

SKRIPSI

VIRTUAL JOGGING APP UNTUK GOOGLE CARDBOARD



RICHARD WIJAYA

NPM: 2016730014

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2020

UNDERGRADUATE THESIS

VIRTUAL JOGGING APP FOR GOOGLE CARDBOARD



RICHARD WIJAYA

NPM: 2016730014

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2020**

LEMBAR PENGESAHAN

VIRTUAL JOGGING APP UNTUK GOOGLE CARDBOARD

RICHARD WIJAYA

NPM: 2016730014

Bandung, «**tanggal**» «**bulan**» 2020

Menyetujui,

Pembimbing

Pascal Alfadian, M.Comp.

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:

VIRTUAL JOGGING APP UNTUK 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» 2020

Meterai Rp. 6000

RICHARD WIJAYA
NPM: 2016730014

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia»

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

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris»

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

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

KATA PENGANTAR

«Tuliskan kata pengantar dari anda di sini . . . »

Bandung, «bulan» 2020

Penulis

DAFTAR ISI

KATA PENGANTAR	xv
DAFTAR ISI	xvii
DAFTAR GAMBAR	xix
DAFTAR TABEL	xxi
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	1
1.3 Tujuan	1
1.4 Batasan Masalah	2
1.5 Metodologi Penelitian	2
1.6 Sistematika Pembahasan	2
2 LANDASAN TEORI	3
2.1 Google VR	3
2.1.1 Google VR SDK	3
2.1.2 Aplikasi HelloVR	5
2.2 Google <i>StreetView API</i>	8
2.2.1 <i>API Key</i>	9
2.2.2 Penggunaan <i>StreetView API</i>	9
2.2.3 Atribut Parameter <i>StreetView API</i>	9
2.3 Google <i>Directions API</i>	11
2.3.1 Penggunaan <i>Directions API</i>	11
2.3.2 Hasil Pemanggilan <i>Directions API</i>	11
2.4 <i>Motion Sensor</i>	14
2.4.1 Deskripsi <i>Motion Sensor</i>	14
2.4.2 Deskripsi <i>Step Detector</i>	14
3 ANALISIS	15
3.1 Membuat Dunia VR	15
3.2 Menampilkan <i>StreetView API</i> pada Dunia VR	16
3.3 Integrasi dengan <i>Directions API</i>	16
3.3.1 Menentukan Atribut yang Bermanfaat	17
3.3.2 Cara Memanfaatkan Atribut	17
3.4 Cara Memanfaatkan <i>Step Detector</i> Sensor	17
DAFTAR REFERENSI	19

DAFTAR GAMBAR

2.1	Tampilan <i>UI</i> permainan <i>treasure hunt</i> pada aplikasi HelloVR	6
2.2	Gambar-gambar <i>assets</i> aplikasi HelloVR	6
2.3	Tampilan <i>UI Google Cloud</i> saat mengakses <i>API Key (API Key disamaraskan)</i>	9
2.4	Pemanggilan <i>StreetView API</i> yang berhasil, dengan parameter <i>size=600x300</i> dan <i>location=-6.8746537,107.6046282</i> dan <i>API key</i> yang sah (<i>API key</i> tidak disebutkan)	11
3.1	Tampilan <i>UI Blender</i> Blender versi 2.81	15
3.2	Pemanggilan <i>StreetView API</i> yang berhasil, dengan parameter <i>size=600x300</i> , <i>location=-6.8746537,107.6046282</i> , dan <i>API key</i> yang sah (<i>API key</i> tidak disebutkan), dengan atribut <i>heading</i> yang berbeda-beda	16
3.3	Contoh hasil penggabungan empat gambar <i>StreetView</i>	17

DAFTAR TABEL

2.1 Atribut-Atribut Opsional <i>StreetView API</i>	10
2.2 Atribut-Atribut Opsional <i>Directions API</i>	12

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Zaman modern adalah zaman saat profesi sudah dan sedang berkembang sehingga ada banyak sekali jumlah bidangnya serta perkembangannya. Mayoritas orang menekuni bidang-bidang profesi yang tak terhitung banyaknya untuk mengembangkan setiap bidang profesi. Hal ini menyebabkan kesulitan pengaturan waktu untuk berolahraga, yang adalah salah satu kebutuhan manusia untuk menjaga kesehatan. Salah satu aktivitas olahraga yang paling mudah dan tidak memerlukan gerakan yang sulit adalah berlari, namun kegiatan ini memerlukan lahan yang cukup besar agar dapat dilakukan dengan leluasa. Selain kebutuhan lahan, aktivitas berlari lebih menyenangkan jika dilakukan di luar rumah. Agar dapat dilakukan di dalam rumah, berlari dapat dilakukan di rumah adalah *treadmill*, akan tetapi masalah lingkungan yang monoton dan membosankan di dalam rumah membuat orang enggan untuk melakukan aktivitas berlari. Bila suasana dunia luar dapat dibawa ke dalam rumah, aktivitas ini dapat dilakukan di dalam rumah, tetapi suasana yang dirasakan adalah seperti di luar rumah.

Pada skripsi ini, akan dibuat sebuah perangkat lunak yang dapat menampilkan simulasi aktivitas berlari pada lingkungan yang diinginkan saat berlari di *treadmill*. Dengan menggunakan perangkat lunak tersebut, orang yang berlari dapat menikmati pemandangan yang dipilih saat berlari di dalam rumah sehingga merasa seperti berlari di lingkungan yang dipilih tersebut.

Teknologi yang dapat dimanfaatkan untuk membuat aplikasi VR untuk berlari adalah *Google Cardboard* dan sensor perangkat bergerak, dan untuk *Application Programming Interface (API)* yang digunakan adalah *Google Streetview API* dan *Google Directions API* [1] [2] [3].

1.2 Rumusan Masalah

Rumusan masalah yang ada pada skripsi ini adalah:

- Bagaimana memanfaatkan Google VR SDK for Android untuk menampilkan gambar dengan perangkat VR?
- Bagaimana menampilkan hasil dari *Google StreetView API* dalam bentuk VR?
- Bagaimana mengintegrasikan *Google Directions API*, gambar *Google StreetView* dan Google VR dalam perangkat lunak *virtual jogging*

1.3 Tujuan

Pada skripsi ini, hal-hal yang coba untuk dicapai adalah :

- Menggunakan Google VR SDK for Android untuk menampilkan gambar dengan *Google Cardboard*.
- Menampilkan hasil gambar dari *Google StreetView API* pada *Google Cardboard*.

- Mengintegrasikan *Google Directions API*, gambar dari *Google StreetView* dan Google VR (*Cardboard*) dalam perangkat lunak *virtual jogging*.

1.4 Batasan Masalah

Karena ada banyak tempat atau pemandangan di dunia ini, hanya beberapa lokasi saja yang dapat dipilih dari yang telah disediakan.

1.5 Metodologi Penelitian

Metodologi penelitian yang akan digunakan adalah sebagai berikut:

- Melakukan studi literatur dari situs-situs web tentang Google VR SDK, *StreetView API*, *Directions*, sensor tentang langkah, baik melalui media tulisan maupun video.
- Menampilkan pemandangan *StreetView* pada *Google Cardboards*.
- Mengintegrasikan *Google Directions API* dengan pemandangan *StreetView* yang telah ditampilkan pada *Google Cardboard*.
- Menganalisis sensor langkah dan menyinkronisasikannya dengan perubahan pemandangan *StreetView*.

Untuk membuat skripsi ini, peneliti . Setelah mempelajari semua komponen dari aplikasi yang akan dibuat, peneliti akan melakukan implementasi.

1.6 Sistematika Pembahasan

Dokumen dibagi ke dalam beberapa bab dengan sistematika pembahasan sebagai berikut:

1. Bab 1: Pendahuluan, yang menjelaskan gambaran umum penelitian. Mengandung latar belakang, rumusan masalah, tujuan,, batasan masalah, metodologi penelitian, serta sistematika pembahasan.
2. Bab 2: Dasar Teori, berisi landasan dari teori-teori yang berhubungan serta mendukung penelitian. Mengandung Google VR, Google *StreetView API*, Google *Directions API*, dan sensor.
3. Bab 3: Analisis, menjelaskan mengenai proses analisis masalah untuk menemukan solusi untuk menyelesaikan masalah. Mengandung cara membuat dunia VR, cara memanfaatkan *Google StreetView*, cara memanfaatkan *Google Directions API*, dan cara memanfaatkan sensor *step-detector*.

BAB 2

LANDASAN TEORI

Pada bab ini akan dijelaskan mengenai Google VR, Google StreetView API, Google Directions API, dan *motion sensor*.

2.1 Google VR

Google VR adalah teknologi yang diciptakan perusahaan Google untuk membuat pemandangan dunia maya yang terlihat nyata dengan menempatkan *smartphone* pada alat yang bernama *viewer* [1]. *Viewer* adalah alat untuk melihat dunia VR pada aplikasi VR di *smartphone*. Dua *viewer* yang diciptakan Google adalah *Google Daydream* dan *Cardboard* [1]. Yang digunakan penelitian ini adalah *Google Cardboard*. Google menyediakan sebuah alat bantu bagi pengembang perangkat lunak untuk memudahkan pengembangan aplikasi VR yang disebut Google VR SDK [1].

2.1.1 Google VR SDK

Google VR SDK adalah alat bantu berlisensi Apache 2.0 yang disediakan Google pada repository Github yang berisi kode program dari aplikasi *virtual reality* (VR) yang tersedia untuk Android (Java), Android NDK, Unity, dan iOS [4]. Secara umum, SDK ini dibuat agar pengembang perangkat lunak dapat mempelajari serta memanfaatkan teknologi VR yang disediakan Google [1]. Ada beberapa aplikasi yang tersedia pada SDK tersebut seperti aplikasi demo bernama HelloVR dan pemutar video dalam VR, tetapi penulis akan memanfaatkan bagian aplikasi demo HelloVR untuk Android Java pada Google VR SDK. Untuk menggunakan SDK ini, dibutuhkan perangkat lunak Android Studio 2.3.3 dan lebih tinggi, dengan Android SDK versi 7.1.1 (API Level 25) atau lebih tinggi [1].

Google menyediakan *library* untuk bahasa pemrograman Java pada *package com.google.vr.sdk.base* agar dapat digunakan pengembang perangkat lunak untuk membuat aplikasi VR [5]. Berikut adalah beberapa *class* dan *interface* dari *package com.google.vr.sdk.base*:

1. GvrView.StereoRenderer

Interface untuk *renderer* yang menyerahkan seluruh penyajian pemandangan stereo pada pemandangan (*view*). *Method-emethod* abstrak yang dimiliki *interface* ini adalah:

- **public abstract void onDrawEye (Eye eye)**
Method yang menggambar pemandangan untuk satu mata. *Method* ini tidak memiliki nilai yang dikembalikan ataupun *exception*.
- **public abstract void onFinishFrame (Viewport viewport)** *Method* yang dipanggil sebelum *frame* selesai dibuat. Parameter yang dimiliki *method* ini adalah:
 - **Viewport viewport**: *Object* *viewport* yang dari *GL surface*.*Method* ini tidak memiliki nilai yang dikembalikan ataupun *exception*.

- `public abstract void onNewFrame (HeadTransform headTransform)` Method untuk menggambar perubahan *frame* dari pemandangan. Parameter yang dimiliki *method* ini adalah:

- `HeadTransform headTransform`: *Object* dari kelas *headtransform*, yang mendefinisikan perubahan posisi kepala pengguna.

Method ini tidak memiliki nilai yang dikembalikan, juga tidak memiliki *exception*.

- `public abstract void onRendererShutdown()` Method yang dipanggil ketika *thread renderer* dari pemandangan berhenti atau dimatikan, ketika method `onSurfaceCreated` dipanggil. *Method* ini tidak memiliki parameter, nilai yang dikembalikan, maupun *exception*.

- `public abstract void onSurfaceChanged (int width, int height)` Method yang terpanggil ketika ada perubahan dimensi pada permukaan dunia VR. Parameter yang dimiliki *method* ini adalah:

- `int width`: Nilai dari lebar pemandangan satu *eye* dalam satuan *pixel*.
 - `int height`: Nilai dari tinggi pemandangan satu *eye* dalam satuan *pixel*.

- `public abstract void onSurfaceCreated (EGLConfig config)` Method untuk membuat dunia VR. Parameter yang dimiliki *method* ini adalah:
 - `EGLConfig config`: Konfigurasi EGL yang digunakan untuk membuat permukaan dunia VR.

2. AndroidCompat Utility class yang disediakan Android untuk menggunakan fitur-fitur VR.

- `public static void setSustainedPerformanceMode (Activity activity, boolean enabled)`
- `public static boolean setVrModeEnabled (Activity activity, boolean enabled)`
- `public static boolean trySetVrModeEnabled (Activity activity, boolean enabled)`

3. Eye

Class yang mendefinisikan rincian *rendering* stereo dari satu *eye*.

- `public float[] getEyeView ()`
- `public FieldOfView getFov ()`
- `public float[] getPerspective (float zNear, float zFar)`
- `public boolean getProjectionChanged ()`
- `public int getType ()`
- `public Viewport getViewport ()`
- `public void setProjectionChanged()`

4. GvrActivity Activity dasar yang mudah diintegrasikan dengan perangkat Google VR.

- `public GvrView getGvrView ()`
- `public void onBackPressed ()`
- `public void onCardboardTrigger ()`
- `public void setGvrView (GvrView gvrView)`
- `public void setGvrView (GvrView gvrView, boolean enableVrModeFallbacks)`

5. GvrView View yang mendukung *rendering* VR.

6. `HeadTransform Class` yang mendefinisikan perubahan posisi kepala pengguna terhadap dunia VR.

- `public void getEulerAngles (float[] eulerAngles, int offset)`
- `public void getForwardVector (float[] forward, int offset)`
- `public void getHeadView (float[] headView, int offset)`
- `public void getQuaternion (float[] quaternion, int offset)`
- `public void getRightVector (float[] right, int offset)`
- `public void getTranslation (float[] translation, int offset)`
- `public void getUpVector (float[] up, int offset)`

7. `Viewport Class` yang mendefinisikan *viewport* berbentuk persegi panjang.

Atribut-atribut yang dimiliki kelas ini adalah sebagai berikut:

- `public int height`
- `public int width`
- `public int x`
- `public int y`

Method-method yang dimiliki kelas ini adalah sebagai berikut:

- `public void getAsArray (int[] array, int offset)`
- `public void setGLScissor ()`
- `public void setGLViewport ()`
- `public void setViewport (int x, int y, int width, int height)`

[5]

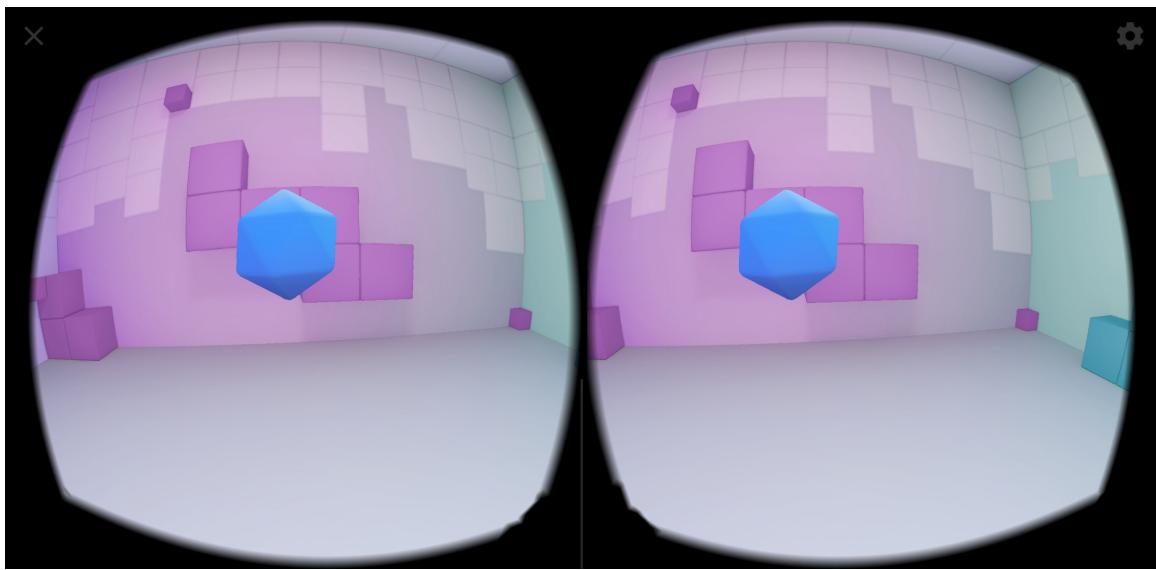
Bagian dari Google VR SDK yang akan digunakan pada penelitian ini adalah aplikasi HelloVR yang akan dijelaskan pada Subbab [2.1.2](#).

2.1.2 Aplikasi HelloVR

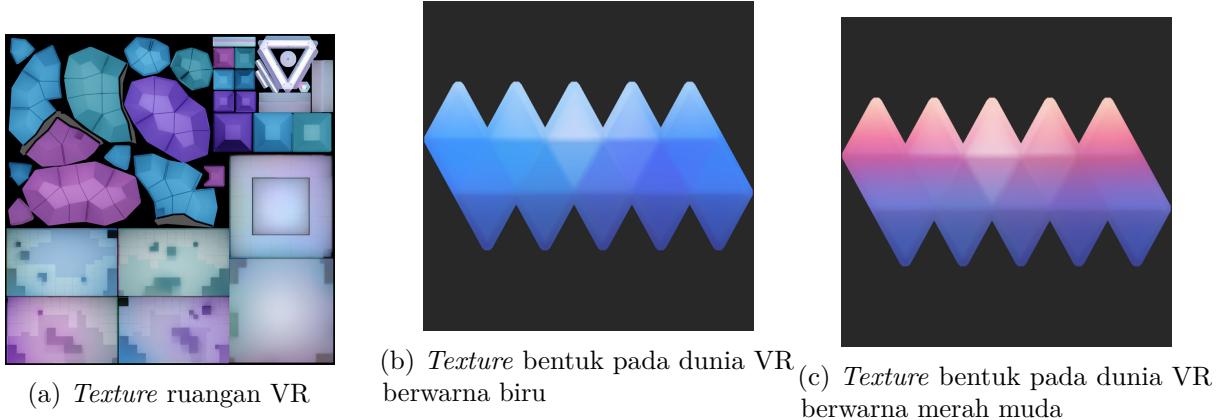
HelloVR, aplikasi yang diperoleh dari Google VR SDK pada direktori `./samples/sdk-hellovr` adalah sebuah aplikasi demo permainan *treasure hunt*, yaitu sejenis permainan mencari bentuk yang mengapung di dunia VR dengan melihat tepat pada bentuk tersebut dan menyalakan pemicu pada Google Cardboard. Setelah kondisi untuk menangkap bentuk yang ada, bentuk tersebut akan menghilang, lalu bentuk yang lain akan muncul di tempat lain. Gambar [2.1](#) menunjukkan tampilan UI permainan *treasure hunt* HelloVR.

Komponen Aplikasi HelloVR

Dunia VR pada aplikasi ini dibuat dari *file Wavefront Object* (OBJ) dengan tekstur *file Portable Network Graphics* (PNG) ([Gambar 2.2a](#)) yang telah dengan sangat tepat dipetakan pada *file* OBJ yang ada sehingga dunia VR terlihat sangat nyata. Bentuk-bentuk yang akan dicari pengguna dibuat dari tiga file OBJ yang merepresentasikan tiga macam bentuk yang akan muncul. Masing-masing file OBJ memiliki dua tekstur yang telah dipetakan pada masing-masing file OBJ dalam file PNG. Satu tekstur ([Gambar 2.2b](#)) digunakan ketika bentuk sedang tidak ada di tengah-tengah titik pengelihatan pengguna, sedangkan satu tekstur yang lain ([Gambar 2.2c](#)) digunakan ketika pengguna sedang melihat bentuk tepat di titik tengah pengelihatan pengguna.



Gambar 2.1: Tampilan UI permainan *treasure hunt* pada aplikasi HelloVR



Gambar 2.2: Gambar-gambar *assets* aplikasi HelloVR

Rancangan kelas Aplikasi HelloVR

Aplikasi HelloVR memiliki empat kelas pada programnya, di antaranya:

1. Texture

Kelas yang memuat tekstur yang akan digunakan. Atribut yang dimiliki kelas ini adalah:

- `int[] textureId` - Atribut yang menyimpan representasi tekstur ruangan yang dapat digunakan dalam kode program.

Method-method yang dimiliki kelas ini di antaranya:

- `public void bind()`

Method ini adalah *method* mengikat tekstur ke `GL_TEXTURE0` dari `GLES20`, yang adalah penyaji (*renderer*) dari dunia VR.

2. TexturedMesh

Kelas ini memuat sebuah bentuk tiga dimensi yang sudah diberi tekstur sehingga terlihat indah dan berwarna. Atribut-atribut yang dimiliki kelas ini adalah:

- `private final FloatBuffer vertices`
Atribut ini adalah atribut dari sudut-sudut ruang tiga dimensi.
- `private final FloatBuffer uv`
Atribut ini adalah atribut dari koordinat tekstur yang digunakan.
- `private final ShortBuffer indices`
Atribut ini adalah atribut indeks sudut-sudut dari permukaan ruang tiga dimensi.
- `private final int positionAttrib`
Atribut ini adalah atribut dari posisi ruang tiga dimensi pada *shader*.
- `private final int uvAttrib`
Atribut ini adalah atribut dari koordinat tekstur pada *shader*. *Shader* adalah program yang mewarnai ruang.

Method yang dimiliki kelas ini adalah: `public void draw()`

Method untuk menggambar ruang dengan tekstur.

3. Util

Kelas yang digunakan untuk menghitung vektor dan sudut yang dibentuk antara mata pengguna dan bentuk yang akan dicari, serta mengatur pengaturan yang tepat untuk *OpenGL*, yang adalah *renderer* yang digunakan untuk menggambar bentuk dan ruangan. Atribut-atribut yang dimiliki kelas ini adalah:

- `private static final boolean HALT_ON_GL_ERROR`
Atribut ini menentukan apakah proses *build* program dihentikan jika ada masalah atau tidak.

Method-method yang dimiliki kelas ini di antaranya:

- `public static void checkGlError(String label)`
Method ini digunakan untuk menjalankan GLES20.

Parameter:

- `String label`
Parameter ini adalah nilai *label* yang akan diteruskan saat galat terjadi.

Return Value: Tidak ada

Exception: Tidak ada

- `public static int compileProgram(String[] vertexCode, String[] fragmentCode)`
Method ini digunakan untuk meng-*compile* program *shader* GLES20.

Parameter:

- `String[] vertexCode`
Parameter ini adalah nilai kumpulan sudut dari program *shader* GLES20
- `String[] fragmentCode`
Parameter ini adalah nilai pecahan-pecahan program *shader* GLES20.

Return Value: *id* dari program GLES20.

Exception: Tidak ada

4. HelloVrActivity

Kelas ini merupakan kelas *activity* Google VR. Berikut adalah diagram kelas untuk memperjelas hubungan antara semua kelas aplikasi HelloVR. Kelas ini akan menggunakan tiga kelas lainnya untuk mendapat ruangan dan bentuk yang akan digambar, serta keadaan (*state*) dari permainan, seperti sedang menatap pada bentuk atau tidak dan bagian ruangan yang sedang dilihat. Atribut-atribut yang dimiliki kelas ini adalah sebagai berikut:

- `private float[] camera`
Atribut *camera* yang direpresentasikan kumpulan bilangan *float*.
- `private int objectProgram`
Atribut dari *id* program *shader*.
- `private int objectPositionParam`
Atribut dari *id* parameter posisi dari objek-objek dalam dunia tiga dimensi.
- `private int objectUvParam`
Atribut dari *id* parameter koordinat tekstur dari objek-objek dunia tiga dimensi.
- `private int objectModelViewProjectionParam`
Atribut dari *id* parameter model view dari objek-objek dunia tiga dimensi.
- `private TexturedMesh room`
Atribut dari ruang tiga dimensi yang telah diberi tekstur.
- `private Texture roomTex`
Atribut dari tekstur yang akan digunakan untuk ruang tiga dimensi.
- `private ArrayList<TexturedMesh> targetObjectMeshes`
Atribut yang memuat kumpulan bentuk tiga dimensi dari target.
- `private ArrayList<Texture> targetObjectNotSelectedTextures`
Atribut yang memuat tekstur dari objek yang sedang tidak dilihat.
- `private ArrayList<Texture> targetObjectSelectedTextures`
Atribut yang memuat tekstur dari objek yang sedang dilihat.

Method-method yang dimiliki kelas ini adalah:

- `public void initializeGvrView()`
Method yang digunakan untuk menginisialisasi pemandangan VR. **Parameter:** Tidak ada
Return Value: Tidak ada
Exception: Tidak ada
- `public void onSurfaceCreated(EGLConfig config)`
Parameter:
 - `EGLConfig config`
Konfigurasi dari OpenGL yang akan digunakan. **Return Value:** Tidak ada **Exception:** Tidak ada

2.2 Google StreetView API

Google StreetView API adalah API yang disediakan Google untuk mendapatkan pemandangan sesuai masukan pengguna melalui *HTTP request* [2]. Ada dua jenis *StreetView API* yang disediakan Google, yaitu *static* dan *dynamic* [2]. *StreetView API* yang statis akan menampilkan pemandangan yang tetap tanpa pergerakan pada pemandangannya, sedangkan yang dinamis menampilkan pemandangan yang berubah-ubah seperti *video*. *StreetView API* yang digunakan pada penelitian ini adalah *Static StreetView API*.



Gambar 2.3: Tampilan UI Google Cloud saat mengakses API Key (API Key disamarkan)

2.2.1 API Key

Agar dapat menggunakan *StreetView API* (dan *Directions API* pada Subbab 2.3), ada *API key* yang harus diperoleh pada Google Cloud Platform Console dengan memasukkan nomor kartu kredit. Gambar 2.3 menunjukkan tampilan *Google Cloud* setelah mendapatkan API key [6]. *API key* yang diberikan terdiri atas dua puluh dan delapan belas karakter alfanumerik (bisa huruf kapital dan huruf kecil) yang dihubungkan dengan tanda "-". *API key* yang telah diperoleh akan digunakan sebagai salah satu parameter masukan agar Google API dapat diakses.

2.2.2 Penggunaan StreetView API

Secara umum, API diakses menggunakan URL Web sebagai berikut:

`https://maps.googleapis.com/maps/api/streetview?parameters`

[2] "Parameters" pada URL Web adalah atribut-atribut dengan parameter yang diterima *StreetView*. Sintaks parameter tersebut adalah:

$$X = Y$$

X adalah atribut dari *StreetView*, sedangkan Y adalah nilainya, dan nilai tersebut harus sesuai dengan tipe dan rentang nilai masing-masing atribut [2]. Untuk atribut kedua dan seterusnya yang akan dimasukkan dalam parameter (jika ada), dapat diteruskan dengan tanda "&", lalu diikuti dengan pola seperti rumus di atas. Saat mengakses *StreetView API*, ada dua kemungkinan hasil yang diperoleh, yaitu berhasil dan gagal [2]. Pemanggilan *API* yang berhasil akan menghasilkan gambar pemandangan dari lokasi sesuai masukan pengguna, sementara pemanggilan yang gagal menghasilkan sebuah gambar dengan penjelasan bahwa gambar tidak tersedia.

2.2.3 Atribut Parameter StreetView API

Untuk menampilkan pemandangan yang sesuai keinginan pengguna, beberapa parameter masukan harus ditentukan. Ada dua jenis parameter masukan, di antaranya parameter wajib dan parameter opsional [2]. Pengaksesan atau pemanggilan *StreetView API* yang berhasil akan mengembalikan sebuah gambar pemandangan dari lokasi sesuai parameter masukan.

Parameter Wajib

Parameter wajib adalah parameter yang harus dimasukkan oleh pengguna dan jika tidak dimasukkan akan mengakibatkan pemanggilan yang gagal [2]. Beberapa parameter wajib pada *StreetView API* adalah:

- *size*

Ukuran dari gambar yang dihasilkan, dalam *pixel*. Format parameter adalah:

banyak pixel secara horizontal × banyak pixel secara vertikal

- *key*
API key yang dijelaskan pada Subbab 2.2.1 [2].

- *location* atau *pano* (salah satu)

Lokasi dari pemandangan yang ingin ditampilkan [2]. *location* menerima dua jenis parameter garis lintang dan garis bujur (*longitude* dan *latitude*) atau *String* nama lokasi, sementara *pano* menerima *panorama id* dari lokasi atau panorama [2].

Parameter Opsional

Selain parameter wajib, ada parameter opsional, yaitu parameter yang tidak perlu diisi agar pengaksesan *API* berhasil dan biasanya atribut parameter tersebut sudah memiliki nilai bawaan (*default*) [2]. Ada beberapa parameter opsional yang dapat digunakan sebagai parameter untuk mengubah pengaturan dari pemandangan yang diambil:

- *signature*
- *heading*
- *fov (field of view)*
- *pitch*
- *radius*
- *source* [2].

Tabel 2.1 menjelaskan semua parameter opsional dari *StreetView API*.

Tabel 2.1: Atribut-Atribut Opsional *StreetView API*

Nama Atribut/Tipe/Rentang Nilai	Penjelasan
<i>signature/String (alfabetik)/-</i>	Atribut untuk memastikan bahwa <i>request</i> dikirim dengan <i>API key</i> sesuai jenis <i>signature</i> yang diatur pemilik <i>API key</i> .
<i>heading/integer/</i> $0 \leq x \leq 360$	menyatakan arah pandangan secara horizontal, nilai atribut menyatakan sudut yang dibentuk dari arah utara dengan arah pandang yang diinginkan (sudut yang dibentuk dari arah berlawanan jarum jam)
<i>fov (field of view)/integer/</i> $10 \leq x \leq 120$ (yang berdampak)	menyatakan seberapa perbesaran pemandangan (nilai dalam satuan derajat)
<i>pitch/integer/</i> $-90 \leq x \leq 90$	menyatakan pandangan pengguna secara vertikal, satuan nilai dalam derajat.
<i>radius/integer/</i> $x > 0$	menyatakan jarak dalam meter, yang adalah titik pengambilan pemandangan
<i>source/String (alfabetik)/</i> " <i>default</i> " atau " <i>outdoor</i> "	menyatakan pemandangan, <i>default</i> atau <i>outdoor</i>

Hasil Pemanggilan *StreetView API*

Pemanggilan *StreetView* yang berhasil akan menghasilkan sebuah gambar dari pemandangan sesuai lokasi [2]. Gambar 2.4 memperlihatkan pemanggilan *StreetView API* yang berhasil dengan parameter tertentu.



Gambar 2.4: Pemanggilan *StreetView API* yang berhasil, dengan parameter *size*=600x300 dan *location*=-6.8746537,107.6046282 dan *API key* yang sah (*API key* tidak disebutkan)

2.3 Google Directions API

Google *Directions API* adalah layanan berbasis *HTTP/HTTPS* dari Google yang membantu mencari dan menghitung arah dari satu tempat ke tempat yang lain (sumber) [3]. Ada beberapa *mode* dari arah yang dapat dicari seperti *driving*, *transit*, *walking*, dan *cycling* [3]. Pengaksesan *Directions API* sangat mirip dengan *StreetView API*, yaitu membutuhkan *API Key*, seperti yang dijelaskan pada Subbab 2.2.1, sebagai salah satu atribut wajib, juga memiliki atribut wajib dan opsional yang dapat diatur lewat parameter [3].

2.3.1 Penggunaan *Directions API*

Sintaks untuk mengaksesnya pun mirip dengan *StreetView API* dengan URL sebagai berikut:

`https://maps.googleapis.com/maps/api/directions/filetype?parameter`

[3] Bagian "filetype" pada URL diganti dengan tipe *file* keluaran atau hasil (xml atau json). Bagian "parameter" diganti dengan parameter-parameter dari *Directions API*. *Directions API* juga memiliki dua jenis parameter seperti *StreetView API*: wajib dan opsional [3]. Tabel 2.2 menyebutkan serta menjelaskan mengenai atribut-atribut parameter dari *Directions API*.

2.3.2 Hasil Pemanggilan *Directions API*

Hal yang dihasilkan oleh pemanggilan *Directions API* adalah *script Javascript Notation Object* (JSON) yang menyatakan arah sesuai lokasi asal dan tujuan, serta mode perjalanan yang adalah masukan pengguna [3]. Selain arah, *script JSON* yang dikembalikan juga mengandung informasi mengenai jarak jalan yang ditempuh serta waktu tempuh perjalanan [3]. Listing 2.1 menunjukkan contoh *script JSON* dengan masukan lokasi asal "UNPAR" dan lokasi tujuan "Rumah Sakit Santo Borromeus". Pemanggilan yang gagal akan mengembalikan *script JSON* yang menyatakan jalan di antara dua lokasi masukan tidak dapat ditemukan.

Listing 2.1: Hasil Pemanggilan *Directions API* yang Berhasil

Tabel 2.2: Atribut-Atribut Opsional *Directions API*

Nama Atribut/Tipe Parameter(Wajib atau Opsional)/Tipe	Penjelasan	Rentang Nilai <i>Valid</i> (yang berdampak)
origin/Wajib/String (alfabetik)	Atribut yang menyatakan lokasi asal yang dimasukkan pengguna.	String lokasi yang <i>sah</i>
destination/Wajib/String	Atribut yang menyatakan lokasi tujuan yang dimasukkan pengguna	String lokasi yang <i>valid</i>
key/Wajib/String	<i>API Key</i> pengguna yang membuat pengguna dapat mengakses <i>API</i>	String <i>API key</i> yang <i>valid</i> (Subbab 2.2.1)
mode/Opsional/String	menyatakan <i>mode</i> perjalanan yang akan ditempuh.	" <i>driving</i> ", " <i>walking</i> ", " <i>bicycling</i> ", atau " <i>transit</i> "
waypoints/Opsional/integer	menyatakan lokasi yang ingin ditempuh dalam perjalanan	String lokasi yang <i>valid</i>
alternatives/Opsional/bit (boolean)	menyatakan apakah rute yang disediakan <i>Directions API</i> menyediakan beberapa pilihan rute	<i>true</i> atau <i>false</i>
avoid/Opsional/String (alfabetik)	menyatakan jenis-jenis jalan yang harus dihindari, seperti jalan tol, jembatan, dan lain-lain	" <i>tolls</i> ", " <i>highways</i> ", " <i>ferries</i> , dan/atau " <i>indoor</i> (gunakan " " jika ada beberapa)
language/Opsional/String (alfabetik)	bahasa yang digunakan untuk menyajikan rute perjalanan	Bahasa yang didukung Google
units/Opsional/String (alfabetik)	jenis satuan yang akan digunakan	" <i>metrics</i> " atau " <i>imperial</i> "

```
{
  "geocoded_waypoints" : [
    {
      "geocoder_status" : "OK",
      "place_id" : "ChIJbYmcEu7maC4RRijB2oKhHLA",
      "types" : [ "establishment", "point_of_interest", "university" ]
    },
    {
      "geocoder_status" : "OK",
      "place_id" : "ChIJU8k7DIHmaC4RQ2mUo1ERm1k",
      "types" : [ "establishment", "hospital", "point_of_interest" ]
    }
  ], "routes" : [
    {
      "bounds" : {
        "northeast" : {
          "lat" : -6.8746719,
          "lng" : 107.6137497
        },
        "southwest" : {
          "lat" : -6.893777099999999,
          "lng" : 107.6034922
        }
      },
      "copyrights" : "Map data 2020",
      "legs" : [
        {
          "distance" : {
            "text" : "1.1 km",
            "value" : 1100
          },
          "duration" : {
            "text" : "2 min",
            "value" : 120
          },
          "end_address" : "Jl. Pahlawan No. 12, Kecamatan Ciputat, Kota Tangerang Selatan, Banten 15311, Indonesia",
          "end_location" : {
            "lat" : -6.8746719,
            "lng" : 107.6137497
          },
          "estimated_time" : 120,
          "html_instructions" : "Jl. Pahlawan No. 12, Kecamatan Ciputat, Kota Tangerang Selatan, Banten 15311, Indonesia",
          "maneuver" : "Turn right at Jl. Pahlawan",
          "start_address" : "Jl. Raya Ciputat No. 10, Kecamatan Ciputat, Kota Tangerang Selatan, Banten 15311, Indonesia",
          "start_location" : {
            "lat" : -6.893777099999999,
            "lng" : 107.6034922
          },
          "travel_mode" : "DRIVING"
        }
      ],
      "summary" : {
        "distance" : {
          "text" : "1.1 km",
          "value" : 1100
        },
        "duration" : {
          "text" : "2 min",
          "value" : 120
        }
      }
    }
  ]
}
```

```
"legs" : [
  {
    "distance" : {
      "text" : "3.0 km",
      "value" : 3042
    },
    "duration" : {
      "text" : "10 mins",
      "value" : 614
    },
    "end_address" : "Jl. Ir. H. Juanda No.100, Lebakgede . . .",
    "end_location" : {
      "lat" : -6.89377099999999,
      "lng" : 107.613021
    },
    "start_address" : "Jl. Ciumbuleuit No.94, Hegarmanah . . .",
    "start_location" : {
      "lat" : -6.8746719,
      "lng" : 107.6046127
    },
    "steps" : [
      {
        "distance" : {
          "text" : "1.0 km",
          "value" : 1008
        },
        "duration" : {
          "text" : "4 mins",
          "value" : 251
        },
        "end_location" : {
          "lat" : -6.8833328,
          "lng" : 107.6049108
        },
        "html_instructions" : "Head \u003cb\u003esouth\u003c . . .",
        "polyline" : {
          "points" : "tuh@yowoSdAP|AL'Cb@dAHHBvC'@l@ . . ."
        },
        "start_location" : {
          "lat" : -6.8746719,
          "lng" : 107.6046127
        },
        "travel_mode" : "DRIVING"
      },
      ...
    ],
    "status" : "OK"
  }
]
```

2.4 Motion Sensor

2.4.1 Deskripsi Motion Sensor

Motion sensor adalah sensor pada *smartphone* yang mendeteksi pergerakan gawai *smartphone* [7]. Pergerakan yang dapat dideteksi termasuk saat gawai dimiringkan, digoyangkan, diayunkan, atau diputar (sumber). Beberapa contoh *motion sensor* pada *smartphone* adalah:

- *accelerometer*

Sensor yang mendeteksi gerakan *smartphone* terhadap sumbu *x*, *y*, dan *z*, termasuk gaya gravitasi terhadap masing-masing sumbu [7].

- *gravity sensor*

Sensor yang mendeteksi gaya gravitasi terhadap sumbu *x*, *y*, dan *z* [7].

- *gyroscope*

Sensor yang mendeteksi putaran gawai terhadap sumbu *x*, *y*, dan *z* [7].

- *linear acceleration sensor*

Sensor yang mendeteksi pergerakan linier, terhadap sumbu *x*, *y*, dan *z* tanpa gaya gravitasi pada masing-masing sumbu [7].

- *rotation vector sensor*

Sensor yang mendeteksi vektor putaran pada gawai terhadap sumbu *x*, *y*, dan *z* [7].

- *step counter*

Sensor yang menghitung jumlah langkah saat berjalan yang pengguna gawai ambil [7].

- *step detector*

Sensor yang men-trigger sebuah *event* saat pengguna mengambil langkah saat berjalan [7].

Pada penelitian ini, *motion sensor* yang akan digunakan adalah *step detector*. Sintaks untuk menggunakan sensor ini tertera pada Listing 2.2.

Listing 2.2: Sintaks menggunakan sensor *step detector*

```
private SensorManager sensorManager;
private Sensor sensor;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
sensor = sensorManager.getDefaultSensor(Sensor.TYPE_STEP_DETECTOR);
```

2.4.2 Deskripsi Step Detector

Step detector adalah sensor yang digunakan untuk mendeteksi langkah kaki pengguna. Sensor ini dapat memicu suatu *event* setiap kali langkah kaki pengguna diambil. Nilai yang dikembalikan adalah 1.0 dan *timestamp* dari saat langkah kaki diambil pengguna (nilai 1.0 menunjukkan bahwa ada langkah yang telah diambil). *Permission* yang dibutuhkan untuk mengaktifkan sensor ini adalah `android.permission.ACTIVITY_RECOGNITION`.

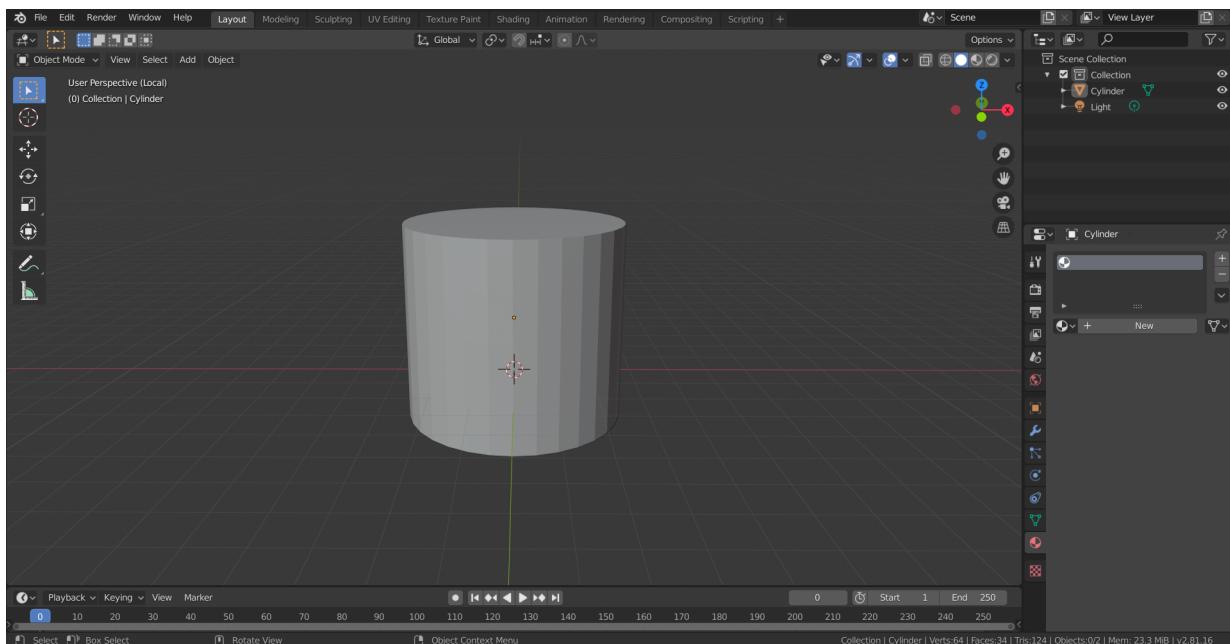
BAB 3

ANALISIS

Pada bab ini dijelaskan mengenai cara memanfaatkan Google VR SDK, *Google StreetView API*, *Google Directions API*, dan cara memanfaatkan sensor *step detector*, termasuk cara mengintegrasikan semua komponen tersebut secara bertahap.

3.1 Membuat Dunia VR

Dari komponen-komponen Google VR SDK, bagian *assets*-lah yang dapat dimodifikasi untuk menampilkan pemandangan VR yang ingin ditampilkan. Dunia VR terdiri atas dua *file*, yaitu *file* OBJ sebagai bentuk ruangan dan *file* PNG sebagai tekstur ruangan *file* OBJ. Dunia yang ingin ditampilkan pada dunia VR adalah dunia dengan bentuk silinder agar terlihat seperti dunia nyata, terutama bagian permukaan samping (yang melengkung). Untuk membuat dunia berbentuk silinder, kertas yang digunakan adalah *Blender* versi 2.81. Gambar menunjukkan tampilan *UI Blender* versi 2.81.



Gambar 3.1: Tampilan *UI Blender* Blender versi 2.81

Bentuk silinder dipilih sebagai bentuk dari dunia VR karena permukaan samping dapat menampilkan pemandangan seperti di dunia nyata. Permukaan samping berbentuk persegi panjang, dan sifat melingkar dari permukaan dapat membuat menampilkan pemandangan di sekitar.



Gambar 3.2: Pemanggilan *StreetView API* yang berhasil, dengan parameter *size*=600x300, *location*=6.8746537,107.6046282, dan *API key* yang sah (*API key* tidak disebutkan), dengan atribut *heading* yang berbeda-beda

3.2 Menampilkan *StreetView API* pada Dunia VR

Android menyediakan kelas untuk menampilkan langsung pada *fragment* yang disebut *StreetViewPanoramaFragment* (sumber). Namun, hal ini tidak dapat dimanfaatkan karena tidak dapat dimanfaatkan karena *fragment* tidak dapat ditampilkan dalam pemandangan dunia VR. Alasan inilah yang membuat *StreetView API* harus diakses melalui HTTP/HTTPS menggunakan URL.

Dengan menggunakan HTTP/HTTPS URL, gambar yang dihasilkan merupakan gambar persegi panjang saat menghadap satu arah sehingga tidak dapat secara langsung ditampilkan pada dunia VR. Untuk melakukannya, gambar dari empat arah harus digabungkan untuk menghasilkan pemandangan seperti dunia nyata. Hal yang harus dilakukan untuk mencapai hal tersebut adalah menggabungkan gambar-gambar dari atribut *heading* dari empat arah, dengan satu arah yang berlawanan dengan arah yang lain, lalu dua arah lain yang tegak lurus dengan arah pertama. Dengan kata lain, selisih setiap dua nilai *heading* yang berurutan bernilai 90 (misalnya *heading* dengan nilai 0, 90, 180 dan 270). Gambar 3.2 memperlihatkan empat gambar *StreetView* dengan berbagai macam *StreetView* dengan syarat tersebut.

Setelah mendapatkan gambar *StreetView* dari semua arah, gambar-gambar tersebut digabungkan sehingga membentuk pemandangan seperti ada di lokasi tersebut. Gambar 3.3 menunjukkan hasil penggabungan empat gambar dengan deskripsi di atas.

Setelah mendapatkan gambar *StreetView* yang sudah digabungkan tersebut, gambar tersebut dapat dimanfaatkan sebagai tekstur untuk ruang pada file OBJ yang berbentuk silinder sehingga pemandangan *StreetView* dapat ditampilkan pada dunia VR.

3.3 Integrasi dengan *Directions API*

Untuk mendapatkan rute perjalanan, *Directions API* dapat dimanfaatkan. File yang diperoleh lewat *Directions API* adalah file JSON yang memiliki *key* dan *value*. Ada beberapa atribut (*key*) dengan nilai (*value*) yang ada pada file JSON dari hasil pemanggilan *Directions API* yang dapat



Gambar 3.3: Contoh hasil penggabungan empat gambar *StreetView*

dimanfaatkan.

3.3.1 Menentukan Atribut yang Bermanfaat

File JSON yang dihasilkan memiliki beberapa tingkat *key* dan *value*. Pada tingkat pertama, ada tiga *key*: *geocoded_waypoints*, *status* dan *routes*. *Key* yang dapat digunakan adalah *routes* yang menunjukkan jalan yang akan ditempuh. Pada tingkat berikutnya, *key routes* memiliki *value* seperti *bounds*, *copyrights*, dan *legs*. Jika melihat bagian *legs* yang menampung atribut-atribut jalan yang ditempuh, ada *distance*, *duration*, *end_location*, *html_instructions*, *maneuver*, *polyline*, *start_location*, dan *travel_mode*. Dari beberapa atribut dari *legs*, yang dapat digunakan adalah *end_location* dan *start_location*, yang menunjuk kepada posisi garis lintang dan garis bujur titik ujung dari jalan yang sedang ditempuh.

3.3.2 Cara Memanfaatkan Atribut

Setelah memperoleh nilai *end_location* dan *start_location*, jalur tempuh pada jalan yang sedang ditempuh dapat ditentukan lewat posisi garis lintang dan garis bujur kedua titik ujung jalan. Cara untuk membentuk jalan secara matematis adalah menggunakan selisih antara garis lintang dan garis bujur satu titik ujung jalan ke titik ujung jalan lain.

3.4 Cara Memanfaatkan *Step Detector* Sensor

Untuk membuat animasi atau perubahan pemandangan sesuai rute tempuh yang sudah diperoleh, harus ada perubahan gambar sesuai langkah kaki yang diambil pengguna. Jadi, setiap kali sensor mendapat rangsang, *event* yang dipicu agar terjadi adalah gambar berubah seperti yang dijelaskan pada Subbab 3.3.2, tetapi perubahan gambar *StreetView* yang sesuai *Directions API* harus berubah dengan tahap yang benar agar animasi pemandangan terlihat mulus.

DAFTAR REFERENSI

- [1] (2018) *Quickstart for Google VR SDK for Android* / Google Developers.
- [2] (2020) *Developer Guide / Street View Static API* / Google Developers.
- [3] (2020) *Developer Guide / Directions API* / Google Developers.
- [4] (2018) *Choose Your Development Environment* / Google VR / Google Developers.
- [5] (2020) *com.google.vr.sdk.base* / Google VR / Google Developers.
- [6] (2020) *Get an API Key and Signature* / Street View Static API.
- [7] (2020) *Motion sensors* / Android Developers.