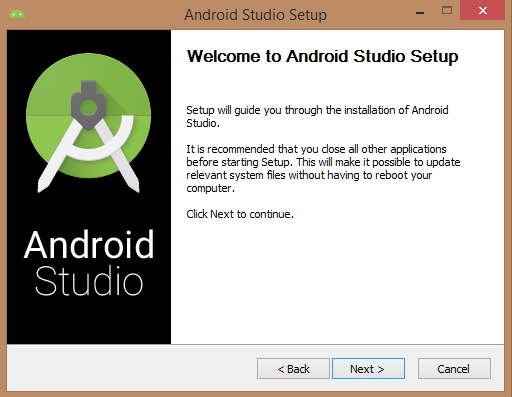
A Java Development Kit (JDK) is a program development environment for writing Java applets and applications. In this thesis, I must install JDK to develop my application in Android Studio.

To install JDK:

1. Install the IDK software
2. Set JAVA\_HOME: add JAVA\_HOME in Environment Variables (C:\Program Files\Java\jdk1.8.0\_25).

### **4.1.2. Install Android Studio**

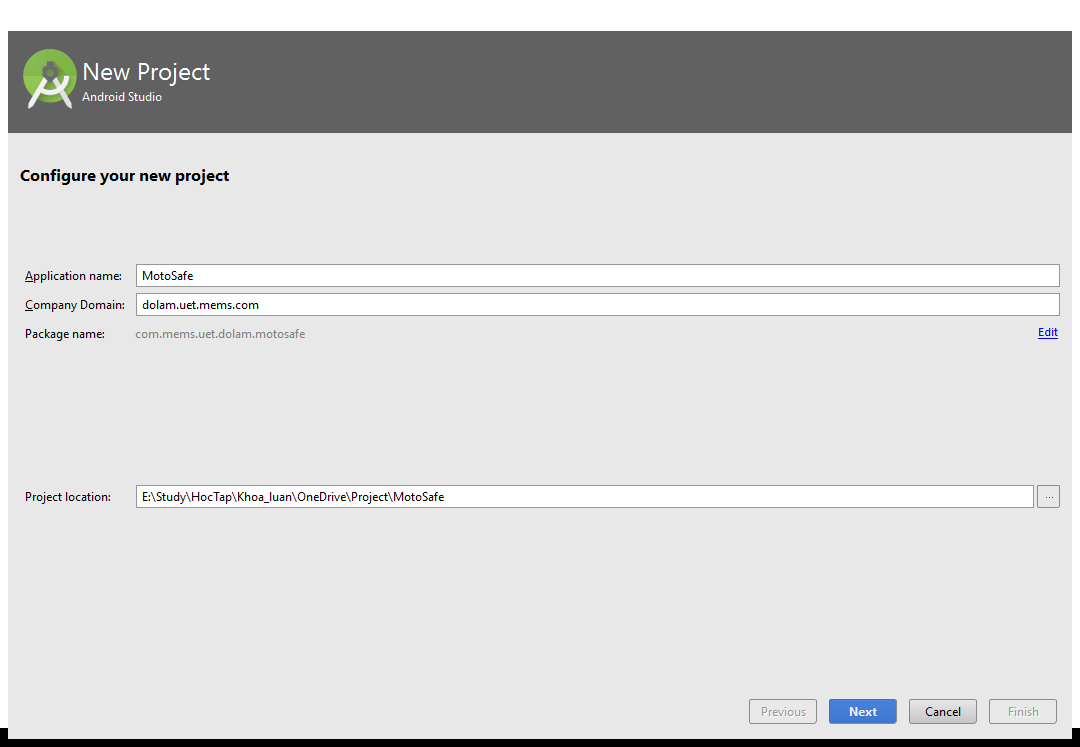
I use IDE Android Studio to develop my application.



**Figure 4-1 - Setup Android Studio**

### **4.1.3. Setup project**

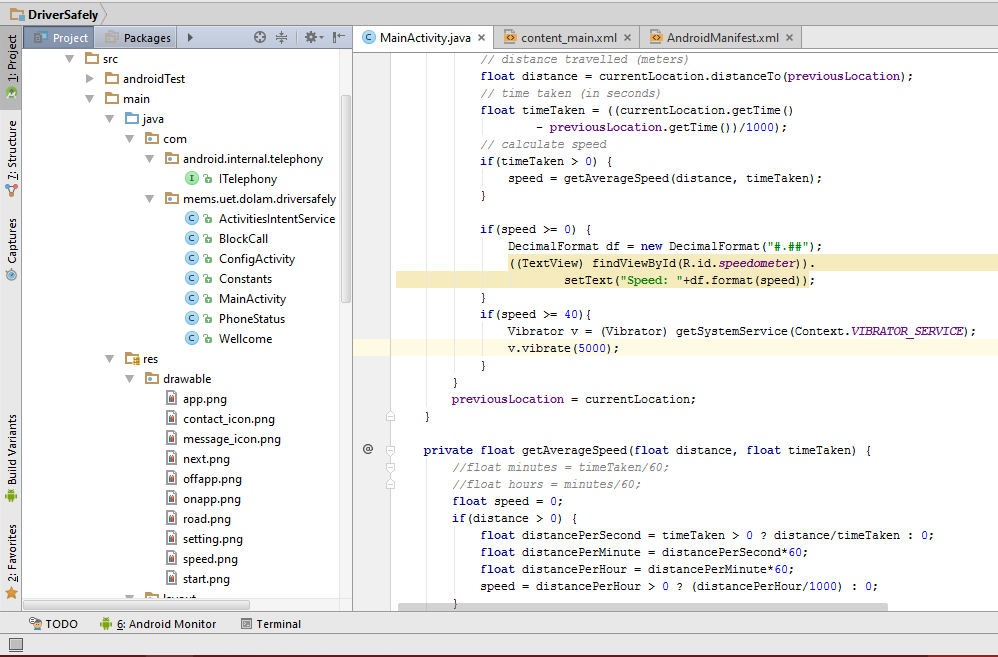
Start a new android project.



**Figure 4-2 - Create a new project**

Create a new project with name “MotorSafe” and finish.

Create classes and package to develop application.



**Figure 4-3 - Setup class and resource for project**

Then, set up interface for project and implement algorithm.

## **4.2. Implementation**

### **4.2.1. Detecting user’s status**

In the first, in order to use Activity Recognition, I need a specific permission:

<**uses-permission android:name="com.google.android.gms.permission.ACTIVITY\_RECOGNITION"** />

Import the Play Service under the dependencies:

compile **'com.google.android.gms:play-services:8.1.0'**

Create a new class with name ActivitiesIntentService, and have it extends IntentService. When Google Play Services returns the user's activity, it will be sent to this IntentService.

**public class** ActivitiesIntentService **extends** IntentService {  
 **public** ActivitiesIntentService() {  
 **super**(***“***ActivitiesIntentService***”***);  
 }  
  
 @Override  
 **protected void** onHandleIntent(Intent intent) {  
 }  
}

In the main activity, I implements ConnectionCallbacks and OnConnectionFailedListener interfaces.

public class MainActivity extends AppCompatActivity implements GoogleApiClient.ConnectionCallbacks, GoogleApiClient.OnConnectionFailedListener {

    public GoogleApiClient mApiClient;

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity\_main);

    }

    @Override

    public void onConnected(Bundle bundle) {

    }

    @Override

    public void onConnectionSuspended(int i) {

    }

    @Override

    public void onConnectionFailed(ConnectionResult connectionResult) {

    }

}

Initialize the client and connect to Google Play Services in onCreate() by requesting the ActivityRecognition.API and associating your listeners with the GoogleApiClient instance.

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity\_main);

    mApiClient = new GoogleApiClient.Builder(this)

            .addApi(ActivityRecognition.API)

            .addConnectionCallbacks(this)

            .addOnConnectionFailedListener(this)

            .build();

    mApiClient.connect();

}

When GoogleApiClient instance has connected, onCreated() is called, I create a PendingIntent that goes to the IntentService.

@Override

public void onConnected(Bundle bundle) {

    Intent intent = new Intent( this, ActivitiesIntentService.class );

    PendingIntent pendingIntent = PendingIntent.getService( this, 0, intent, PendingIntent.FLAG\_UPDATE\_CURRENT );

    ActivityRecognition.ActivityRecognitionApi.requestActivityUpdates( mApiClient, 0, pendingIntent );

}

When receiving Intent contains activity recognizing data, the method onHandleIntent() in ActivitiesIntentService extract the ActivityRecognitionResult from the Intent to see what activities user might be performing.

@Override

protected void onHandleIntent(Intent intent) {

    if(ActivityRecognitionResult.hasResult(intent)) {

        ActivityRecognitionResult result = ActivityRecognitionResult.extractResult(intent);

        handleDetectedActivities( result.getProbableActivities() );

    }

}

For case in which device detecting user is driving, user’s smartphone will change silent mode and nobody can call to this user except the contacts in “VIP contacts”. Any incoming call will be rejected and has message send to caller. So, in order to implement this function, I create method which is called “blockCall()”.

### **4.2.2. Determine the velocity**

Firstly, I give permission for Get Location data in Manifest file:

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION"/>

      <uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION"/>

In the main activity, I implement LocationListener. In order to speed of device, in the onLocationChanged, I calculate distance and interval time between two locations. After that, get speed by divide distance to interval time.

**public class** MainActivity **extends** AppCompatActivity **implements**  
 LocationListener {)

Let's add the required methods:

@Override  
**public void** onLocationChanged(Location currentLocation) {}

@Override  
**public void** onStatusChanged(String provider, **int** status, Bundle extras) {}  
  
@Override  
**public void** onProviderEnabled(String provider) {}  
  
@Override  
**public void** onProviderDisabled(String provider) {}

Declaring private variable in the class:

LocationManager **lm**;

Initializing the variables which is just declared and request update:

**lm** = (LocationManager) getSystemService(Context.***LOCATION\_SERVICE***);

**lm**.requestLocationUpdates(LocationManager.***GPS\_PROVIDER***, 0, 0, **this**);

By detecting velocity of user, if user’s moving with speeding is over the maximum speed which is sat by user, when device will notification to user by activing the phone vibration.

### **4.2.3. Detecting accidents**

Firstly, in order to using SensorEventListener, the activity class needs to implement SensorEventListener interface:

**public class** CarAccident **extends** Service **implements** SensorEventListener {}

Let's add the two required methods:

@Override  
**public void** onSensorChanged(SensorEvent event) {}

@Override  
**public void** onAccuracyChanged(Sensor sensor, **int** accuracy) {}

Declaring two private variable in the class:

**private** SensorManager **sensorManager**;  
**private** Sensor **accelerometer**;

Initializing the variables which is just declared and registered a listener:

sensorManager = (SensorManager) getSystemService(Context.***SENSOR\_SERVICE***);  
**accelerometer** = sensorManager.getDefaultSensor(Sensor.***TYPE\_ACCELEROMETER***);

**sensorManager**.registerListener(**this**, **accelerometer**, SensorManager.***SENSOR\_DELAY\_FASTEST***);

Because, this class I extend Service, so I unregisterListener in method onDestroy():

@Override  
**public void** onDestroy(){  
 **super**.onDestroy();**sensorManager**.unregisterListener(**this**);  
}

The value acceleration will be obtained in onSensorChanged method:

@Override  
**public void** onSensorChanged(SensorEvent event) {

**float** rawX = event.**values**[0];  
 **float** rawY = event.**values**[1];  
 **float** rawZ = event.**values**[2];

**}**

# Chapter 5

# **RESULTS AND DISCUSSIONS**

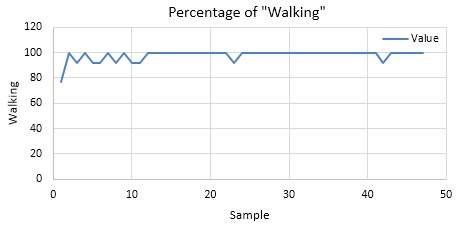
## **5.1. Results**

### **5.1.1. Detecting user’s status**

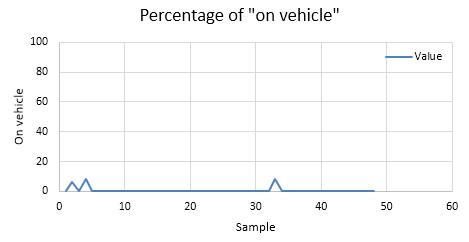
By using Google Play Services Activity Recognition API, I am successful to detect the status of user (on walking, on vehicle). The value obtains from API just only change when the status of user change, such as: on driving and stopped suddenly; avoid obstacles; etc…

The following result is obtained by training:

* Walking:



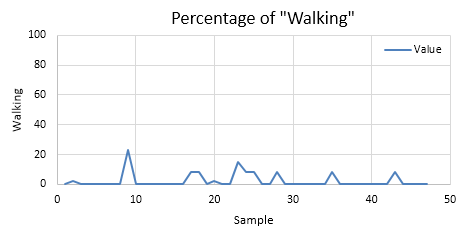
**Figure 5-1 - The percentage of detecting “Walking” in training “Walking”**



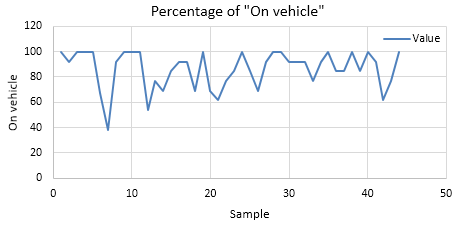
**Figure 5-2 - The percentage of detecting “On vehicle” in training “Walking”**

The figure 5.1 and 5.2 show result of training “Walking”. For this situation, the device is successful in determining the status of users. As can be seen from figure 5.1, the percentage of “Walking” is from 92% to 100 % and in figure 5.2, the percentage of “on vehicle” is from 0% to 8%.

* On vehicle:

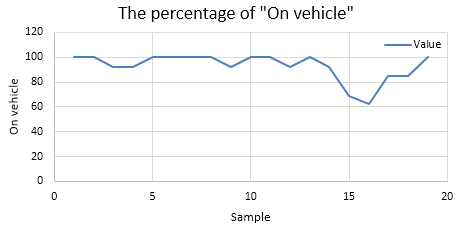


**Figure 5-3 - The percentage of detecting “Walking” in training “On vehicle”**



**Figure 5-4 - The percentage of detecting “On vehicle” in training “On vehicle”**

The figure 5.3 and 5.4 show result of training “On vehicle”. In this training, I drive with speed from 20 km/h to 40 km/h and the distance traveling is 2 km. For this situation, the device is successful in determining the status of user. As can be seen from figure 5.3 and 5.4, the percentage of “On vehicle” is always bigger than the percentage of “Walking”. The value of “Waling” is from 0% to 23% and “On vehicle” is from 38% to 100%, however in “On vehicle” most of value is bigger than 70%. In this training, I also have another result for “On vehicle” and I drive with stable speed 30km/h:



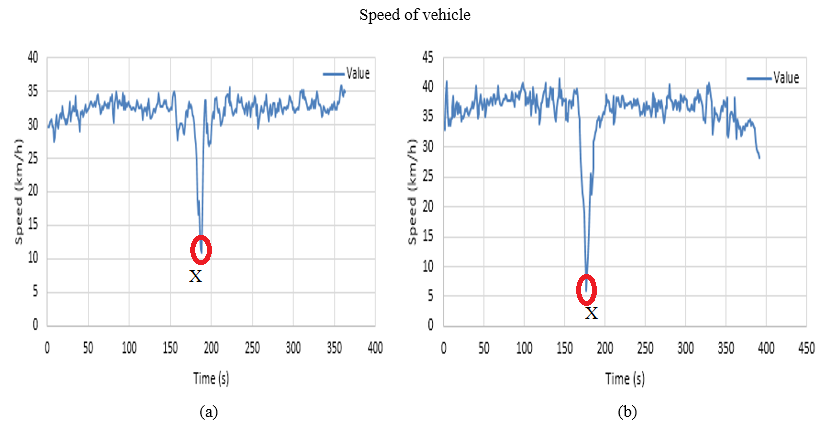
**Figure 5-5 - The percentage of detecting “On vehicle” in training “On vehicle” with stable speed**

Figure 5.5 shows that the percentage of “On vehicle” only small changing.

For this application, I choose threshold which detect on vehicle is 70.

### **5.1.2. Determine the velocity**

By using GPS, I has calculated the velocity of vehicle. The results will show as figure follow:



**Figure 5-6 - Result training: (a) Speed 33 km/h (± 2 km/h); (b) Speed 37 km/h (± 2 km/h)**

In this training, I drive at a speed of approximately 33 km/h (± 2km/h) and 37 km/h (± 2km/h), in 2 km. As can be seen from the figure, the top and bottom of the line is the time which the motorbike begin to speed up and slow down, respectively. In the middle – point “X”, this is time when vehicle reduced speed to turn around. In the part starting to “X” and “X” to end, the value obtained is always approximately 33 km/h for training with 33 km/h and 37 km/h for training with 37 km/h. So, get speed by using GPS given positive results.

### **5.1.3. Detecting accidents**

I am successful by requesting data from accelerometer sensor.



**Figure 5-7 - Accelerometer data for training walking**

Figure 5.7 shows accelerometer values in three axis for training walking. As can be seen from figure 5.7, the values of each axis change in interval short time, after that these values come back value which device does not move, after that changing when user moves.



**Figure 5-8 - Accelerometer data for training on vehicle (No accident)**

Figure 5.3 shows that the data obtained from accelerometer for training on vehicle, in this case, I drive with stable speed. As can be seen from figure 5.3, the value of accelerometer in three axis is small changing between samples.

For demo accident, I has some result:



**Figure 5-9 - Accelerometer data from demo accident**

This above figure is data which obtained from demo accident. As can be seen from figure, when accident occurred, the values accelerometer change suddenly.



**Figure 5-10 - Total accelerometer (Acc) and mean (win-acc) by windowSize = 5**

Figure 5.5 show the total accelerometer which is calculated by (8) and the mean by using moving-average filter which is show by (9).

As can be seen from this figure, when accident occurs, both values total accelerometer and mean are increase suddenly. So, basing on this feature to detect accident and send message to notification.

### **5.1.4. Application**

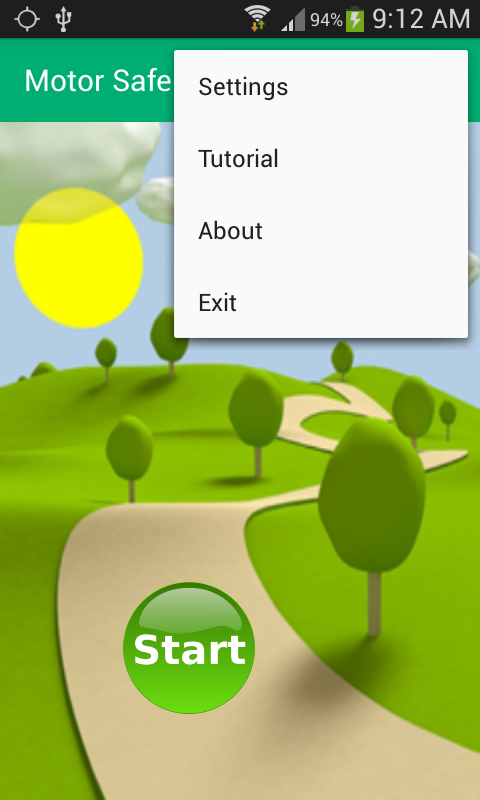
When application is installed on device, user must configuration before starting application.



**Figure 5-11 - User Interface**

In the first time, after application is installed on device, in the user interface, there are two options for user:

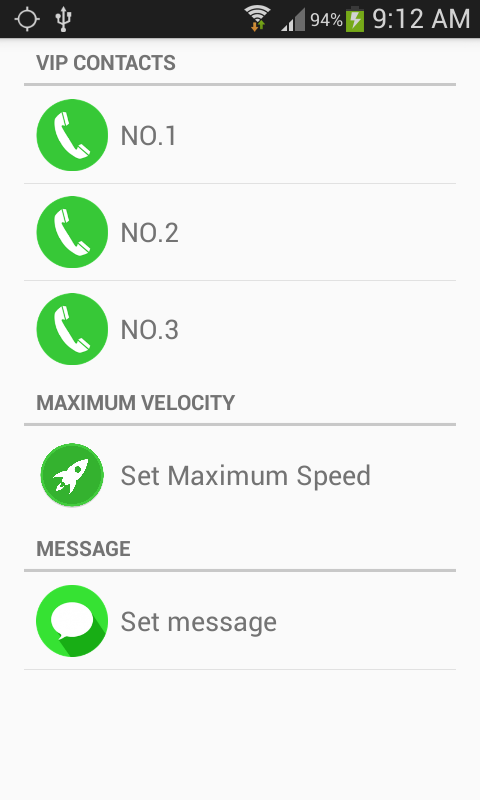
* Click to (1) to open menu setting (recommended).
* Click to (2) to start application. If user choose this immediately after installing application, some functions will do not work as expected.

****

**Figure 5-12 - Menu setting**

In menu setting, user have four options, this is:

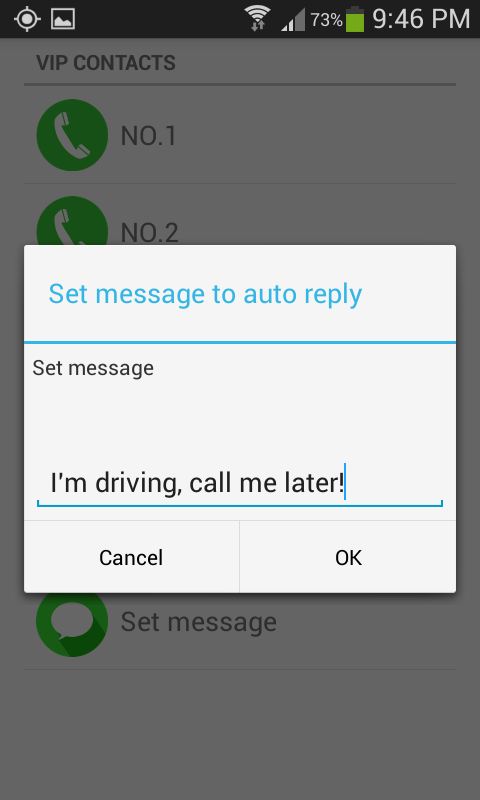
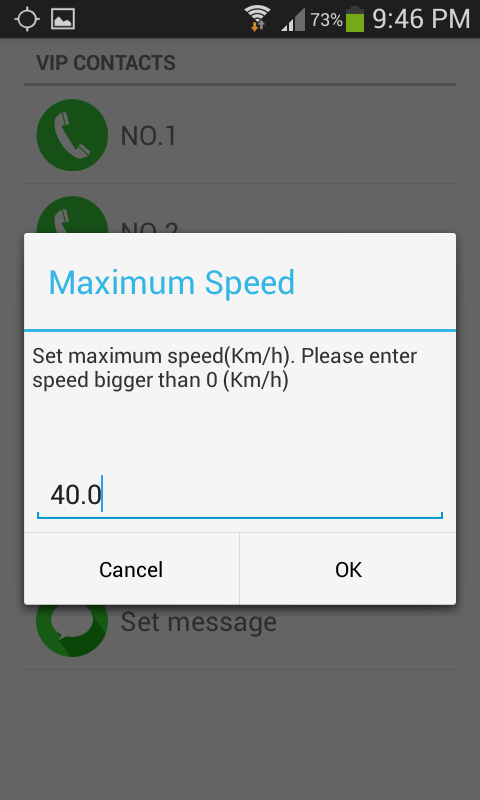
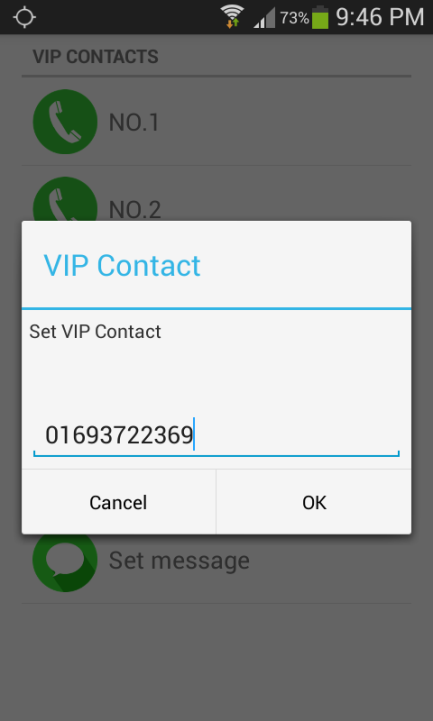
* Setting: set up some parameter for application
* Tutorial: give some guide to user use this application
* About: given some information about developer, version of application
* Exit: when user wants to finish application, user must click to “exit” to finish and out from application.

****

**Figure 5-13 - Setting initial values**

When user clicks to “setting” in menu setting, this layout will appear. In this layout, there three section to set up, this is:

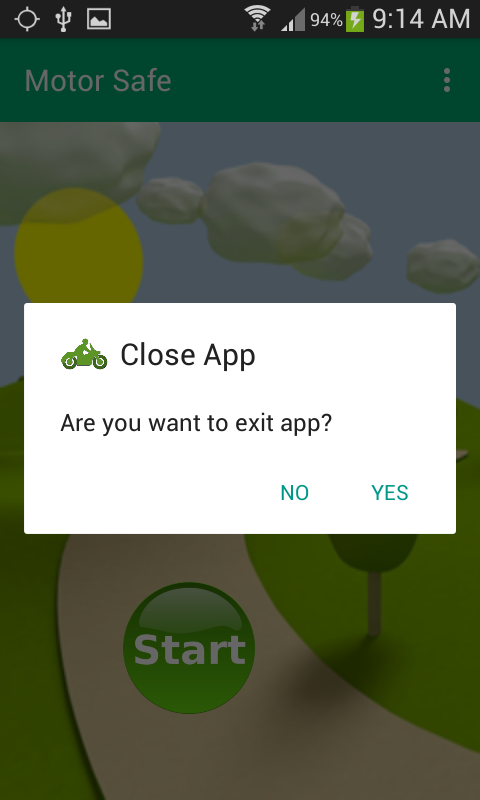
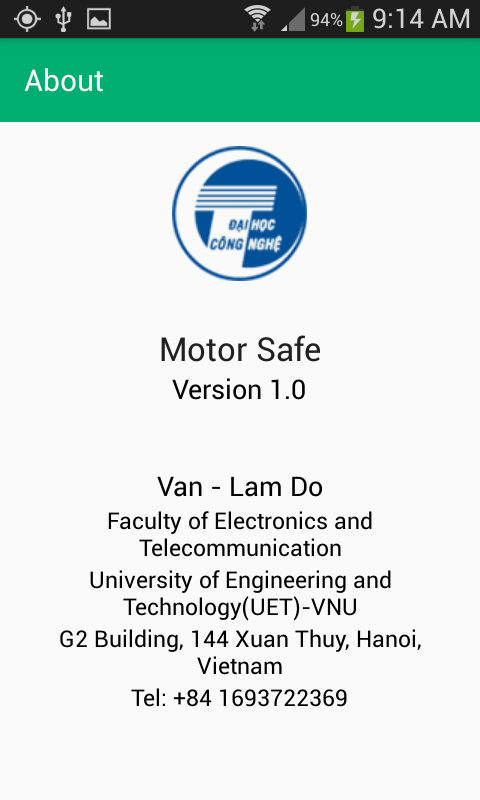
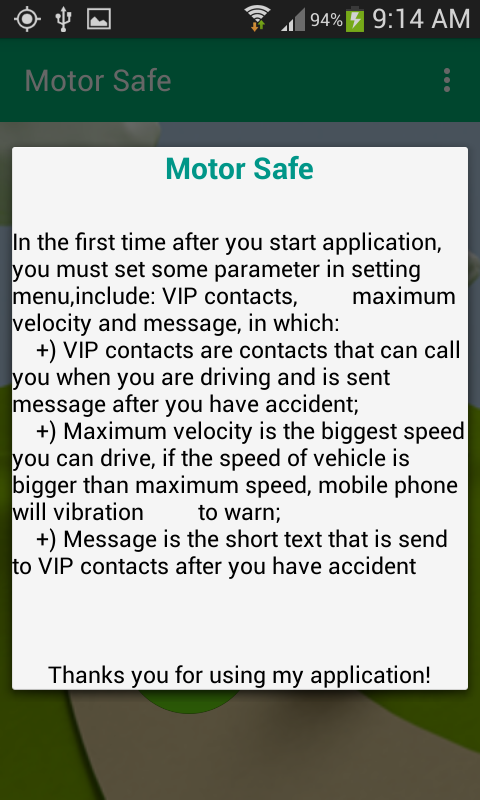
* VIP contacts
* Maximum velocity
* Set message



1. (b) (c)

**Figure 5-14 - (a) Setting the VIP contacts; (b) Setting the maximum speed; (c) Setting message**

* VIP contacts: application allows set up three VIP contacts. These contacts can call user when user is driving (this time, device changes to silent mode).
* Maximum velocity: in this section, user can set up the maximum speed such as 40 km/h,… Note: the maximum velocity must be bigger than 0 Km/h.
* Message: in this section, user can change default message which is set by application. This message is used to automatic sending to callers when they call user who is driving.

****

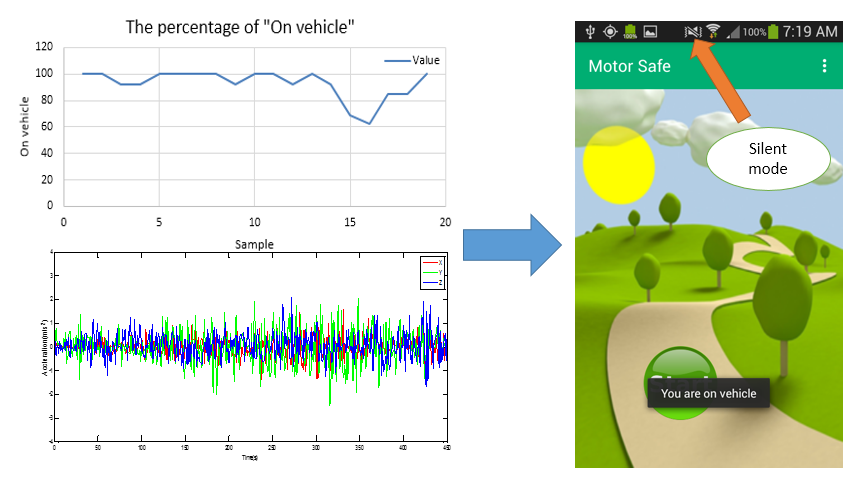
(a) (b) (c)

**Figure 5-15 - (a) Tutorial; (b) About; (c) Confirm to exit application**

When user click item on menu setting, new screen will display respectively for each items:

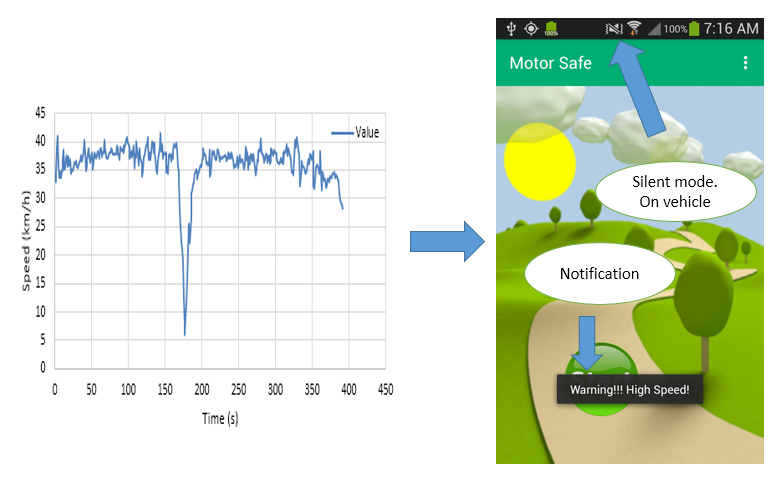
* Tutorial: this screen show guide for using application.
* About: show information about application and developer.
* Exit: new alert dialog will be showed, user can choose one of two options. If user choose “Yes”, application will finish and do not have any service run on background. On other hand, application will continue running.

When device detects user moving on motorbike, the mobile phone will change to silent mode:



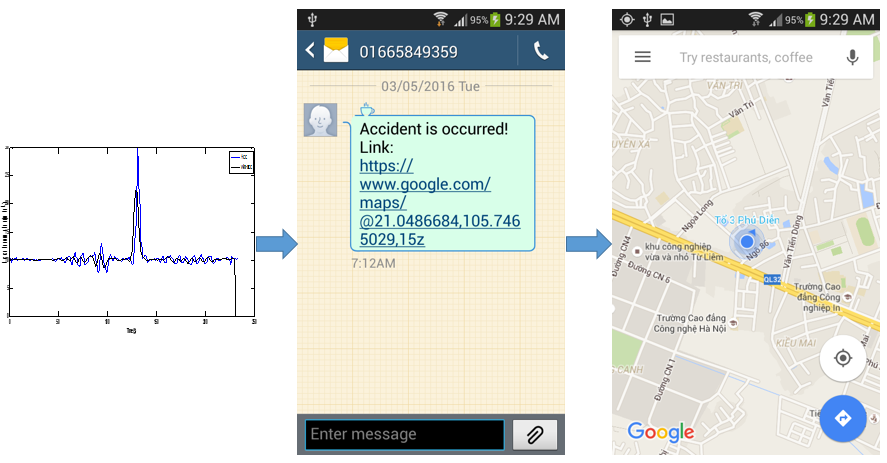
**Figure 5-16 - Change to silent mode while on vehicle**

When user moving with high speed which is bigger than maximum speed, the mobile phone will vibration and notification on screen:



**Figure 5-17 - Warning high speed**

In the situation, in which accident has occurred, the automatic message will send user’s parent, this contacts are defined in VIP contacts. This message has short text and address of accident.

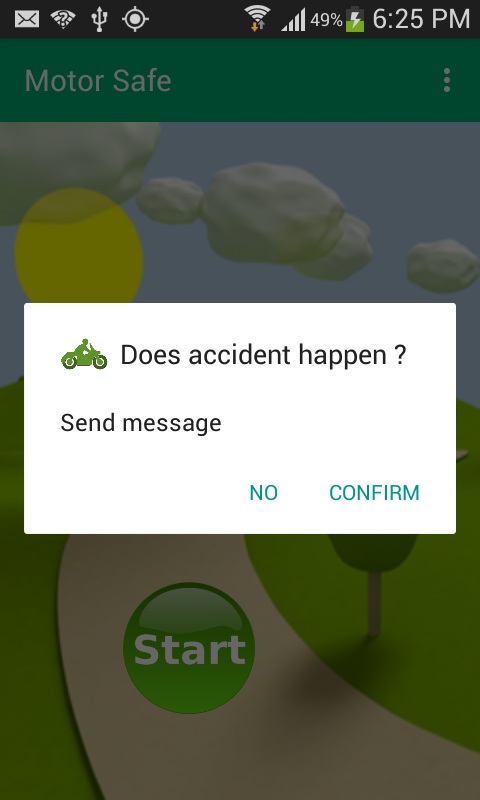


**Figure 5-18 - Message notification and address of accident**

When user’s parent have received message, they can click to link to find address of accident. This helps to work the emergency victims which is faster.

## **5.2. Discussions**

In the case “Detecting accidents”, the system cannot determine device free falling. So, in order to improve accuracy of application, I propose a method to detect device free falling. When device is free falling, there are alert notification displaying on screen:



**Figure 5-19 – Notification when accident or free falling**

There are two options for user:

* No: when user clicks to this option, this mean no accident occur and message discard
* Confirm: user confirm accident occurring and message will automatic send to user’s parent.

This alert notification will show on screen in 10 second, after this time, by default configuration, an automatic message will send to user’s parent.

# Chapter 6

# **CONCLUSIONS**

## **6.1. Conclusions**

In this thesis, I had success to develop an application which can be installed on Android smartphones. For my phone, three functions of this application work well. This application has several features such as: automatic detecting the status of users when they are driving, automatic rejecting incoming calls with SMS while driving except VIP contacts, and giving notification to users when they moving with high speed which is over the maximum speed and the last feature, this is application can detect an accident, automatic sending message notification to user’s relatives. By using “Motor Safe” user will be safer in traffic, because they do not be annoyed by incoming calls, warning when the speed of the vehicle in excess of a certain speed and emergency after the accident occurred is faster. ”Motor Safe” is suitable for everybody and every age.

Through this thesis, I have learnt how to handle the data from GPS, accelerometer sensor; how to use and combine components in android. That knowledge will be definitely useful for my work in the future.

## **6.2. Future works**

In the future, beside of improving my application, making this application will be more user friendly. I will integrate new sensor (camera, microphone) to increase accuracy of my application.

# **Reference**

[1] Wei-Meng Lee, “Beginning Android Application Development in full color”, March, 2011.

[2] Wallace Jackson, “Android Apps for Absolute Beginners”, 2nd edition, 2012.

[3] Google APIs for Android, “[*https://developers.google.com*](https://developers.google.com)*”*, November 5, 2015.

[4] Detected Activity, “[*https://developers.google.com*](https://developers.google.com)”, November 5, 2015.

[5] Location Strategies, “http://developer.android.com”.

[6] Garmin, “GPS guide for beginner”, December, 2000.

[7] Android tutorial, Simply Easy Learning, 216 pages, “tutorialsspoint.com”.

[8] Deepak Punetha, Deepak Kumar, Vartika Mehta, “Design and Realization of the Accelerometer based Transportation System”, International Journal of Computer Applications (0975 - 8887), July 2012

[9] Android architecture, “http://androidpace.com”, January 19, 2016