PERMAINAN TRADISIONAL BALAP KELERENG BERBASIS VIRTUAL REALITY MENGGUNAKAN ALGORITMA COMPLEMENTARY FILTER

TRADITIONAL MARBLES RACING GAME BASED ON VIRTUAL REALITY USING COMPLEMENTARY FILTER ALGORITHM

Raka Arya Pangestu[1],Tito Waluyo Purboyo[2],Anton Siswo Raharjo Ansori[3]

[1,2,3] Universitas Telkom, Bandung
[1]rakaarya@student.telkomuniversity.ac.id, [2]titowaluyo@telkomuniversity.ac.id, [3]raharjo@telkomuniversity.ac.id

Abstrak

Permainan tradisional balap kelereng membutuhkan detail gerakan seperti gerakan menahan sendok pada mulut pemain agar kelereng yang berada pada sendok tidak terjatuh hingga garis akhir. Dikarenakan permainan ini sudah jarang dimainkan di perkotaan maupun di daerah maka, tidak sedikit orang yang malas melakukannya bahkan tidak tahu permainan tersebut. Sedikitnya minat bermain balap kelereng ini menjadikan budaya permainan tradisional ini sudah jarang ditemui. Dalam hal ini pada penelitian yang dilakukan bertujuan untuk mengembangkan atau membuat permainan simulasi balap kelereng secara virtual menggunakan *complementary filter*.Perkembangan software dalam *virtual reality* dengan cepat menjadi populer, terutama untuk aplikasi permainan. Mendukung realisasi pembuatan permainan yang didukung oleh Unity 3D. Unity 3D adalah mesin *game cross platform*. Alat ini terintegrasi untuk membuat permainan, membangun arsitektur, dan simulasi. Dengan menggunakan algoritma *complementary filter* memungkinkan gerakan dari getaran sendok bisa terdeteksi dan dapat membaca apabila kelereng jatuh ke permukaan.Untuk mengestimasi getaran sendok dan kelereng, permainan ini membutuhkan alat sensor berupa sensor *accelerometer* dan *gyroscope*. sensor *accelerometer* dapat mengukur kecepatan dari sebuah objek sedangkan, sensor *gyroscope* dapat mengukur dan mempertahankan orientasi dari sebuah objek. Untuk mendapatkan hasil yang akurat, membutuhkan *Low Pass Filter* (LPF), dan *High Pass Filter* (HPF).

Kata Kunci : *Complementary Filter*, Permainan Balap Kelereng, Permainan tradisional, *Unity 3D, Virtual Reality,*

Abstract

The traditional game of marbles racing requires detailed movements such as the movement of holding the spoon in the player's mouth so that the marbles on the spoon do not fall to the finish line. Because this game is rarely played in urban or regional areas, not a few people are lazy to do it and don't even know the game. The lack of interest in playing marbles racing makes this traditional game culture rarely found. In this case, the research carried out aims to develop or create a virtual marble racing simulation game using a Complementary Filter. Software developments in virtual reality are rapidly becoming popular, especially for gaming applications. Supports the realization of game creation supported by Unity3D. Unity3D is a cross platform game engine. These tools are integrated for creating games, building architecture, and simulations. By using the Complementary Filter algorithm, it is possible to detect the movement of the spoon vibration and can read if the marbles fall to the surface. To optimize the vibration of spoons and marbles, this game requires sensors in the form of accelerometer and gyroscope sensors. The accelerometer sensor can measure the speed of an object whereas, the gyroscope sensor can measure and maintain the orientation of an object. To get accurate results, you need a Low Pass Filter (LPF), and a High Pass Filter (HPF).In this case, the author make a virtual reality game of marbles with a real location and can be played in locations that have enough space around the campus, such as in the FTE area of the Telkom University

campus.Keywords : Complementary Filter, Unity3D, Virtual Reality, Traditional Game, Marble Racing Game.

1. Pendahuluan

Pada era teknologi yang semakin maju, industri pengembangan permainan berlomba - lomba menciptakan aneka ragam jenis permainan. Dari kategori Racing, First Person Shooter (FPS), action, Role-Playing Game (RPG), dan lain - lain. Dalam hal ini, agar masyarakat tidak merasa bosan dan jenuh pada kegiatan sehari - hari. Tidak sedikit juga permainan yang diadaptasi dari permainan tradisional, salah satunya yang penulis teliti adalah permainan balap kelereng.

Permainan balap kelereng merupakan permainan tradisional Indonesia yang dimainkan oleh dua orang atau lebih, biasanya dimainkan saat lomba 17 Agustus. Cara melakukan permainan ini adalah dengan menggunakan sendok yang diberi sebuah kelereng, lalu melaju ke garis finish. Kelereng tidak boleh terjatuh, dan harus bertahan hingga akhir. Permainan ini, bisa juga dimainkan secara berkelompok dengan mengumpulkan kelereng terbanyak, dengan waktu yang ditentukan [1].

Penulis berencana menerapkan permainan ini pada salah satu platform yang sedang tren di masa kini adalah Virtual Reality atau yang biasa disingkat VR adalah sebuah teknologi yang membuat pengguna atau user dapat berinteraksi dengan lingkungan yang ada dalam dunia maya dan disimulasikan oleh komputer yang mampu membangkitkan suasana 3 dimensi

sehingga membuat pemakai seolah-olah berada pada lingkungan tersebut secara nyata. Pada hal ini pemakai bisa merasakan lomba 17 Agustus secara virtual tanpa harus menunggu lomba tersebut dilakukan pada tanggal yang sudah ditetapkan [4].

Untuk melakukan penelitian terkait permainan balap kelereng, penulis berencana menggunakan algoritma complementary filter dalam pembuatan permainan balap kelereng berbasis virtual reality. complementary filter adalah algoritma filter yang terdiri dari 2 buah filter yaitu Low Pass Filter dan High Pass Filter [2]. Output dari sensor accelerometer akan diberi Low Pass Filter, sedangkan High Pass Filter untuk output dari sensor gyroscop[3].

2. Kajian Pustaka

1. Permainan Tradsional

Perkembangan zaman sudah sangat melesat dengan cepat, teknologi juga menjadi semakin kompleks,. Di zaman serba teknologi pada sekarang ini, permainan tradisional sudah sangat jarang ditemui, khususnya di perkotaan. Budaya permainan tradisional harusnya bisa dipertahankan dan dimainkan hingga saat ini, tetapi dengan adanya permainan virtual membuat permainan tradisional ditinggalkan dan dilupakan.

2. Permainan Balap Kelereng

Permainan balap kelereng sering kita temui pada saat lomba 17 Agustus, dengan memperingati hari kemerdekaan Indonesia. Membutuhkan dua orang atau lebih di setiap permainannya, dan juga dapat berkelompok. Sendok dan kelereng sebagai tumpuan dari permainan ini, melaju hingga garis finish tanpa menjatuhkan kelereng menyentuh dasar permukaan. Pemenang ditentukan oleh, yang melaju ke garis finish dahulu dan mampu mempertahankan kelereng yang berada pada sendok.

3. Virtual Reality

Virtual reality merupakan gabungan dari kata bahasa Inggris yaitu "virtual" dan "reality". "Virtual" mengacu pada kedekatan, dan "realitas" mengacu pada hal-hal nyata yang kita alami sebagai manusia. Oleh karena itu, dapat disimpulkan bahwa virtual reality memiliki arti “mendekati kenyataan”. Bagi orang yang mempunyai jarak yang jauh, Virtual Reality bisa menjadi solusi lain untuk bertemu satu sama lain[10].

Virtual reality dapat ditemukan dengan berbagai cara, seperti pada headset, treadmill, dan sarung tangan khusus. Saat menggunakan realitas maya, otak manusia dan kelima organ Indera akan menghasilkan halusinasi. Oleh karena itu, banyak orang menggunakan teknologi ini untuk bermain permainan. Menggunakan Virtual Reality bisa membuat orang merasa berada di tempat itu, sehingga bermain akan lebih menyenangkan..

4. Unity 3D

Unity3D adalah aplikasi untuk mengembangkan game multi-platform yang mudah digunakan. Unity sangat bagus dan penuh dengan aplikasi profesional. Editor Unity terdiri dari antarmuka pengguna yang sederhana. Editor dibuat setelah ribuan jam kerja keras, dan butuh ribuan jam untuk menjadikannya alat nomor satu di antara editor game.

5. Accelerometer

Sensor Accelerometer merupakan alat yang digunakan untuk mengukur suatu kecepatan pada benda atau objek. Sensor ini dapat mengukur dua jenis kecepatan, dinamis dan statis.Kecepatan dengan yang sedang bergerak dikatakan dengan dinamis, sedangkan kecepatan dengan mengukur terhadap gravitasi bumi yaitu statis. Penelitian ini sangat membutuhkan sensor ini karena mendeteksi dalam penelitian balap kelereng.

6. Gyroscope

Gyroscope adalah perangkat untuk mengukur atau mempertahankan orientasi, dengan prinsip ketetapan sudut, alat ini bekerja sama dengan *accelerometer*. Mekanismenya adalah sebuah roda berputar dengan piringan di dalamnya yang tetap stabil.Alat ini sering digunakan pada robot atau *drone* serta alat-alat canggih lainnya. Sensor ini dapat untuk menentukan orientasi gerak dengan bertumpu pada roda atau cakram yang berotasi dengan cepat pada sumbu. *Gyroscope* sensor sendiri memiliki fungsi untuk mendeteksi gerakan sesuai gravitasi, atau dengan *customized structure* lain mendeteksi gerakan pengguna.

7. Complementary Filter

Complementary Filter merupakan algoritma yang berfungsi sebagai optimasi sensor *accelerometer* dan *gyroscope*. Serta menghitung perhitungan tindakan yang akan terjadi selanjutnya. Algoritma ini dapat meningkatkan akurasi sensor sehingga mendeteksi getaran sendok serta mendeteksi kelereng yang jatuh ke permukaan. Filter ini membutuhkan *low pass filter* (LPF) *dan high pass filter* (HPF) untuk menjadi lebih akurat.

8. ESP-32

ESP32 adalah *microcontroller* yang dikenalkan oleh Ekspresif Sistem merupakan penerus dari *microcontroller* ESP8266. Pada *microcontroller* ini sudah tersedia modul *Wi-Fi* dalam *chip* sehingga sangat mendukung untuk membuat sistem aplikasi *Internet of Things*. terlihat pada gambar di atas merupakan *pin out* dari ESP32. Pin tersebut dapat dijadikan *input* atau *output* untuk menyalakan LCD, lampu, bahkan untuk menggerakkan motor DC.

9. Modul Sensor MPU-6050

Sensor MPU 6050 merupakan sensor keseimbangan yang sudah dilengkapi dengan sensor *Accelerometer* dan *Gyroscope* yang mampu membaca sudut kemiringan objek dengan cakupan tiga dimensi. Sensor ini mempunyai

pin yang akan dihubungkan ke *microcontroller* ESP32.

10. Firebase (*cloud storage*)

Firebase adalah suatu layanan dari Google untuk memberikan kemudahan bahkan mempermudah para developer aplikasi dalam mengembangkan aplikasinya. Firebase alias *BaaS* (*Backend as a Service*) merupakan solusi yang ditawarkan oleh *Google* untuk mempercepat pekerjaan developer. Dengan menggunakan *Firebase*, *apps* developer bisa fokus dalam mengembangkan aplikasi tanpa memberikan *effort* yang besar untuk urusan *backend*.

11. Arduino IDE

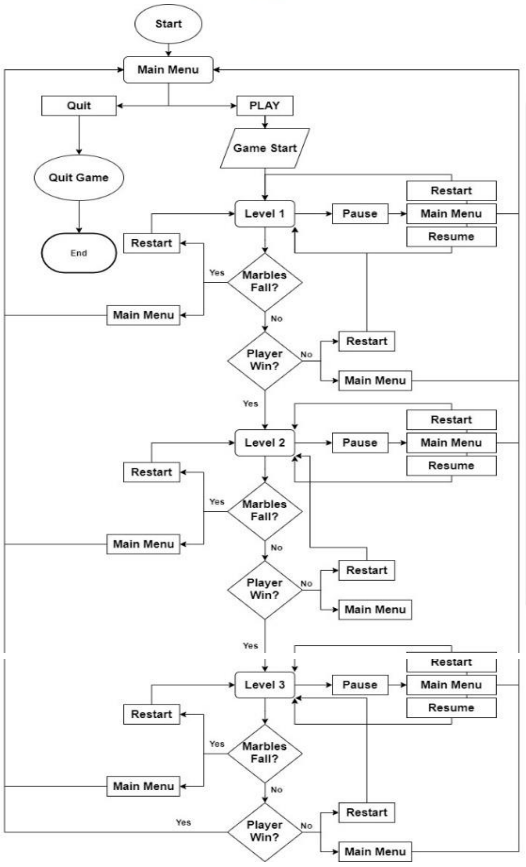
Arduino IDE *(Integrated Development Environment)* adalah *software* yang di gunakan untuk memprogram di *arduino*, dengan kata lain *Arduino* IDE sebagai media untuk memprogram *board Arduino*. *Arduino IDE* bisa di-*download* secara gratis di *website* resmi Arduino IDE. *Arduino IDE* ini berguna sebagai *text* editor untuk membuat, mengedit, dan juga mevalidasi kode program. bisa juga digunakan untuk meng-*upload* ke *board Arduino*. Kode program yang digunakan pada Arduino disebut dengan istilah *Arduino "sketch"* atau disebut *juga source code arduino*, dengan ekstensi *file source code* "*.ino*".

3. Perancangan Sistem

3.1 Desain Sistem

Dalam bab ini menjelaskan tentang bagaimana urutan sistem yang dibuat agar dapat bermain di permainan tradisional balap kelereng. Yang bertujuan membuat perancangan sistem permainan, pengumpulan data, serta pengolahan data.

3.1.1 *Flowchart* Permainan

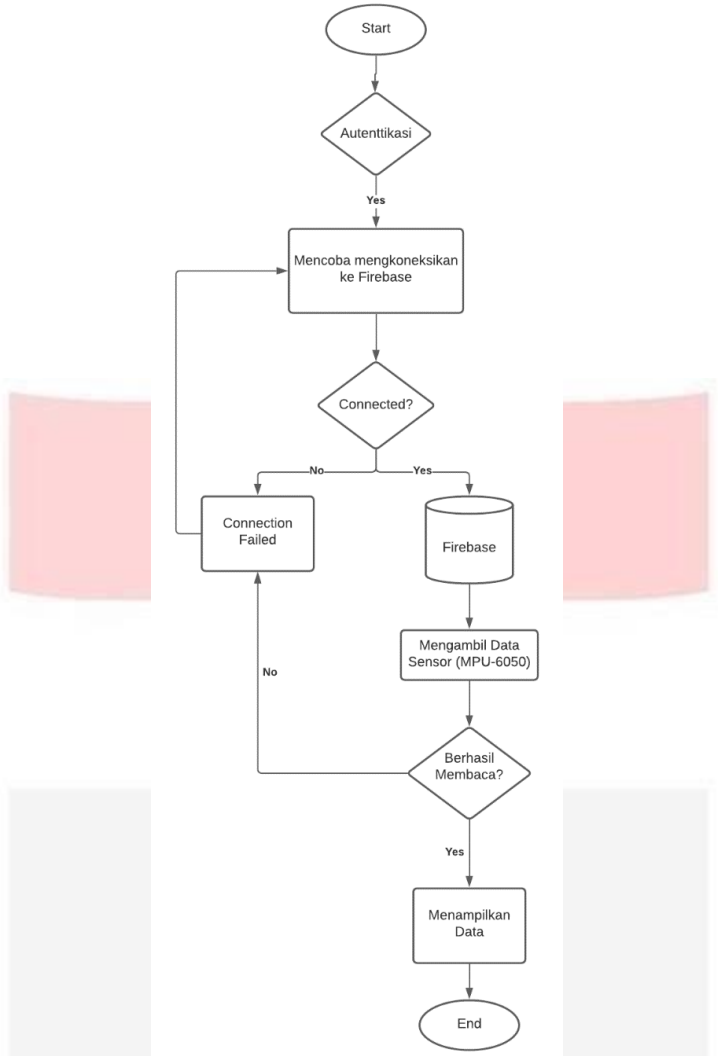


Gambar 3.1 flowchart permainan

Dalam aplikasi permainan tradisional balap kelereng mempunyai alur yang akan dilakukan. Pemain akan memilih menu start untuk memulai permainan. Permainan akan dimulai dari level 1 (mudah). Jika pemain berhasil mengalahkan lawan bermain, pemain akan melanjutkan permainan ke level 2 (sedang). Setelah pemain berhasil melalui level 2, permainan akan berlanjut ke level 3 yaitu level terakhir yang harus dimainkan. Apabila pemain kalah dengan lawan bermain pemain bisa memilih untuk lanjut dengan menekan tombol *restart* atau memberhentikan permainan

3.1.2 Diagram Sensor MPU-6050 Pada ESP32

Gambar di bawah ini menunjukkan alur dari proses implementasi sensor pada ESP32 yang berisikan *source code complementary filter*.

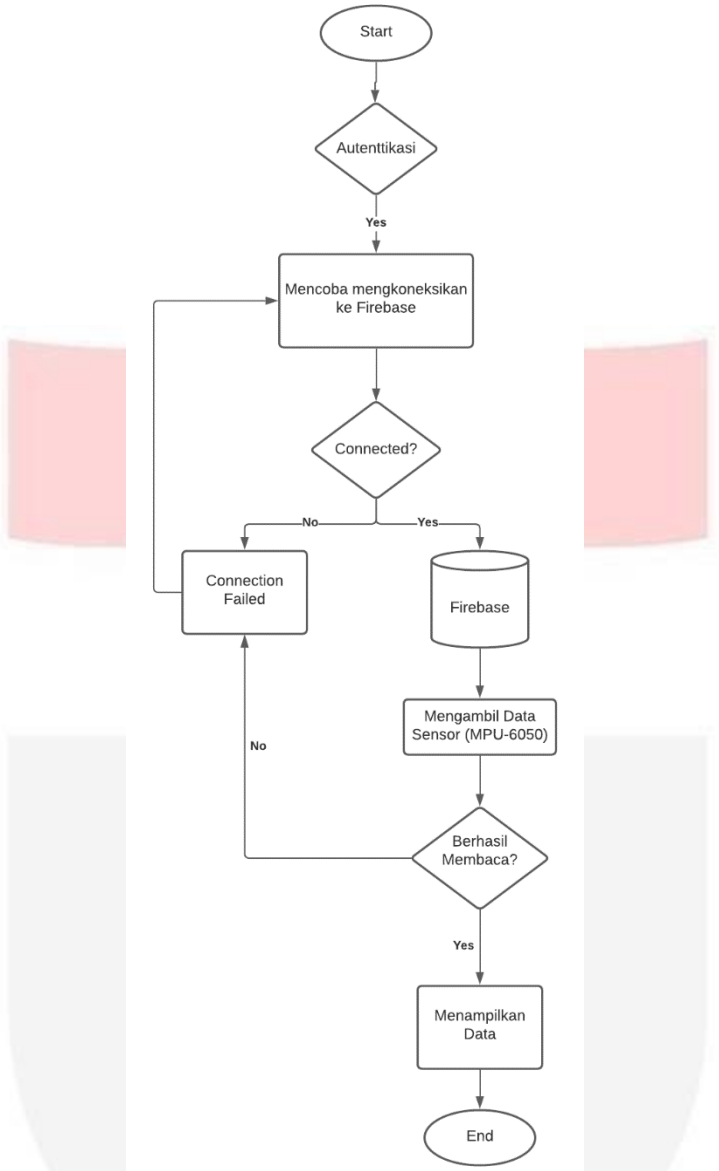


Gambar 3.2 *flowchart* sensor pada ESP32.

Sensor MPU6050 digunakan sebagai pengukur nilai sudut dari suatu benda. Data dari sensor yang masih berbentuk *raw* akan dikonversikan ke *complementary*. Setelah dikonversikan, nilai tersebut akan menjadi data yang akan digunakan untuk menggerakkan objek pada *gam*

3.1.3 Diagram Sensor MPU-6050 Pada ESP32

Gambar di bawah ini menunjukkan alur dari proses *firebase* menyimpan data sensor yang sudah dikirimkan melalui aplikasi *Arduino IDE* dan akan diambil datanya lewat aplikasi *Unity 3D*.



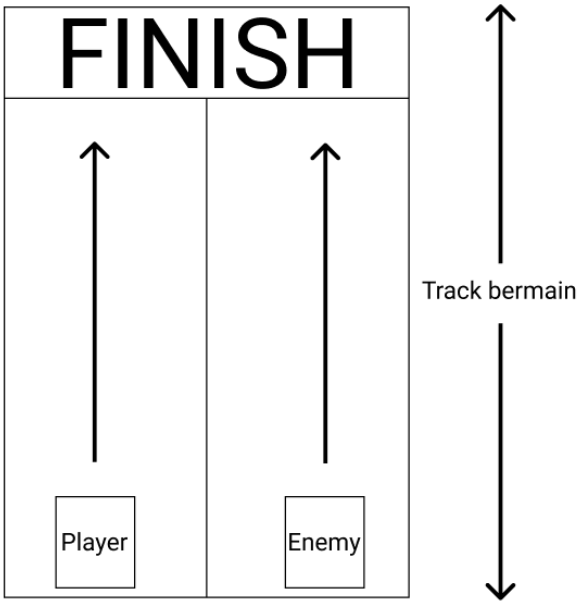
Gambar 3.3 *flowchart Firebase* untuk menyimpan data

3.2 Pemodelan permainan simulasi sepeda balap pada aplikasi *unity* untuk *player*.

Perancangan sistem adalah bagian dari metode pengembangan perangkat lunak yang digunakan untuk memberikan gambaran rinci tentang *game*, termasuk *gameplay*, perhitungan algoritma *complementary filter*, desain *Environment*, desain kontrol *game*, dan desain antarmuka.

3.2.1 Skenario Permainan

Pada menu awal bermain, pemain akan memulai permainan dengan memilih tombol *start*. Lalu, pemain akan berusaha mencapai garis *finish* bersama *Enemy* (*bot*) yang menjadi lawan bermain dengan jarak yang telah ditentukan. Setelah pemain berhasil mencapai garis *finish* lebuh dahulu, pemain akan melanjutkan ke tingkatan permainan selanjutnya dengan jarak yang lebih jauh dari sebelumnya. Permainan akan berjalan seperti aturan yang ditentukan hingga mencapai tingkatan terakhir. Apabila pemain gagal dalam mencapai garis *finish* atau kalah dalam melawan *enemy (bot)* akan dinyatakan kalah, begitu juga dengan menjatuhkan kelereng yang berada di atas sendok.



Gambar 3.4 Skenario *Track* dari permainan

3.2.1 Skenario Permainan

Konfigurasi ini berfungsi untuk menghubungkan ESP32 dengan MPU-6050 agar dapat menyediakan data yang didapatkan oleh sensor dan dikirim ke server.

Tabel 3.1 Konfigurasi Pin ESP32.

Nama Pin (MPU6050)	**Terhubung ke (ESP32)**
VCC	*VCC*
GPO 22	*SDA*
GPO 21	*SCL*
GND	*GND*

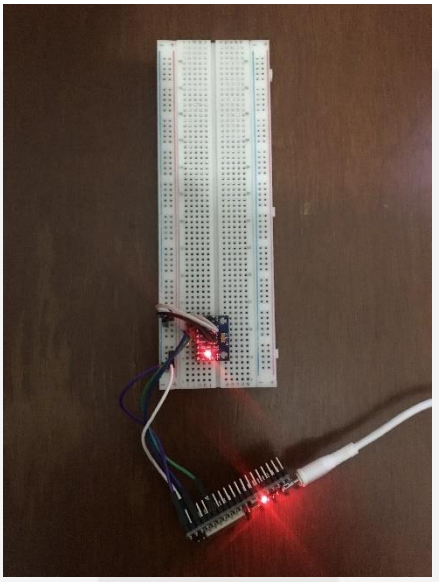
Pada Tabel 3.1 menunjukkan bahwa kedua *pin* saling terhubung dan akan membaca alamat bus dari ESP32. Setiap pin dari masing komponen akan terhubung sehingga dapat mengirim data dan membaca alama

3.3 Perancangan *Environment*

Game ini dapat dijalankan dengan menggunakan *control* yang sudah tersedia pada *Oculus Go*. Model dan tata letak *control* sebagai berikut.

Gambar 3.5 Controller Oculus Go.

3.2.5. Perancangan Perangkat Keras Kontrol Sendok

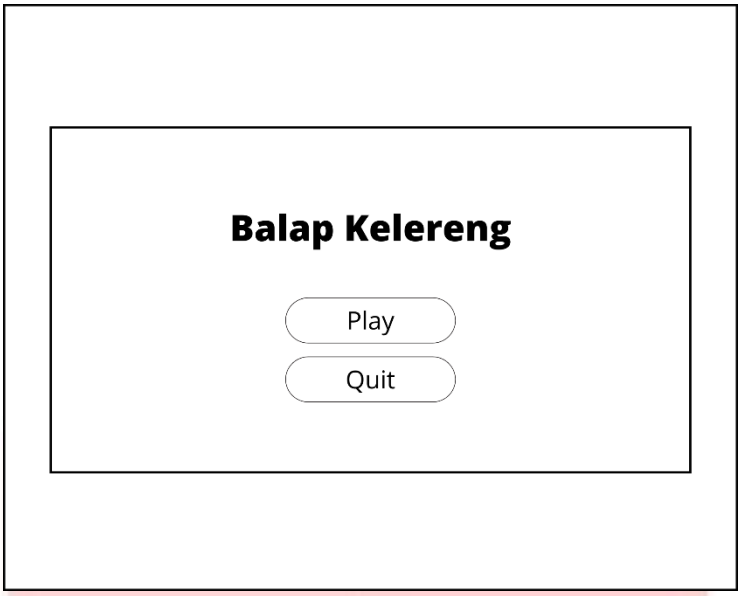


Gambar 3.6 Perangkat keras untuk menggerakkan sendok dalam *game*.

Pada Gambar 3.6 dapat dilihat perangkat keras untuk menggerakkan objek sendok dalam *game* menggunakan sensor MPU-6050 yang dihubungkan dengan ESP32.

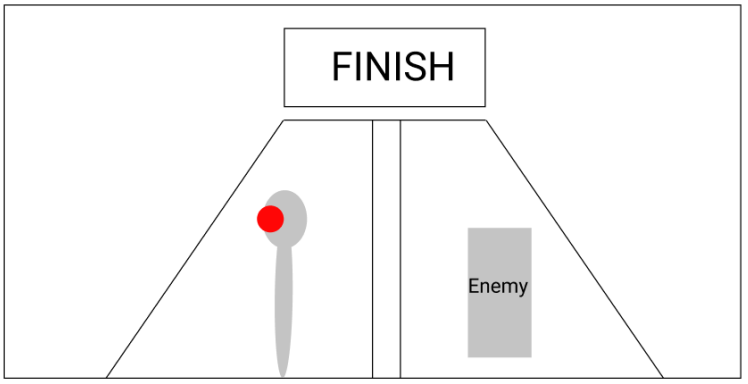
3.2.5. Storyboard dan Interface

Rancangan *storyboard* dan *interface* adalah gambaran dari tampilan dari keseluruhan pada *game*.mulai dari saat menu awal sampai akhir dari permainan. Rancangan *storyboard* dan *interface* permainan sebagai berikut.



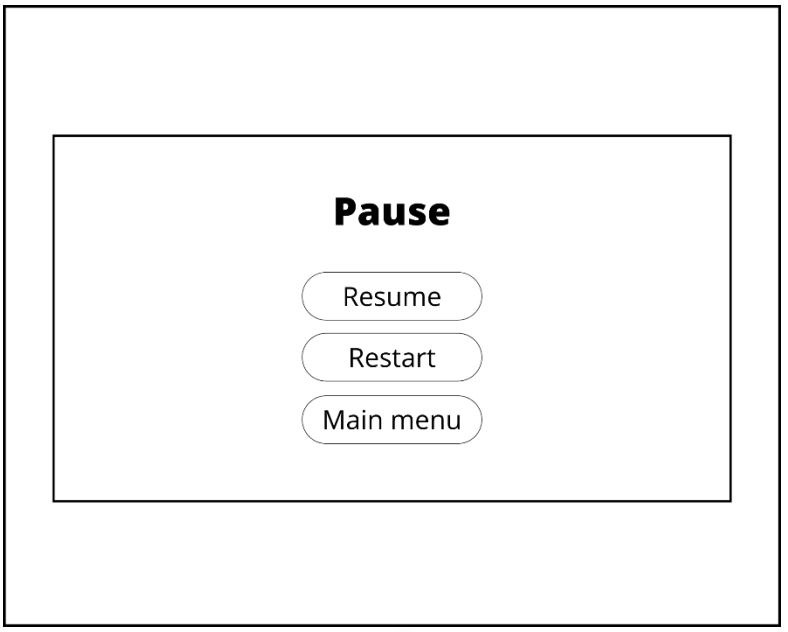
Gambar 3.7 Rancangan tampilan antar muka *game saat menu awal*.

Pada Gambar 3.7 diperlihatkan rancangan tampilan antar muka saat pertama kali masuk ke dalam permainan dan akan memilih 2 *menu* yang sudah disediakan. *Button play* untuk bermain dan *button quit* untuk keluar dari permainan.



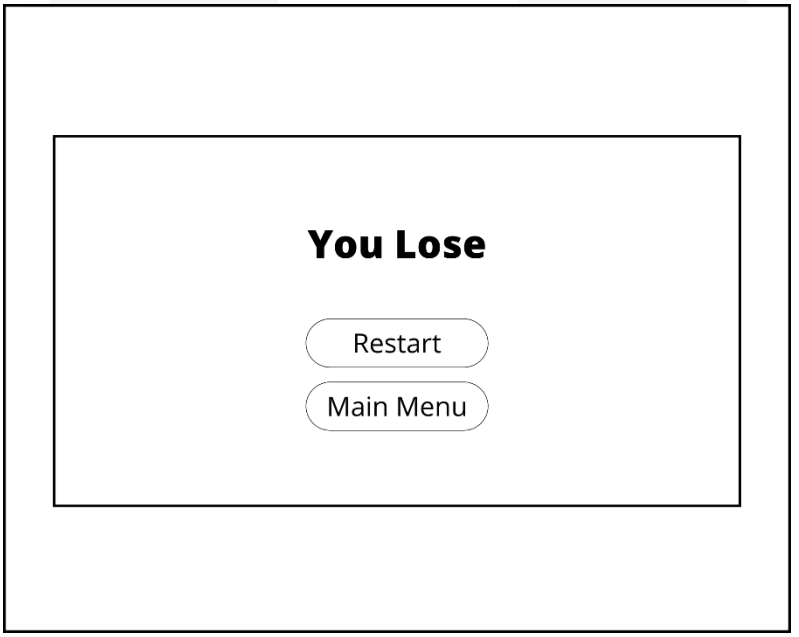
Gambar 3.8 Rancangan tampilan antar muka *gameplay*.

Pada Gambar 3.8 diperlihatkan rancangan tampilan antar muka saat bermain. Diperlihatkan dari *point of view* pemain ketika sendok sejajar dengan mulut pemain.



Gambar 3.9 Rancangan tampilan antar muka *game* ketika menu *pause*.

Pada Gambar 3.9 diperlihatkan rancangan tampilan antar muka saat *pause*. Dalam menu *pause* terdapat 3 *button* yaitu, *resume* untuk melanjutkan permainan, *restart* untuk mengulang permainan, dan *main menu* untuk kembali ke menu awal



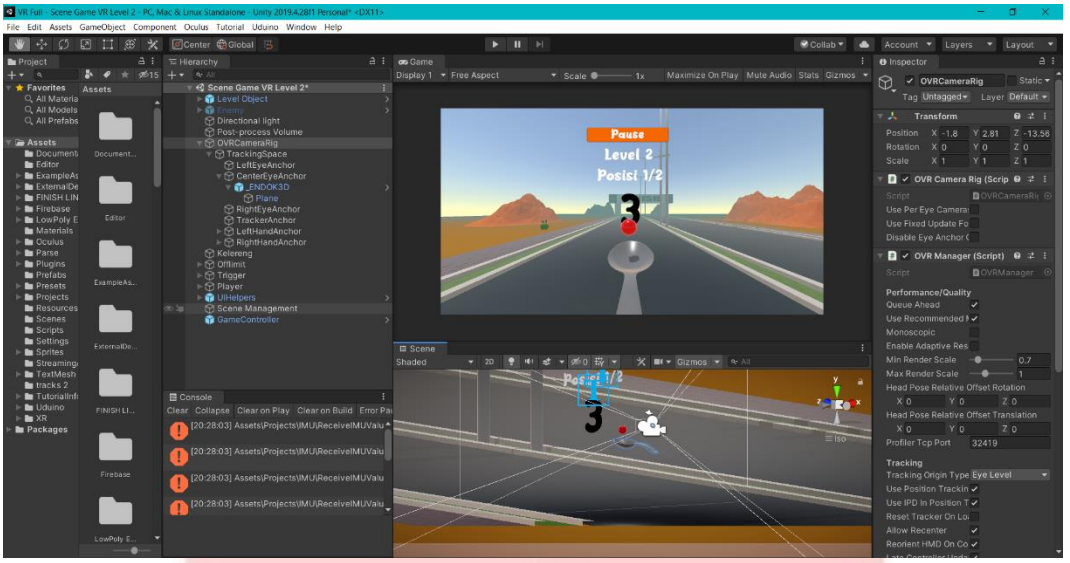
Gambar 3.10 Rancangan tampilan antar muka game ketika kalah saat bermain.

Pada Gambar 3.10 diperlihatkan rancangan tamplian antar muka saat pemain kalah dalam permainan. Terdapat 2 *button* yaitu *restart* dan *main menu*

4. Implementasi dan Pengujian Sistem

4.1 Implementasi *game*

Implementasi ini sebagai proses penerapan dari seluruh rancangan yang telah dibuat menjadi suatu aplikasi/*game* agar bisa dijalankan. Selain itu, tujuan dari implementasi yaitu untuk mengetahui keberhasilan dari suatu rancangan yang telah dibuat sedemikian rupa hingga menjadi suatu aplikasi.



Gambar 4.1 *Project game* yang dibuat di aplikasi *Unity3D.*

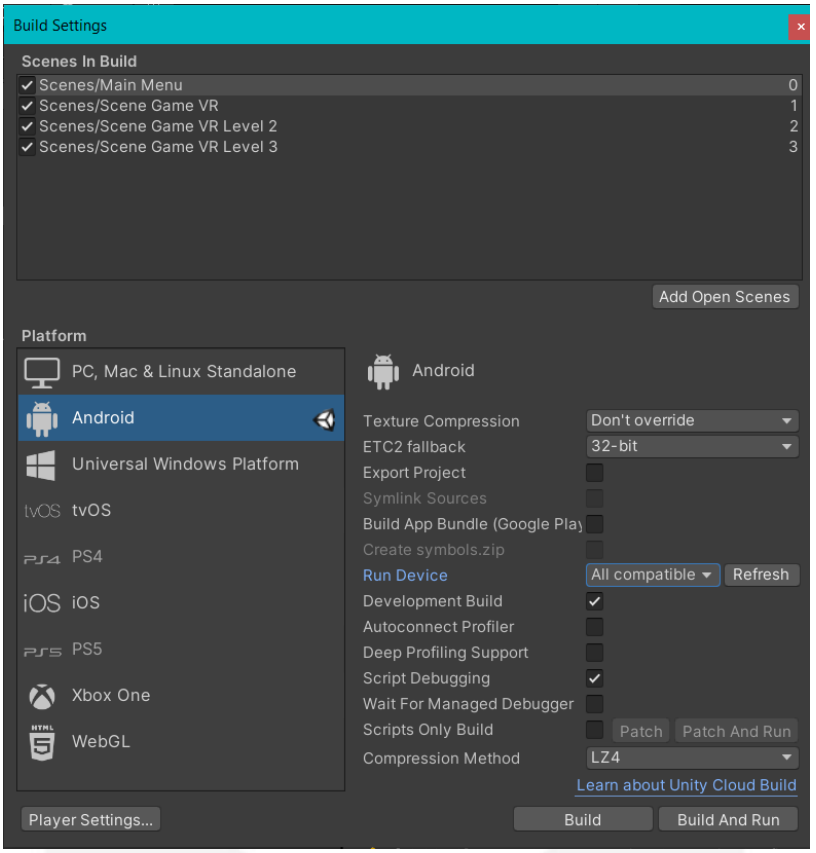
Pada gambar terdapat beberapa *window* dengan fungsi yang berbeda-beda. *Hierarchy* mempunyai fungsi untuk memodifikasi atau menambah aset pada *game*. *Inspector* adalah *window* untuk memodifikasi ataupun memanipulasi objek yang ada pada game. Pada lembar kerja memiliki dua *window* yang berbeda yaitu *window game* dan *scene*. Pada *window game* berguna untuk melihat tampilan dari perspektif *user* yang akan dimainkan nantinya, sedangkan pada *window scene* berguna untuk mengatur dan menempatkan objek pada *game* sehingga bisa diatur sesuai yang diinginkan bagi developer. *Library project* berfungsi sebagai penyimpanan *file* pada *game* yang berupa aset, *script*, *object* ataupun *library* yang ingin ditambahkan ke dalam *project*.

4.1.1. Implementasi *Complementary filter* pada sensor MPU-6050

Complementary filter akan mengurangi adanya getaran (*noise)* pada keluaran *accelerometer* dan *gyroscope*. Sehingga mendapatkan keakuratan lebih pada saat sensor akan digerakkan. Pengukuran sensor meliputi *pitch*, *roll*, dan *yaw*. 3 komponen itu merupakan 3 dimensi dari objek ketika digerakkan. *Pitch* akan menggerakkan bagian depan dan ekor sendok, *yaw* akan menggerakkan bagian badan sendok dari sisi ke sisi, sedangkan *roll* adalah gerakan melingkar yang searah dengan jarum jam ataupun berlawanan dengan arah jarum jam jika bagian badan sendok saat bergerak maju.

4.1.2. Implementasi Build Unity Menjadi Aplikasi

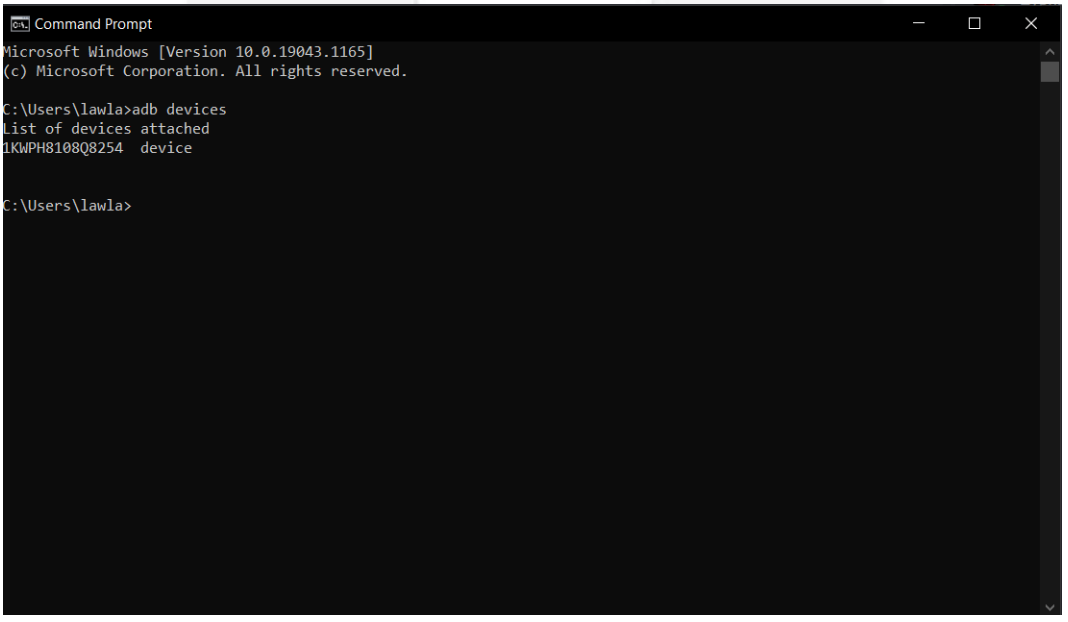
Implementasi ini bertujuan untuk seluruh rancangan akan dijadikan *file* dengan format *file.apk.* Sehingga pengembang tidak perlu menyerahkan *file* mentahan ke publik, cukup dengan *file.apk*. karena *Oculus Go* berbasis *android* maka aplikasi akan dijadikan dengan *format* sesuai dengan *Operation Sistem* di *Oculus Go.*



Gambar 4.2 Tampilan pengatuan Build.

4.1.3. Implementasi Sideload Apk Pada Oculus Go

Implementasi *Sideload Apk* ini dirancang untuk *install file.apk* yang dibuat di *Unity* di *Oculus Go*. Cara *install file.apk* adalah dengan menggunakan *Android Software Development Kit* yaitu *driver* ADB atau *Android Debug Bridge*. ADB adalah program *client-server* untuk pengembangan aplikasi *Android*. ADB biasanya digunakan untuk mengelola perangkat Android. Implementasi *Sideload* ini menggunakan *command prompt* atau artinya kita biasanya mendengarkan cmd. Pastikan *driver* ADB sudah terinstal di PC, dan akan memindahkan *file.apk* yang dibuat di *Unity* ke C: adb.



Gambar 4.3 Pemberitahuan *device Oculus Go* sudah terkoneksi.

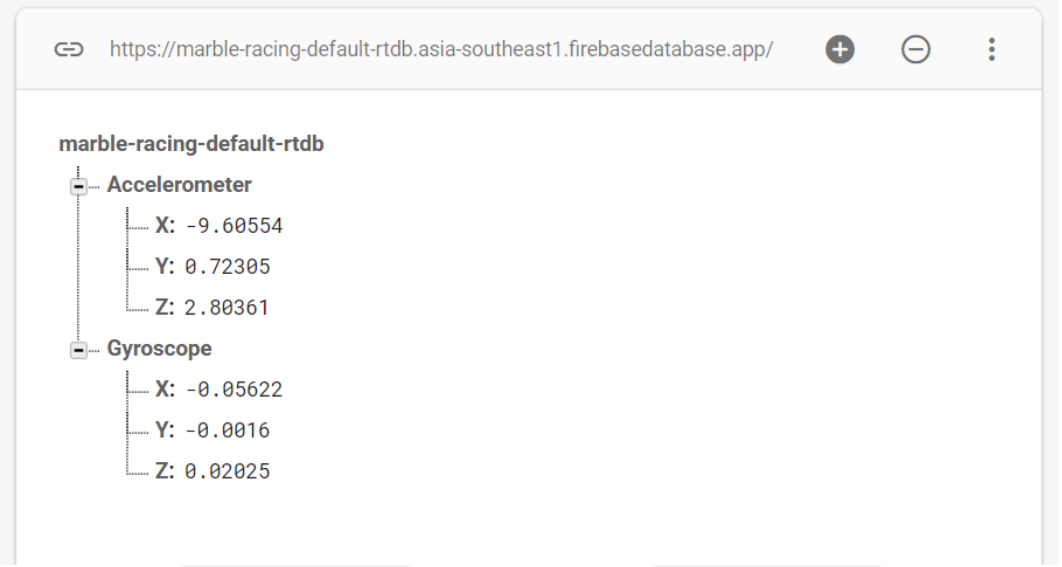
4.2. Pengujian

Setelah permainan dibuat menggunakan *Unity 3D* tahap selanjutnya memasukkan permainan ke dalam *Oculus Go* agar dapat dimainkan secara virtual di ruang nyata. Untuk tidak gagal dalam mengimplementasikannya dibutuhkan evaluasi terhadap program yang sudah dibuat meliputi seluruh fungsi dari fitur dan juga pengujian pada *gameplay*.

4.2.1. Pengujian mengirim data ke *Database* Firebase

Pengujian ini bertujuan untuk mengirimkan data yang sudah terbaca oleh sensor MPU-6050 dan dikirimkan ke *database firebase.* Terdapat beberapa Langkah yang harus dilakukan agar data yang sudah terbaca oleh sensor dapat di kirim kan ke *firebase*. Pertama. menghubungkan *microcontroller* ESP32 ke *Wi-Fi*. *Microcontroller* terlebih dahulu harus terhubung dengan *Wi-Fi* karena dengan menggunakan *script* di bawah ini. Setelah itu, Menghubungkan ke *firebase*. Langkah ini dimulai dengan menambahkan *script firebase* pada *Arduino* IDE Agar terhubung dengan *console* yang sudah dibuat di *firebase* yang terdapat pada *script* Arduino yang sudah dibuat..Lalu, Mengirimkan data ke database firebase. Setelah ESP32 terhubung dengan *Wi-Fi* dan sudah autentikasi dengan *firebase*, selanjutnya mengirimkan data ke *firebase*. Berikut adalah *script*-nya. Terakhir, mengecek data melalui *realtime database firebase.*

Pengujian ini bertujuan untuk mengecek data sensor dari MPU-6050 ke *firebase* dan *Unity3D* Akan membaca data dari *firebase* untuk mengecek data. Data akan dikirimkan secara *realtime* dan masuk ke fitur *realtime database* dari *firebase.*



Gambar 4.4 Data sensor yang telah dikirim ke *firebase.*

Pada gambar 4.4 data sensor sudah berhasil dikirim ke *firebase,* Data tersebut meliputi data dari *accelerometer* dan *gyroscope*. *Accelerometer* mempunyai sumbu X, Y, dan Z. begitu juga dengan *gyroscope* mempunyai sumbu X, Y, dan Z. data ini dikirim secara *realtime* dengan kecepatan 1 detik.

4.2.3. Pengujian Fungsional

Pada tahapan ini penulis menampilkan hasil dari pengujian yang telah dilakukan, di mana pengujian tersebut meliputi pengujian *controller player*, pengujian fungsionalitas dan pengujian *user*. Pengujian controller pada player berfungsi untuk mengetahui apakah semua button yang ada pada Oculus Controller yang digunakan untuk bermain dalam game berfungsi sesuai dengan yang diharapkan atau tidak. Tabel Pengujian Controller Player ditunjukkan pada Tabel 4.1

Tabel 4.1 Tabel Pengujian controller.

No	*Controller*	Fungsi	Output	Behasil	Gagal
1	*Trigger*	Memilih *button* menu pada *game*	Memilih *button menu* yang dipilih pemain	√	-
2	*Joystick*	Menggerakkan pemain maju atau	Pemain maju dan mundur	√	-

		mundur			
3	*VR Headset*	Menggerakkan kamera pemain	Kamera bergerak dan sendok bergerak	√	-
4	MPU-6050	Menggerakkan sendok	Sendok bergerak	√	-

Dari Tabel 4.1 Hasil dari pengujian kontrol pada pemain berfungsi dengan baik dan sesuai dengan rancangan sistem yang telah dibuat dengan tingkat persentase 100%

Pengujian *user* dilakukan untuk mengetahui apakah aplikasi yang dibuat sudah berjalan dengan baik atau tidak. Pengujian *user* dilakukan pada mahasiswa Telkom University dengan metode wawancara terhadap *user*. Berikut ini pengujian *user* ditunjukkan pada Tabel 4.2

Tabel 4.2 Pengujian *user*.

No.	Pertanyaan	Penilaian		
		Kurang	Cukup	Baik
1	Apakah *game* dapat dijalankan oleh user?	-	1 Orang	9 Orang
2	Apakah tampilan dari *game* sudah menarik untuk *user*	-	2 Orang	8 Orang
3	Apakah VR *headset* sudah dapat digunakan dalam menggerakkan rotasi dari kamera *user*?	-	2 Orang	8 Orang
4	Apakah pemain puas dengan sistem yang dirancang dalam *game*?	1 Orang	4 Orang	5 Orang
5	Apakah sensor MPU6050 yang menggerakkan sendok sudah bergerak sesuai dengan aturan bermain dalam *game*?	3 Orang	3 Orang	4 Orang
6.	Apakah c*ontroller player* dalam *game* sudah dapat digunakan oleh *user*	1 Orang	5 Orang	4 Orang
Jumlah		5 (8,33%)	17 (28,3%)	38 (63,3%)

Berdasarkan Tabel 4.2 didapatkan hasil dari pengujian *user* yang berjumlah 10 orang. Hasil pengujian sebagai berikut, Kurang sebesar 8,33%, Cukup 28,3%, Baik 63,3%. Oleh karena itu, hasil pengujian terhadap 18 orang responden cenderung menilai *game* dengan indikasi baik sebesar 63,6%.

4.2.4 Pengujian Sensor MPU-6050

Perubahan gerakan dari sensor menghasilkan sudut kemiringan yang akan berbeda. Dalam hal ini *output* dari sumbu X dan sumbu Y menjadi bagian dari pengujian. Parameter dari pengujian ini memakai sudut 0º, 30º, 60º dan 90º. Pengambilan data akan diperlihatkan contoh sensor yang sudah di filter dan yang belum di filter. Dan akan mengambil 5 sampel dengan 2 kali pengujian.

Metode yang dilakukan untuk menguji sensor dengan mencari standar deviasi dan rata-ratanya. Standar deviasi berfungsi untuk menentukan bagaimana persebaran data dalam suatu sampel dan melihat seberapa dekat data-data tersebut dengan *mean* atau rata-rata dari sampel tersebut.

Tabel 4.3 Pengujian pertama *pitch* dengan mengambil 5 sampel.

Pengujian Pitch				
Tanpa Filter	0°	30°	60°	90°
sampel ke-1	0,66	30,81	60,55	90,41
sampel ke-2	0,32	30,46	60,32	90,16
sampel ke-3	0,67	30,56	60.27	90,36
sampel ke-4	0,33	30,62	60,54	90,26
sampel ke-5	0,43	30,68	60,89	90,61
Rata - Rata	0,482	30,626	60,575	90,36
Standar Deviasi	0,173	0,1311	0,2353	0,17
Complementary Filter	0°	30°	60°	90°
sampel ke-1	0,55	30,76	60,59	90,29
sampel ke-2	0,57	30,76	60,59	90,26
sampel ke-3	0,55	30,98	60,61	90,27
sampel ke-4	0,57	30,88	60,61	90,29
sampel ke-5	0,57	30,88	60,61	90,23
Rata - Rata	0,562	30,852	60,602	90,27
Standar Deviasi	0,011	0,0934	0,01095	0,026

Tabel 4.4 Pengujian kedua *pitch* dengan mengambil 5 sampel.

Pengujian Pitch				
Tanpa Filter	0°	30°	60°	90°
sampel ke-1	0,33	31,42	60,85	90,09
sampel ke-2	0,22	30,65	60,76	90,29
sampel ke-3	0,76	30,12	60,69	90,12
sampel ke-4	0,97	31,28	60,03	90,06
sampel ke-5	0,56	30,43	60,31	90,01
Rata - Rata	0,568	30,78	60,528	90,11
Standar Deviasi	0,307	0,5556	0,34615	0,106
Complementary Filter	0°	30°	60°	90°
sampel ke-1	0,43	30,96	60,66	90,17
sampel ke-2	0,42	30,95	60,59	90,17
sampel ke-3	0,42	30,95	60,62	90,15
sampel ke-4	0,43	30,96	60,6	90,17
sampel ke-5	0,43	30,96	60,6	90,16
Rata - Rata	0,426	30,956	60,614	90,16
Standar Deviasi	0,005	0,0055	0,02793	0,009

Tabel 4.5 Pengujian pertama *roll* dengan mengambil 5 sampel.

Pengujian Roll				
Tanpa Filter	0°	30°	60°	90°
sampel ke-1	0,33	30,35	60,87	90,46

Pengujian Roll				
sampel ke-2	0,21	31,21	59,36	90,45
sampel ke-3	0,55	31,71	60,38	90,52
sampel ke-4	0,65	30,55	60,77	90,87
sampel ke-5	0.32	30,74	60,76	90,71
Rata - Rata	0,435	30,912	60,428	90,602
Standar Deviasi	0,200915	0,548197	0,625596	0,182675
Complementary Filter	0°	30°	60°	90°
sampel ke-1	0,24	30,32	60,45	90,32
sampel ke-2	0,22	30,32	60,44	90,33
sampel ke-3	0,24	30,32	60,44	90,31
sampel ke-4	0,24	30,33	60,45	90,32
sampel ke-5	0,21	30,31	60,45	90,34
Rata - Rata	0,23	30,32	60,446	90,324
Standar Deviasi	0,014142	0,007071	0,005477	0,011402

Tabel 4.6 Pengujian kedua *roll* dengan mengambil 5 sampel.

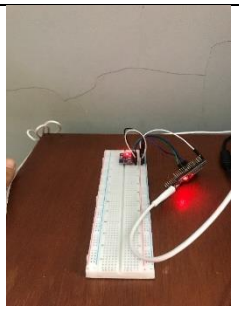
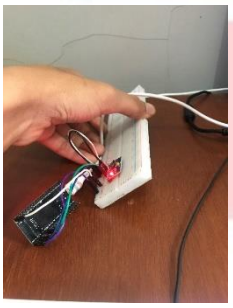
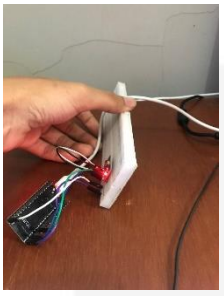
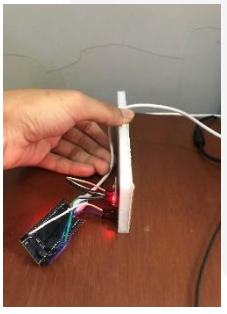
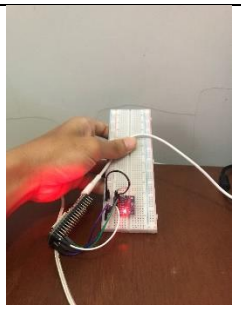
Pengujian Roll				
Tanpa Filter	0°	30°	60°	90°
sampel ke-1	0,23	30,13	60,78	90,54
sampel ke-2	0,86	30,89	60,12	90,74
sampel ke-3	0,85	30,86	60,67	90,78
sampel ke-4	0,78	30,58	61,25	90,21
sampel ke-5	0,13	30,44	60,03	90,56
Rata - Rata	0,57	30,58	60,57	90,566
Standar Deviasi	0,359096	0,314881	0,502643	0,225566
Complementary Filter	0°	30°	60°	90°
sampel ke-1	0,28	30,31	60,32	90,28
sampel ke-2	0,28	30,29	60,32	90,27
sampel ke-3	0,27	30,29	60,31	90,27
sampel ke-4	0,26	30,28	60,31	90,27
sampel ke-5	0,28	30,29	60,32	90,28
Rata - Rata	0,274	30,292	60,316	90,274
Standar Deviasi	0,008944	0,010954	0,005477	0,005477

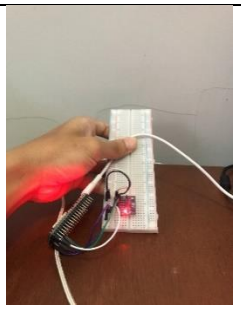
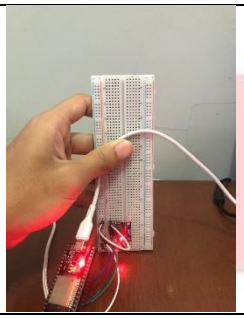
Pada pengujian sensor MPU-6050 menunjukkan bahwa sensor yang menggunakan *Complementary Filter* lebih baik dalam pengukuran sudut dikarenakan nilai dari standar deviasi pengukuran sudutnya lebih mendekati nilai acuan pengukuran sudut dibandingkan dengan standar deviasi pengukuran sudut dari tanpa filter. Dan, tanpa menggunakan filter terdapat getaran (*noise*) yang lebih besar dikarenakan nilai dari standar deviasinya lebih besar dibandingkan dengan standar deviasi dari *Complementary Filter*.

4.2.5 Pengujian Kondisi Pada Sensor MPU6-050

Pengujian ini berfungsi untuk melihat keakuratan dan kesesuaian gerakan sensor yang menjadi kontrol sendok pada *game* di aplikasi Unity 3D. pengujian ini dilakukan dengan sudut yang berbeda beda.

Tabel 4.7 Hasil pengujian kesesuaian sensor dengan sendok

No.	Kondisi Sensor	Kondisi Sendok	Ket.	%
1.			0°	100%
2.			30° (*Roll*)	100%
3.			60° (*Roll*)	100%
4.			90° (*Roll*)	100%
5.			30° (Pitch)	100%

No.	Kondisi Sensor	Kondisi Sendok	Ket.	%
6.			60° (Pitch)	100%
7.			90° (Pitch)	100%

5. Simpulan dan Saran

5.2 Simpulan

Dari pengujian dan analisis yang sudah dilakukan terhadap sistem yang sudah dirancang maka dapat ditarik kesimpulan sebagai berikut:

1. Berdasarkan dari hasil pengujian yang telah dilakukan, data dari sensor MPU-6050 yang telah menggunakan *Complementary Filter*. Dari pengujian yang telah dilakukan, hasil dari pengujian *Complementary Filter* sudah dapat mengurangi getaran (*noise*) pada sensor IMU dan sesuai dengan kondisi sendok dan kondisi sensor dengan hasil 100%.
2. Pada pengujian sensor MPU-6050 diambil 5 sampel data dengan sudut yang berbeda beda. Percobaan dengan menggunakan *Complementary filter*, dengan menggunakan sudut 0º dengan standar deviasinya 0,011. Sedangkan tanpa filter diperoleh nilai standar deviasinya 0,173. Hal ini menunjukkan bahwa *Complementary Filter* lebih baik dalam mengurangi getaran (*noise*) dan melakukan pengukuran sudut pada sensor. Dikarenakan nilai standar deviasi dari *Complementeary Filter* sudut 0º lebih mendekati nilai acuan pengukuran sudut.
3. Pada perangkat keras kontrol objek sendok dalam *game* dalam menggerakkan objek berhasil diimplementasikan ke dalam aplikasi *Unity 3D* dan dapat diimplementasikan ke *device Oculus Go*.

5.3 Saran

Untuk penelitian lebih lanjut, penulis mempunyai beberapa saran terhadap sistem agar nantinya dapat diperbaiki atau ditingkatkan lagi:

1. Penelitian selanjutnya diharapkan sistem dapat meningkatkan akurasi lebih baik lagi dengan algoritma *Complemetary Filter*.
2. Penelitian selanjutnya diharapkan pengiriman data sensor MPU6050 melalui ESP32 ke *database firebase* dapat mengatasi *delay* yang terjadi.

Referensi:

[1] Abas Setiawan, A. S. (2017). Benthix VR: A Virtual Reality Simulation Application to Preserve A Traditional Game. *ComTech*.

[2] Fábio Matoseiro Dinis, A. S. (2017). Development of Virtual Reality Game-Based interfaces for Civil Engineering Education. *FEUP*.

[3] Febriyanto Pratama Putra, H. T. (2012). Pembuatan Game Animasi 3d Role Playing Game Untuk Pendidikan Budaya Dengan Unity3d Dan Bahasa Pemrograman C#. *Teknik Informatika, Fakultas Komunikasi dan Informatika*.

[4] Hikmah Prisia Yudiwinata, P. H. (2014). PERMAINAN TRADISIONAL DALAM BUDAYA PERKEMBANGAN ANAK. *Paradigma*, Volume 2 Nomer 03.

[5] Rangga Septyan Putra, D. Y. (2018). Pemanfaatan Virtual Reality Pada Perancangan Game Fruit Slash Berbasis Android Menggunakan Unity 3D. *Volume IV*.

[6] Salhazan Nasution, A. H. (2019). Pembuatan Plugin Tile-Based Game Pada Unity3D. *IT Journal Research and Development (ITJRD)*.

[7] Siti Yuliani, H. M. (2016). Kolaborasi Kalman Filter dengan Complementary Filter untuk Mengoptimasi Hasil Sensor Gyroscope dan Accelerometer. *LIPI*.

[8] Sung Lae Kim, H. J. (2014). Using Unity 3D to Facilitate Mobile Augmented Using Unity 3D to Facilitate Mobile Augmented. *Department of Digital Media*.

[9] Tariqul Islam, M. S.-U.-M.-E.-H. (2017). Comparison of complementary and Kalman filter-based data fusion for attitude heading reference sistem. *AIP Conference Proceedings*, 020002.

[10] Theunstoppalbes. (2019). Virtual Reality, Teknologi Masa Depan