# SKRIPSI

# PENDETEKSI GERAKAN KEPALA DENGAN GOOGLE CARDBOARD

EGA PRIANTO

NPM: 2013730047

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
«tahun»

UNDERGRADUATE THESIS

# HEAD MOTION DETECTOR USING GOOGLE CARDBOARD


EGA PRIANTO

NPM: 2013730047

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
«tahun»

# LEMBAR PENGESAHAN

# PENDETEKSI GERAKAN KEPALA DENGAN GOOGLE CARDBOARD


## EGA PRIANTO

## NPM: 2013730047


**Bandung, «tanggal» «bulan» «tahun»**

**Menyetujui,**

| Pembimbing Utama | Pembimbing Pendamping |
|---|---|
| **Pascal Alfadian, M.Comp.** | **«pembimbing pendamping/2»** |
| **Ketua Tim Penguji** | **Anggota Tim Penguji** |
| **«penguji 1»** | **«penguji 2»** |

**Mengetahui,**

**Ketua Program Studi**

**Mariskha Tri Adithia, P.D.Eng**

# PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

**PENDETEKSI GERAKAN KEPALA DENGAN GOOGLE CARDBOARD**

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal «tanggal» «bulan» «tahun»

Meterai

Ega Prianto
NPM: 2013730047

## ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia» Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.



**Kata-kata kunci:** «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Indonesia»

## ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris» Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.



**Keywords:** «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Inggris»

*«kepada siapa anda mempersembahkan skripsi ini...?»*

# KATA PENGANTAR

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Bandung, «bulan» «tahun»

Penulis

# DAFTAR ISI

# DAFTAR GAMBAR

# DAFTAR TABEL

# BAB 1

# PENDAHULUAN

## 1.1 Latar Belakang

*Virtual Reality* adalah teknologi yang mampu membuat penggunanya dapat berinteraksi dengan lingkungan buatan oleh komputer, suatu lingkungan yang sebenarnya ditiru atau hanya ada di dalam imajinasi.[1] *Virtual Reality* membuat pengalaman sensorik, di antaranya penglihatan, pendengaran, perabaan, dan penciuman secara *"'synthetic"'*,*"'illusory"'*, atau *"'virtual"'* .[2] Gawai *Virtual Reality* terbaru sekarang yaitu dengan menggunakan *head-mounted display*, Google Cardboard salah satunya. *Head-mounted display* adalah menempatkan layar di kepala, sehingga pengguna hanya dapat melihat tampilan yang ditampilkan oleh layar.[3]

Google Cardboard[4] adalah gawai murah yang terbuat dari kardus untuk dapat merasakan pengalaman *virtual reality* dengan Android atau iOS *smartphone*. Kita dapat membuat Google Cardboard kita sendiri secara gratis dengan mengunduh templatenya di website Google Cardboard. Template tersebut membantu dalam merakit kardus dengan dibentuk, dilipat dan digunting sedemikian rupa sehingga berbentuk kacamata. Bahan-bahan untuk merakit Google Cardboard hanyalah kardus, lem, dan lensa dengan spesifikasi tertentu.

Pada Gawai Google Cardboard cara pengguna memberikan *input* kepada program sangatlah terbatas. Cara tersebut hanyalah dengan gerakan kepala dan tombol magnet. Tombol magnet ini pun terkadang tidak berfungsi dengan baik, karena bergantung pada medan magnet yang di deteksi oleh *smartphone* yang digunakan. Cara lainnya agar dapat memberikan *input* kepada program adalah dengan menghubungkan *smartphone* yang digunakan dengan *bluetooth controller*. Skripsi ini akan membuat aplikasi untuk menambahkan cara baru memberikan *input* pada Google Cardboard. Pada skripsi ini, akan dibuat dua buah perangkat lunak. Perangkat lunak pertama akan digunakan untuk menganalisis data yang didapat dari sensor-sensor pada Android. Perangkat lunak kedua akan dapat mendeteksi gerakan kepala penggunanya ketika sedang menggeleng atau mengangguk. Pada perangkat lunak kedua ini akan memberikan *input* baru kepada program virtual reality. Jenis *input* yang diberikan kepada komputer hanya ya(mengangguk) atau tidak(menggeleng).

Agar *Virtual Reality* menggunakan Google Cardboard dapat berjalan dengan sempurna, dibutuhkan *smartphone* yang memiliki 3 jenis sensor. Ketiga sensor itu adalah Magnetometer, Accelerometer, dan Gyroscope.[5] Jika salah satu sensor itu tidak ada, tampilan gambar pada *Virtual Reality* akan tidak akurat atau lambat. Magnetometer digunakan untuk mengetahui arah pandang pengguna. Accelerometer digunakan untuk mengetahui arah gaya gravitasi.[6] Gyroscope digunakan untuk mengetahui percepatan perputaran sudut kepala pengguna. Ketiga sensor ini juga harus

menggunakan sensor 3 sumbu. Ketiga sensor tersebut tidak hanya berfungsi agar dapat menjalankan *Virtual Reality* dengan Google Cardboard dan *smartphone*, tetapi juga dapat berfungsi sebagai pendeteksi gerakan kepala.

## 1.2 Rumusan Masalah

- Bagaimana cara menampilkan grafik data yang diambil dari sensor-sensor pada *smartphone*?
- Bagaimana cara mendeteksi gerakan kepala dari data yang didapat dari sensor-sensor pada *smartphone*?

## 1.3 Tujuan

- Mengetahui cara untuk menampilkan grafik data dari sensor-sensor pada *smartphone*.
- Mengetahui cara mendeteksi gerakan kepala dari data yang didapat dari sensor-sensor pada *smartphone*.

## 1.4 Batasan Masalah

Penelitian ini dibuat berdasarkan batasan-batasan sebagai berikut:

1. Program pertama yang akan dibuat dalam skripsi ini hanya akan digunakan untuk membantu dalam menganalisis sensor.
2. Program kedua yang akan dibuat hanya dapat melakukan pendeteksi gerakan kepala khusus untuk mengangguk dan menggeleng saja.
3. 

## 1.5 Metode Penelitian

Berikut adalah metode penelitian yang digunakan dalam penelitian ini:

1. Melakukan studi literatur tentang Android SDK, Google VR SDK, Quaternion, *Sensor Fusion*, dan algoritma *Head Motion Detection*.
2. Merancang dan membuat aplikasi untuk menampilkan grafik sensor-sensor pada *smartphone* Android.
3. Menganalisis aplikasi-aplikasi sejenis.
4. Merekam dan menganalisis grafik dari sensor-sensor pada *smartphone* ketika mengangguk dan menggeleng.
5. Menganalisis metode pendeteksi gerakan kepala.
6. Merancang aplikasi untuk mendeteksi gerakan kepala
7. Mengimplementasi algoritma pendeteksi gerakan kepala ke aplikasi *virtual reality*.

## 1.6 Sistematika Penulisan

Setiap bab dalam penelitian ini memiliki sistematika penulisan yang dijelaskan ke dalam poin-poin sebagai berikut:

1. Bab 1: Pendahuluan, yaitu membahas mengenai gambaran umum penelitian ini. Berisi tenang latar belakang, rumusan masalah, tujuan, batasan masalah, metode penelitian, dan sistematika penulisan.

2. Bab 2: Dasar Teori, yaitu membahas teori-teori yang mendukun berjalannya penelitian ini. Berisi tentang Android SDK, Google VR SDK, Quaternion, dan algoritma *Head Motion Detection*.

3. Bab 3: Analisis, yaitu membahas mengenai analisa masalah. Berisi tentang analisis aplikasi-aplikasi sejenis, analisis grafik dari sensor-sensor pada *smartphone* ketika mengangguk dan menggeleng, analisis metode pendeteksi gerakan kepala.

4. Bab 4: Perancangan yaitu membahas mengenai perancangan

# BAB 2

# DASAR TEORI

Pada bab ini akan dijelaskan dasar-dasar teori mengenai Android SDK, Google VR SDK,Quaternion, *Sensor Fusion*, dan algoritma *Head Motion Detection*.

## 2.1 Android SDK

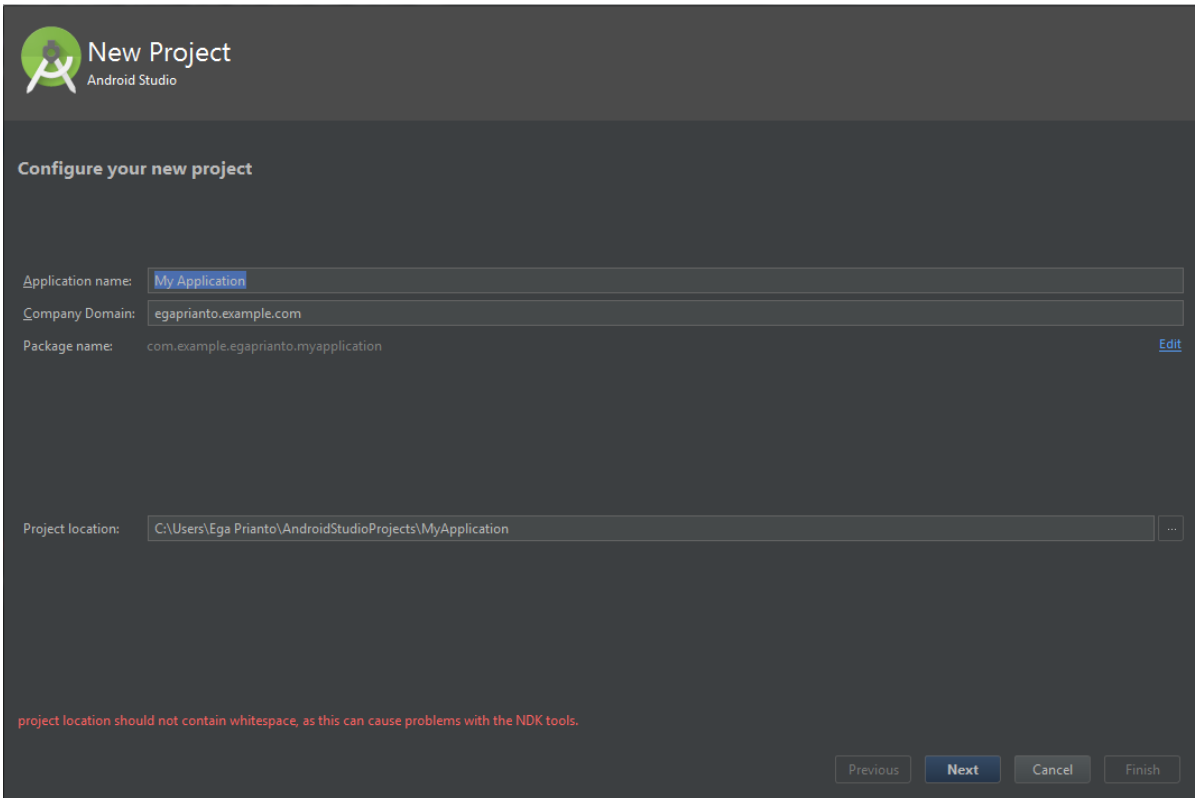
Android SDK(*software development kit*) adalah kumpulan *source code, development tools, emulator*,[7] dan semua *libraries* untuk membuat suatu aplikasi untuk *platform* Android. IDE(*integrated development environment* yang official untuk Android SDK adalah Android Studio. Android Studio dapat di download di halaman website https://developer.android.com/studio/index.html, sekaligus dengan Android SDKnya.

### 2.1.1 Membuat Android *Project*

Pada bagian ini akan ditunjukkan bagaimana membuat Android Project baru dengan Android Studio.

1. Membuah suatu *project* baru di android studio dengan memilih **File >New Project**
2. Kemudian isilah **Application Name** dan **Company Domain** pada halaman **New Project** sesuai dengan keinginan seperti yang ditunjukkan pada Gambar 2.1. Android studio akan mengisikan lokasi project dan nama package secara otomatis.
3. Klik **Next**
4. Pada halaman **Target Android Devices**, klik **Next**.
5. Pada halaman **Add an Activity to Mobile**, pilih **Empty Activity** kemudian klik **Next**.
6. Pada halaman **Target Android Devices**, klik **Next**.
7. Pada halaman **Customize the Activity**, klik **Finish**.

Setelah diproses, Android Studio akan menampilkan aplikasi android dengan tulisan "'Hello Wold"'.




Gambar 2.1: Tampilan ketika membuat suatu *project* baru

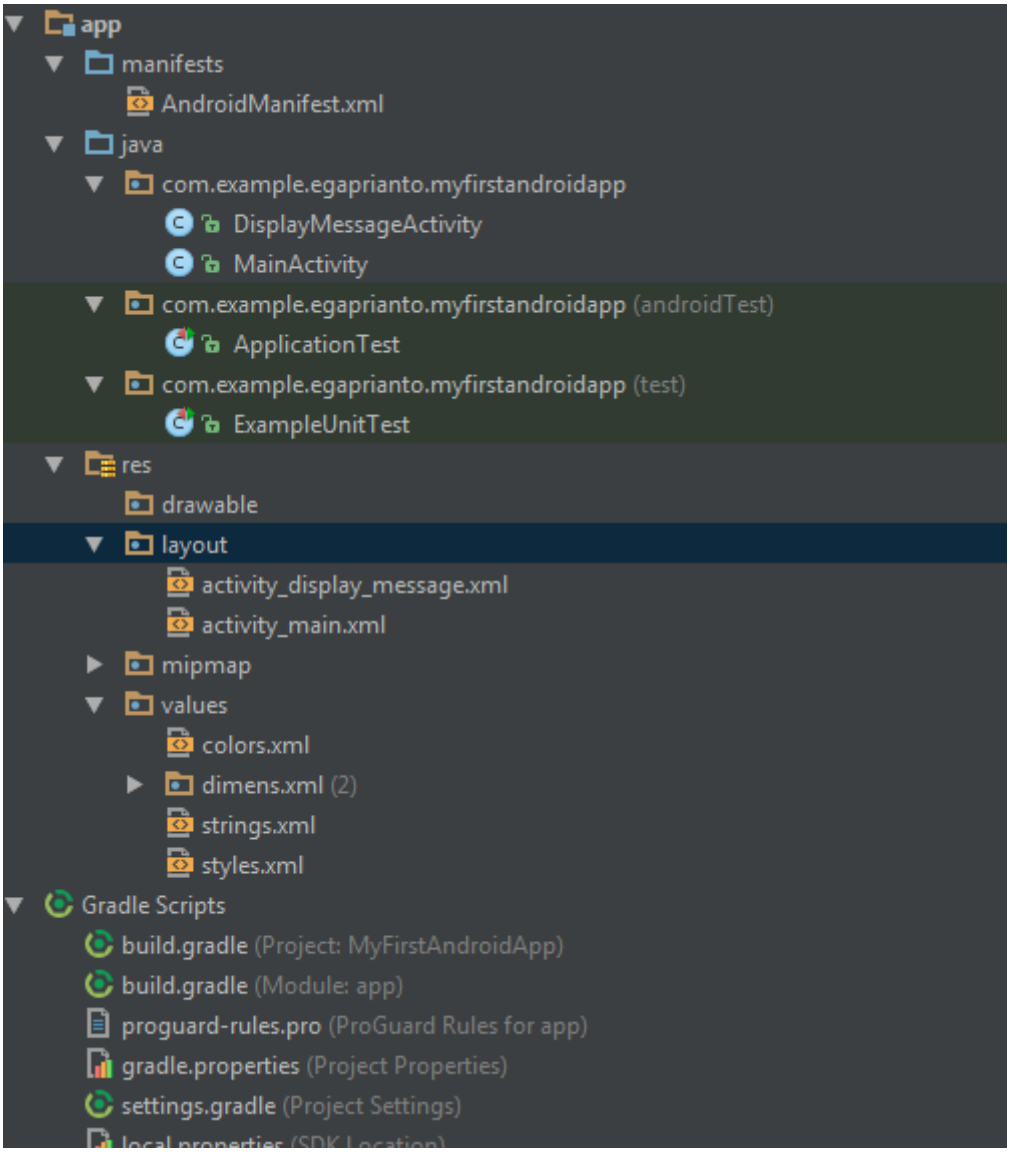
### 2.1.2 Struktur File Android Studio Project

Pada saat **project** baru telah dibuat, Android Studio akan membuatkan direktori-direktori standar(Gambar 2.2). Berikut adalah sebagian penjelasan dari hal yang perlu di perhatikan pada struktur tersebut:

1. File **app>java>com.example.myfirstapp>MainActivity.java** mengandung definisi kelas untuk Activity Utama.
2. File **app>res>layout>activity_main.xml** adalah XML file untuk mendefinisikan*layout*(*User Interface*) dari suatu activity.
3. File **app>manifests>AndroidManifest.xml** mendeskripsikan karakteristik fundamental dari suatu aplikasi dan juga mendefinisikan setiap komponennya.
4. **Gradle Scripts>build.gradle** Android Studio menggunakan Gradle untuk melakukan *compile* dan *build* aplikasi Android. Pasti ada file build.gradle pada modul dan pada keseluruhan *project*.
5. Folder **res** merupakan tempat untuk menyimpan sumber-sumber aplikasi seperti layout, icon, gambar, nilai konstan, dan lain-lain.

### 2.1.3 Membuat User Interface

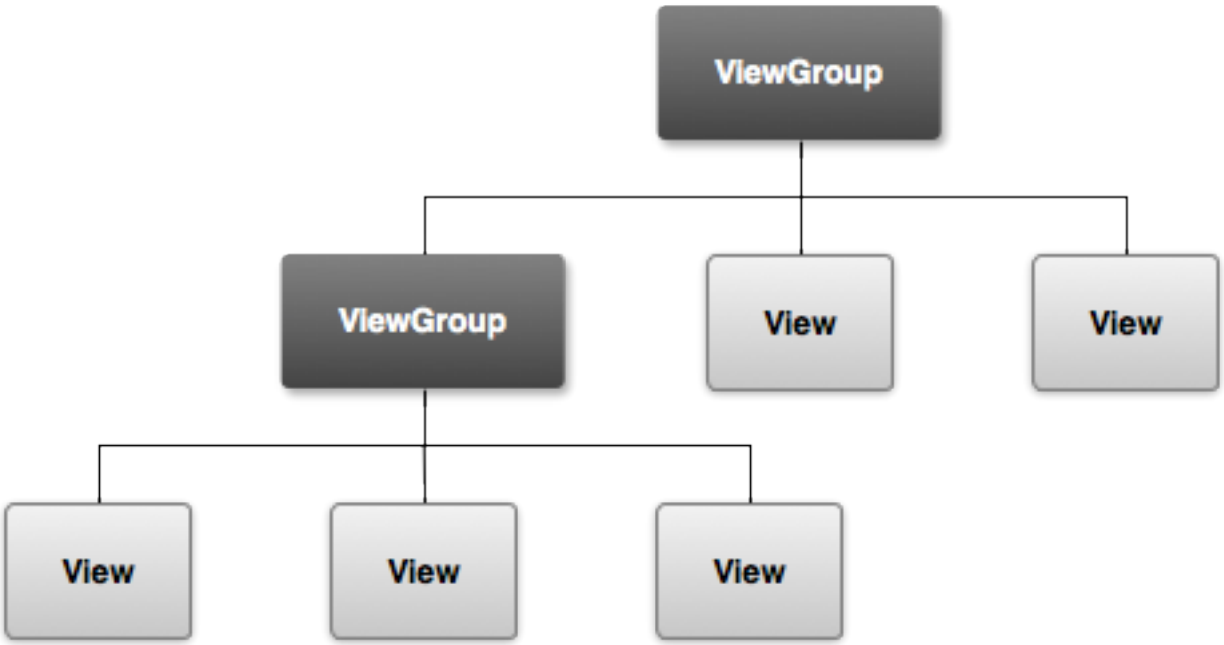
Pada subbab ini akan dijelaskan bagaimana membuat layout di XML termasuk *text field* dan *button*




Gambar 2.2: Tampilan struktur direktori pada *project* Android Studio

### Hierarki *Graphical User Interface*(GUI) untuk Aplikasi Android

GUI untuk aplikasi Android dibuat dengan hierarki dari objek View dan ViewGroup(Gambar 2.3). Objek-objek dari View biasanya adalah *UI Widgets* seperti *button* atau *text field*. Objek-objek dari ViewGroup tidak terlihat oleh *view containers* yang mendefinisikan bagaimana *child views* ditata seperti *grid* atau *vertical list* Android menggunakan XML vocabulary yang berkorespondesi kepada




Gambar 2.3: Illustrasi bagaimana percabangan objek ViewGroup pada *layout* dan mengandung objek View lainnya.

*subclasses* dari View dan ViewGroup, sehingga Ui dapat didefinisikan dalam XML menggunakan hierarki dari elemen UI.

### Attribut-attribut Objek View

Pada subbab ini akan dijelaskan attribut-attribut object View yang digunakan dalam membuat GUI pada file activity_main.xml

- **android:id** Attribut ini merupakan pengidentifikasi dari suatu view. Attribut ini dapat digunakan untuk menjadi referrensi object dari kode aplikasi seperti membaca dan memanipulasi objek tersebut (Akan dijelaskan lebih lanjut pada subbab **??**). Tanda *at* dibutuhkan ketika mereferensi object dari suatu XML. Tanda *at* tersebut diikuti dengan tipe(id pada kasus ini), *slash*, dan nama (edit_message pada Gambar 2.4). Tanda tambah (+) sebelum tipe hanya dibutuhkan jika ingin mendefinisikan *resource ID* untuk pertama kalinya.

- **android:layout_width** dan **android:layout_height** Attribut ini digunakan untuk mendefinisikan panjang dan lebar dari suatu objek View. Daripada menggunakan besar spesifik untuk panjang dan lebarnya, lebih baik menggunakan "'wrap_content"' yang menspesifikasi viewnya hanya akan sebesar yang dibutuhkan untuk memuat konten-konten dari View. Jika

menggunakan "‘match_parent"’ pada kasus Gambar 2.4 View akan memenuhi layar, karena besarnya akan mengikuti besar dari paretnya LinearLayout.

- **android:hint** Attribut ini merupakan *default string* untuk di tampilkan ketika objek View kosong. Daripada menggunakan *hard-coded string* sebagai *nilai* untuk ditampilkan, *value* "‘@string/edit_message"’ mereferensi ke sumber string pada file yang berbeda. Karena mereferensi ke sumber konkrit, maka tidak dibutuhkan tanda tambah (+). Nilai string ini akan di simpan pada file Strings.xml yang ditunjukkan pada Gambar 2.2.
- **android:onClick** Attribut ini akan memberitahu *system* untuk memanggil method sendMessage() di Activity ketika user melakukan klik pada *button* tersebut. Agar *system* dapat memanggil method yang tepat, method tersebut harus memenuhi kriteria berikut.
  - *Access Modifier* haruslah *public*.
  - Harus *void return value*nya.
  - Mempunyai View sebagai parameter satu-satunya. View ini akan diisi dengan View yang di klik.

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
        <EditText android:id="@+id/edit_message"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:hint="@string/edit_message" />
        <Button
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:text="@string/button_send" />
</LinearLayout>
```

Gambar 2.4: Contoh attribute pada objek View

## 2.2 Google VR SDK

## 2.3 Quaternion

Pada Android SDK **SensorEvent.values** [8] tipe sensor **Sensor.TYPE_ROTATION_VECTOR**, yaitu tipe sensor yang mendeteksi vektor perputaran pada *smartphone*. Tipe sensor ini dijelaskan akan mengembalikan nilai-nilai dari komponen quaternion. Quaternion[9] adalah objek penggabungan dari suatu skalar dengan suatu vektor, sesuatu yang tidak dapat didefinisikan dalam *linear algebra* biasa.

### 2.3.1 Struktur Ajabar

Struktur-Struktur aljabar yang digunakan adalah Bilangan Kompleks, Konjugasi Kompleks, *Quaternion Algebra*, dan Operasi-operasi pada *Quaternion*.

#### Bilangan Kompleks

Dalam matematika, bilangan kompleks adalah bilangan yang berbentuk

$$a + bi$$

[9]
dan $a$ dengan $b$ merupakan bilangan riil, dan $i$ merupakan bilangan imajiner tertentu yang memiliki sifat $i^2 = -1$.

Berikut notasi-notasi bilangan kompleks :

$$(a + bi) + (c + di) = (a + c) + i(b + d)$$

$$(a + bi)(c + di) = (acbd) + (bc + ad)i$$

$$a(c + id) = ac + iad$$

[9]

Bilangan kompleks dapat digunakan untuk rotasi dua dimensi. Dengan $a = \cos(\frac{\theta}{2})$, dan $b = \sin(\frac{\theta}{2})$, kemudian akan dikalikan dengan vektor yang ingin di putar, dengan sumbu putar adalah titik pusat.

#### Konjugasi Kompleks

Berhubungan dengan bentuk umum bilangan kompleks,

$$z = a + ib$$

[9]
satu bilangan dikatakan konjugasi kompleks jika

$$\bar{z} = a - ib$$

[9]
Dari kedua persamaan diatas dapat dihasilkan :

$$z + \bar{z} = 2a$$

dan,

$$z\bar{(z)} = a^2 + b^2 = |z|^2$$

[9]

### 2.3.2 *Quaternion Algebra* dan Operasi-operasi pada *Quaternion*

Pada buku "Quaternions and Rotation Sequences : A Primer with Applications to Orbits, Aerospace, and Virtual Reality" disebutkan :

> In 1843 Hamilton invented the so-called hyper-complex number of rank 4, to which he gave the name *quaternion*. Crucial to this invention was his celebrated rule
>
> $$i^2 = j^2 = k^2 = ijk = -1$$
>
> for dealing with the operations on the vector part of the quaternion .

[9]
Dari kutipan diatas bilangan *hyper-complex* tingkat empat yang dapat disebut juga *quaternion* memiliki satu aturan yang dicetuskan oleh Hamilton yaitu:

$$i^2 = j^2 = k^2 = ijk = -1$$

Untuk hasil dari perkalian dua *quaternion* memiliki aturan yang lebih rumit, sehingga memiliki aturan-aturan khusus. Berikut aturan-aturan khususnya :

$$\begin{aligned} ij &= k = -ji \\ jk &= i = -kj \\ ki &= j = -ik \end{aligned} \tag{2.1}$$

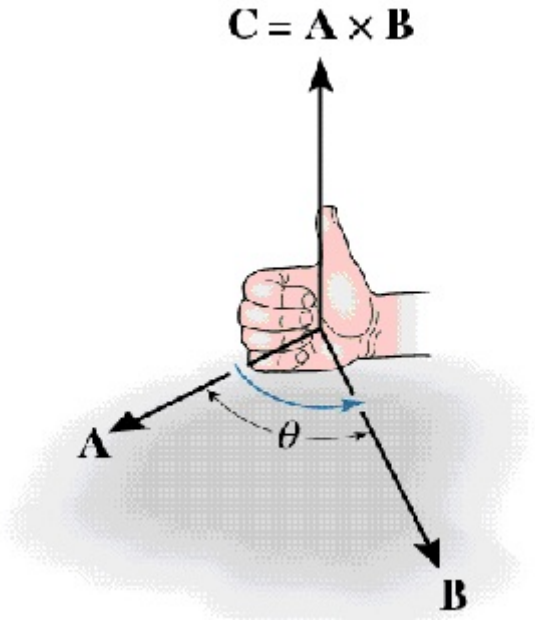
Perhatikan bahwa ketiga persamaan diatas mirip dengan cross product vektor sesuai dengan aturan tangan kanan (*right-hand rule*). Dapat di katakan pada Gambar 2.5, $A$ berperan sebagai $i$, $B$ berperan sebagai $j$, dan $C$ berperan sebagai $k$. Sehingga terpenuhi sesuai dengan persaman-persamaan 2.1

*Quaternion* memiliki persamaan umum yang memiliki empat bilangan riil atau skalar. Persamaan tersebut adalah

$$q = q_0 + iq_1 + jq_2 + kq_3$$

[9] Sama seperti pada bilangan kompleks, *quaternion* juga memiliki konjugasi kompleksnya. Berikut adalah konjugasi kompleks *quaternion*

$$q = q_0 - iq_1 - jq_2 - kq_3$$




Gambar 2.5: Right-hand rule dalam *cross product* vektor

# DAFTAR REFERENSI


[1] T. Parisi, *Learning virtual reality: developing immersive experiences and applications for desktop, web, and mobile.* O'Reilly Media, 1 ed., 2015.

[2] G. J. Kim, *Designing virtual reality systems: the structured approach.* Springer, 2005.

[3] J. Vince, *Introduction to virtual reality.* Springer, 2004.

[4] "Google cardboard." https://vr.google.com/cardboard/. [Online; diakses 10-September-2016].

[5] "Sensor types." https://source.android.com/devices/sensors/sensor-types.html. [Online; diakses 10-September-2016].

[6] G. Bleser and D. Stricker, "Advanced tracking through efficient image processing and visual–inertial sensor fusion," *Computers & Graphics*, vol. 33, no. 1, pp. 59–72, 2009.

[7] A. Developers, "What is android," 2011.

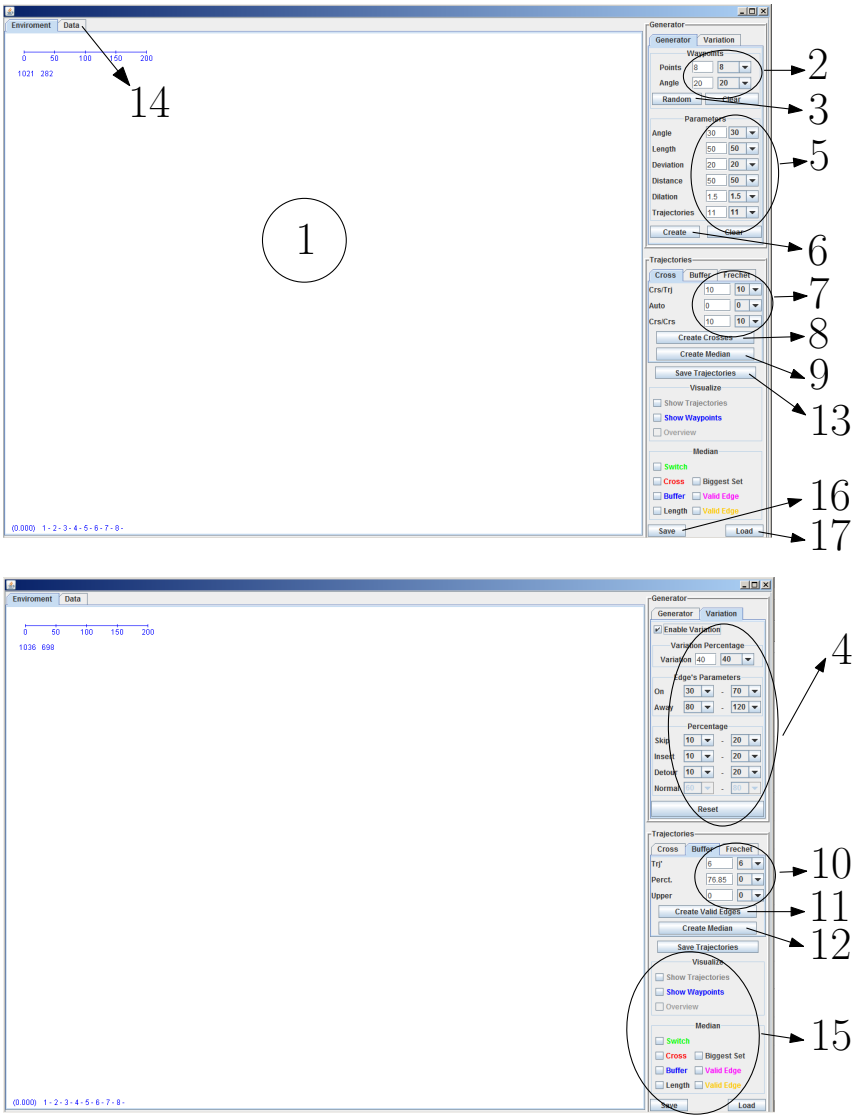
[8] "Android 7.0 nougat!." https://developer.android.com/. [Online; diakses 12-September-2016].

[9] J. B. Kuipers, *Quaternions and Rotation Sequences : A Primer with Applications to Orbits, Aerospace, and Virtual Reality.* Princeton University Press, 1998.

# LAMPIRAN A

# THE PROGRAM

The interface of the program is shown in Figure A.1:




Gambar A.1: Interface of the program

Step by step to compute the median trajectory using the program:

1. Create several waypoints. Click anywhere in the "Environment" area(1) or create them automatically by setting the parameters for waypoint(2) or clicking the button "Random"(3).

2. The "Variation" tab could be used to create variations by providing values needed to make them(4).

3. Create a set of trajectories by setting all parameters(5) and clicking the button "Create"(6).

4. Compute the median using the homotopic algorithm:

   - Define all parameters needed for the homotopic algorithm(7).
   - Create crosses by clicking the "Create Crosses" button(8).
   - Compute the median by clicking the "Compute Median" button(9).

5. Compute the median using the switching method and the buffer algorithm:

   - Define all parameters needed for the buffer algorithm(10).
   - Create valid edges by clicking the "Create Valid Edges"button(11).
   - Compute the median by clicking the "Compute Median"button(12).

6. Save the resulting median by clicking the "Save Trajectories" button(13). The result is saved in the computer memory and can be seen in "Data" tab(14)

7. The set of trajectories and its median trajectories will appear in the "Environment" area(1) and the user can change what to display by selecting various choices in "Visualize" and "Median" area(15).

8. To save all data to the disk, click the "Save"(16) button. A file dialog menu will appear.

9. To load data from the disk, click the "Load"(17) button.

# LAMPIRAN B

# THE SOURCE CODE

Listing B.1: MyFurSet.java

```
1
2  import java.util.ArrayList;
3  import java.util.Collections;
4  import java.util.HashSet;
5
6  /**
7   *
8   * @author Lionov
9   */
10
11 //class for set of vertices close to furthest edge
12 public class MyFurSet {
13     protected int id;                                    //id of the set
14     protected MyEdge FurthestEdge;                       //the furthest edge
15     protected HashSet<MyVertex> set;                     //set of vertices close to furthest edge
16     protected ArrayList<ArrayList<Integer>> ordered;     //list of all vertices in the set for each
           trajectory
17     protected ArrayList<Integer> closeID;                //store the ID of all vertices
18     protected ArrayList<Double> closeDist;               //store the distance of all vertices
19     protected int totaltrj;                              //total trajectories in the set
20
21     /**
22      * Constructor
23      * @param id : id of the set
24      * @param totaltrj : total number of trajectories in the set
25      * @param FurthestEdge : the furthest edge
26      */
27     public MyFurSet(int id,int totaltrj,MyEdge FurthestEdge) {
28         this.id = id;
29         this.totaltrj = totaltrj;
30         this.FurthestEdge = FurthestEdge;
31         set = new HashSet<MyVertex>();
32         ordered = new ArrayList<ArrayList<Integer>>();
33         for (int i=0;i<totaltrj;i++) ordered.add(new ArrayList<Integer>());
34         closeID = new ArrayList<Integer>(totaltrj);
35         closeDist = new ArrayList<Double>(totaltrj);
36         for (int i = 0;i <totaltrj;i++) {
37             closeID.add(-1);
38             closeDist.add(Double.MAX_VALUE);
39         }
40     }
41
42     /**
43      * set a vertex into the set
44      * @param v : vertex to be added to the set
45      */
46     public void add(MyVertex v) {
47         set.add(v);
48     }
49
50     /**
51      * check whether vertex v is a member of the set
52      * @param v : vertex to be checked
53      * @return true if v is a member of the set, false otherwise
54      */
55     public boolean contains(MyVertex v) {
56         return this.set.contains(v);
57     }
58 }
```