

**RANCANG BANGUN APLIKASI PUZZLE GAME “MATCH  
SHAPE” BERBASIS LEAP MOTION CONTROLLER**

**SKRIPSI**

**Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana  
Komputer (S.Kom.)**



**Albert Van Otto**

**11110110082**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNIK INFORMASI DAN KOMUNIKASI  
UNIVERSITAS MULTIMEDIA NUSANTARA  
TANGERANG**

**2016**

**HALAMAN PENGESAHAN**

**RANCANG BANGUN APLIKASI PUZZLE GAME “MATCH  
SHAPE” BERBASIS LEAP MOTION CONTROLLER**

Oleh

Nama : Albert Van Otto

NIM : 11110110082

Program Studi : Teknik Informatika

Fakultas : Teknologi Informasi dan Komunikasi

**Tangerang,** \_\_\_\_\_

Ketua Sidang

Dosen Penguji

(.....) (.....)

Dosen Pembimbing

Dennis Gunawan, S.Kom., M.Sc.

Mengetahui,

Ketua Program Studi

Teknik Informatika

Maria Irmina Prasetyowati, S.Kom., M.T.

## PERNYATAAN TIDAK MELAKUKAN PLAGIAT

Dengan ini saya,

Nama : Albert Van Otto

NIM : 11110110082

Program Studi : Teknik Informatika

Fakultas : Teknologi Informasi dan Komunikasi

Menyatakan bahwa skripsi yang berjudul “**Rancang Bangun Aplikasi Puzzle Game ‘Match Shape’ Berbasis Leap Motion Controller**” adalah karya ilmiah pribadi saya, bukan karya ilmiah yang ditulis oleh orang atau lembaga lain, dan semua karya ilmiah orang lain yang dirujuk dalam skripsi ini telah disebutkan sumber kutipannya serta dicantumkan di Daftar Pustaka.

Jika di kemudian hari terbukti ditemukan kecurangan / penyimpangan baik dalam pelaksanaan skripsi maupun dalam penulisan laporan skripsi, saya bersedia menerima konsekuensi dinyatakan tidak lulus untuk mata kuliah Skripsi yang telah saya tempuh.

Tangerang, \_\_\_\_\_

Albert Van Otto

## **RANCANG BANGUN APLIKASI PUZZLE GAME “MATCH SHAPE” BERBASIS LEAP MOTION CONTROLLER**

### **ABSTRAK**

*Video game* adalah salah satu aktivitas waktu luang yang populer, riset mencantumkan *video game* dapat digunakan sebagai alternative pengobatan atau meningkatkan kemampuan kinerja otak. Pada sebuah penelitian ditemukan bahwa memainkan salah satu jenis *video game* berupa *puzzle* dapat meningkatkan kinerja *executive functions*, namun pengujian dilakukan dengan menggunakan *video game* berbasis *touch screen* yang didesain dalam platform *smartphone* sehingga sugesti mengenai *alternative mechanism* lainnya dapat memberikan *input* penelitian selanjutnya. Oleh karena itu, digunakan Leap Motion Controller sebuah alat yang berfungsi sebagai *gesture recognition* dalam antarmuka natural yang dapat menerima input 3D dibandingkan interaksi *touch screen* yang hanya menerima input 2D. Aplikasi telah berhasil dibangun dan diuji menggunakan metode penelitian Jakob Nielsen ke 5 responden dalam 3 kali iterasi. 15 responden dicari berdasarkan kriteria usia 20-30 tahun, dengan latar belakang pendidikan yang berbeda. Hasil analisis data menunjukkan aplikasi dapat digunakan pengguna dengan hasil akhir 77,74% yang termasuk dalam kategori kuat.

Kata kunci: *Puzzle, Game, Gesture recognition, Leap Motion Controller, Unity*

## **DESIGN AND DEVELOPMENT OF PUZZLE GAME APPLICATION “MATCH SHAPE” BASED ON LEAP MOTION CONTROLLER**

### **ABSTRACT**

Video game is one of the most popular pastimes, research quoting video game could be use as a medical alternative or could increase brain performance. One research found that playing a puzzle video game could increase executive functions performance, but the test was done using touch screen based video game which is designed for smartphone leading to suggestion about other alternative mechanism would give another input for the research. Because of that, Leap Motion Controller a device that function as a gesture recognition in natural interface that able to accept 3D input instead of touch screen interaction which only able to accept 2D input was used. The application has been successfully developed and tested using Jakob Nielsen research method to 5 respondents in 3 iterations. The 15 respondents are in the age of 20-30 years old, with different education background. Data analyzation shows that the application was usable to user with the result of 77,74% which is categorized as strong.

Keywords: Puzzle, Game, gesture recognition, Leap Motion Controller, Unity

## KATA PENGANTAR

Puji syukur kepada Tuhan Yang Maha Esa, atas berkat-Nya sehingga penulis dapat menyelesaikan laporan tugas akhir yang berjudul “Rancang Bangun Aplikasi Puzzle Game “Puzzle Match” berbasis Leap Motion Controller.

Penulis ingin mengucapkan terima kasih kepada:

1. Alvin William, yang telah membantu desain rancangan program dan memberikan semangat selama pembangunan aplikasi dan laporan,
2. Thomas Simpson, S.Ds., yang telah membantu rancangan estetika dan pembangunan *level* dan memberikan semangat selama pembangunan aplikasi dan laporan,
3. Dennis Gunawan, S.Kom., M.Sc., selaku Pembimbing Skripsi 1 dan 2 yang memberikan bimbingan dan semangat selama pembuatan aplikasi dan laporan,
4. Seluruh keluarga penulis, yang telah memberikan dukungan dan semangat kepada penulis,
5. Maria Irminda Prasetyowati, S.Kom., M.T., selaku Ketua Program Studi Teknik Informatika yang memberikan arahan dan semangat selama pembuatan aplikasi dan laporan,
6. Segenap dosen, teman penulis, dan penguji aplikasi yang tidak dapat penulis sebutkan satu-persatu

Semoga laporan tugas akhir ini dapat memberikan petunjuk untuk masa depan teknologi informasi yang lebih baik.

Tangerang, 14 November 2016

Albert Van Otto

## DAFTAR ISI

PERNYATAAN TIDAK MELAKUKAN PLAGIAT .....	iii
ABSTRAK .....	iv
ABSTRACT .....	v
KATA PENGANTAR .....	vi
DAFTAR ISI .....	vii
DAFTAR GAMBAR .....	viii
DAFTAR TABEL .....	ix
BAB I PENDAHULUAN .....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	2
1.3 Batasan Masalah .....	2
1.4 Tujuan Penelitian .....	3
1.5 Manfaat Penelitian .....	3
1.6 Sistematika Penulisan .....	3
BAB II TINJAUAN PUSTAKA .....	4
2.1 Antarmuka Natural .....	4
2.2 Leap Motion Controller .....	5
2.3 Struktur Permainan .....	6
2.4 Post Study Usability Questionnaire .....	7
2.5 Likert Scale .....	9
2.6 Pencarian Problem Usability .....	11
BAB III METODOLOGI PENELITIAN DAN PERANCANGAN SISTEM .....	13
3.1 Metode Penelitian .....	13
3.2 Struktur Permainan .....	14
3.3 Penggunaan Aset .....	16
3.4 Perancangan Sistem .....	18
3.5 Perancangan Tampilan Antarmuka .....	21
BAB IV IMPLEMENTASI DAN UJI COBA .....	29
4.1 Spesifikasi Perangkat .....	29
4.1.1 Perangkat Keras .....	29
4.1.2 Perangkat Lunak .....	29
4.2 Implementasi .....	29
4.3 Hasil Implementasi .....	37
4.4 Pengujian .....	41
4.4.1 Hasil Iterasi Pertama .....	42
4.4.2 Hasil Iterasi Kedua .....	43
4.4.3 Hasil Iterasi Ketiga .....	45
BAB V KESIMPULAN DAN SARAN .....	47
5.1 Kesimpulan .....	47
5.2 Saran .....	48
DAFTAR PUSTAKA .....	49
LAMPIRAN 1 .....	51
LAMPIRAN 2 BIOGRAFI PENULIS .....	52

## DAFTAR GAMBAR

Gambar 2.1 Area interaksi Leap Motion Controller .....	5
Gambar 2.2 Contoh Penyajian Pertanyaan PSSUQ .....	9
Gambar 2.3 Area plotting dari pencarian problem usability .....	11
Gambar 3.1 <i>Flowchart</i> Sistem <i>Video Game</i> Secara Umum .....	18
Gambar 3.2 <i>Flowchart</i> Proses Detail “ <i>Game Scene</i> ” .....	20
Gambar 3.3 Diagram Tampilan <i>Menu Utama</i> .....	22
Gambar 3.4 Diagram Tampilan <i>HOW TO PLAY</i> .....	23
Gambar 3.5 Diagram Tampilan <i>THEME SELECT</i> .....	24
Gambar 3.6 Diagram Tampilan <i>LEVEL SELECT</i> .....	25
Gambar 3.7 Diagram Tampilan <i>Level</i> .....	26
Gambar 3.8 Diagram Tampilan Antarmuka <i>START</i> .....	27
Gambar 3.9 Diagram Tampilan Antarmuka <i>FINISH</i> .....	28
Gambar 4.1 Leap Motion Controller Dengan Penjepit Baju .....	30
Gambar 4.2 Potongan Kode Untuk Inisialisasi <i>Timer</i> .....	31
Gambar 4.3 Potongan Kode <i>LeapRTS</i> Untuk Deteksi Gestur .....	32
Gambar 4.4 Potongan Kode Pengecekan <i>ID ObjectToMove</i> .....	33
Gambar 4.5 Potongan Kode Pengecekan Rotasi <i>ObjectToMove</i> .....	34
Gambar 4.6 Tampilan Pengisian Array Rotasi <i>TargetObject</i> .....	35
Gambar 4.7 Potongan Kode Yang Dijalankan Bila Pengecekan Berhasil .....	35
Gambar 4.8 Potongan Kode Penambahan Skor dan Pengecekan Jumlah Skor ....	36
Gambar 4.9 Potongan Kode Penghentian <i>Timer</i> dan Penyimpanan <i>highScore</i> ....	37
Gambar 4.10 Tampilan <i>Menu Utama</i> .....	38
Gambar 4.11 Tampilan “ <i>How to Play</i> ” .....	38
Gambar 4.12 Tampilan Pemilihan Tema .....	39
Gambar 4.13 Tampilan Pemilihan <i>Level</i> .....	39
Gambar 4.14 Tampilan Permainan Dimulai .....	40
Gambar 4.15 Tampilan Permainan Selesai .....	40



## DAFTAR TABEL

Tabel 3.1 Daftar Aset .....	16
Tabel 4.1 Rekapitulasi Survei Iterasi Pertama .....	42
Tabel 4.2 Tabel Hasil Analisis Data Pengujian Iterasi Pertama .....	43
Tabel 4.3 Rekapitulasi Survei Iterasi Kedua.....	43
Tabel 4.4 Tabel Hasil Analisis Data Pengujian Iterasi Kedua .....	44
Tabel 4.5 Rekapitulasi Survei Iterasi Ketiga.....	45
Tabel 4.6 Tabel Hasil Analisis Data Pengujian Iterasi Ketiga.....	46

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Sebuah statistik dari desainer *game* Jane McGonigal menyatakan bahwa terdapat setengah miliar penduduk bumi yang memainkan *video game* satu jam sehari, 183 juta adalah penduduk Amerika dengan 5 juta penduduk Amerika setidaknya menghabiskan waktu 40 jam seminggu, menjadikan *video game* salah satu aktivitas waktu luang yang populer di Amerika (Morley, 2012). Beberapa riset mencantumkan *video game* dapat digunakan sebagai alternatif pengobatan dalam penyakit stress medis (Carmen Russoniello, Kevin O'Brien, Jennifer Parks, 2009), atau meningkatkan kemampuan kinerja otak (Ian Spence & Jing Feng, 2010).

Salah satu jenis dari *video game* yang mendukung penelitian tersebut adalah *puzzle*, yang memiliki fokus untuk menyelesaikan masalah, dan mempelajari penggunaan *tool* dari mekanisme desain yang ada (Mark J. P. Wolf, 2001). Dalam penelitian yang dilakukan oleh Adam & Michael (2014) menyatakan bahwa penggunaan rutin *puzzle video game* menghasilkan peningkatan yang signifikan pada kinerja *executive functions* pada otak. Namun Adam & Michael berpendapat bahwa penelitian yang dilakukan hanya menggunakan *video game* berbasis *touch screen* yang didesain dalam platform *smartphone* sehingga sugesti mengenai *alternative mechanism* lainnya dapat memberikan *input* penelitian selanjutnya.

Leap Motion Controller adalah sebuah alat yang berfungsi sebagai *gesture recognition* dalam antarmuka natural (Leap Motion, 2014). Dengan *gesture recognition*, Leap Motion Controller dapat menerima input 3D yang lebih natural dibandingkan dengan interaksi *touch screen* yang hanya menerima input 2D

(Eugene Chang, 2012). Oleh karena itu, dilakukan penelitian perancangan aplikasi *puzzle game* dengan *gesture recognition mechanism* menggunakan Leap Motion Controller.

## **1.2 Rumusan Masalah**

Berdasarkan latar belakang masalah, rumusan masalah yang didapat adalah bagaimana cara merancang dan membangun aplikasi *puzzle game* “*Match Shape*” berbasis Leap Motion Controller?

## **1.3 Batasan Masalah**

Batasan masalah dalam penelitian ini adalah sebagai berikut.

1. Permainan akan berupa sebuah *puzzle* yang bertujuan untuk menyamai rupa benda dari bayangan yang muncul di area yang sudah ditentukan.
2. Rupa beda yang disamakan akan berbentuk *geometry* 3D.
3. Pemain menggunakan tangan untuk menggerakkan benda dalam sumbu X, Y, dan Z.
4. Gerakan benda hanya berupa translasi, dan rotasi.
5. Pemain hanya dapat menggunakan satu atau dua tangan untuk mengontrol permainan.
6. Permainan hanya dapat dimainkan oleh satu orang dalam waktu yang bersamaan.
7. Permainan hanya akan menyediakan 16 *puzzle* untuk diselesaikan.
8. Aplikasi akan dibangun dalam game engine Unity3D, dan menggunakan SDK yang sudah disediakan oleh Leap Motion™.
9. Aplikasi akan dijalankan dalam sistem operasi Windows 10.

#### **1.4 Tujuan Penelitian**

Penelitian ini memiliki tujuan untuk merancang dan membangun aplikasi *Puzzle Game “Match Shape”* dengan menggunakan *gesture recognition* pada Leap Motion Controller.

#### **1.5 Manfaat Penelitian**

Dengan dilakukan penelitian ini, diharapkan pengguna dapat menggunakan aplikasi sebagai salah satu alat alternatif dalam peningkatan konsentrasi dan kemampuan kognitif pengguna. Selain itu diharapkan aplikasi dapat digunakan sebagai aplikasi pendukung dalam penelitian *executive function*.

#### **1.6 Sistematika Penulisan**

Sistematika penulisan laporan skripsi ini dijelaskan sebagai berikut.

##### **Bab I Pendahuluan**

Berisi Latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian dan sistematika penulisan.

##### **Bab II Tinjauan Pustaka**

Berisi landasan teori

##### **Bab III Metodologi Penelitian dan Perancangan Sistem**

Berisi metodologi penelitian yang digunakan serta proses perancangan terkait dengan kebutuhan sistem dan desain keseluruhan sistem yang meliputi studi literatur, perancangan sistem, serta daftar penggunaan aset.

##### **Bab IV Implementasi dan Uji Coba**

Berisi penjelasan mengenai implementasi dan hasil uji coba sistem.

##### **Bab V Kesimpulan Dan Saran**

Berisi kesimpulan dan saran untuk penelitian selanjutnya.

## **BAB II**

### **TINJAUAN PUSTAKA**

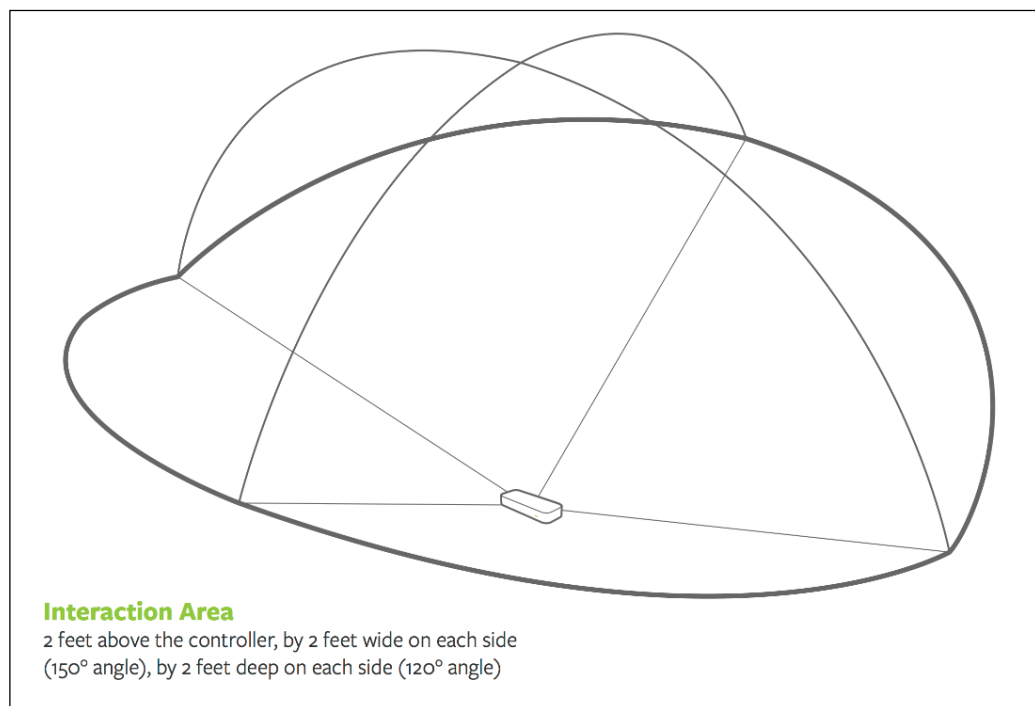
#### **2.1 Antarmuka Natural**

Antarmuka natural adalah sebuah filosofi desain dan sumbu dalam pembangunan sebuah produk yang menyerupai dunia nyata dalam pandangan manusia. Elemen natural dalam antarmuka natural bermaksud pada interaksi pengguna, dan perasaan pengguna dalam penggunaan produk. Bukan bermaksud pada tampilan antarmuka dari sebuah produk. Pengertian antarmuka natural bermaksud pada interaksi sebuah alat yang bermetode selain mouse dan keyboard yang disaat bersamaan terasa natural dan intuitif pada pengguna. Maka, antarmuka natural dapat berbasis pada suara, sentuhan, dan deteksi pergerakan. (Daniel Wigdor & Daniel Wixon, 2011)

Dalam hal praktikalnya, antarmuka natural telah digunakan oleh banyak bidang seperti bidang industri, medis, dan hiburan. Salah satu contoh dari penggunaan antarmuka natural seperti pemanfaatan Kinect yang adalah sebuah kamera dengan fungsi inframerah dimana kamera dapat mendeteksi seluruh gerakan badan dan mendeteksi luas ruangan. Kinect yang dipasarkan sebagai alat untuk bermain *video game* saat ini digunakan oleh salah satu rumah sakit di Kanada yang menggunakan Kinect di ruang teater operasi untuk mengakses foto X-ray pasien (Grzegorz Glonek & Maria Pietruszka, 2012). Selain itu Kinect juga digunakan sebagai alat asistensi untuk para lansia dalam penggunaan komputer atau berkomunikasi dengan orang lain (Bruno Loureiro & Rui Rodrigues, 2011).

## 2.2 Leap Motion Controller

Leap Motion Controller adalah produk yang diciptakan dari Leap Motion™ yang didesain sebagai pengontrol berbasis sensor gerakan. Alat USB dengan 2 kamera inframerah dan 3 LED inframerah dengan bentuk yang kecil berdimensi 1.2 x 3 x 7.6cm (Robert McCartney, Jie Yuan, & Hans-Peter Bischof, 2015) dapat mengikuti gerakan apapun diarea dengan ukuran 2 kaki di atas alat sebesar 150°, dan 2 kaki di samping alat sebesar 120° melalui pantulan cahaya inframerah yang disediakan (Leap Motion, 2014).



Gambar 2.1 Area interaksi Leap Motion Controller (Leap Motion, 2014)

Berbeda dengan Kinect yang adalah sebuah alat sensor gerakan keluaran Microsoft yang dapat mendeteksi gerakan seluruh badan, Leap Motion Controller memiliki kelebihan dalam akurasi dan juga harga yang lebih terjangkau karena secara spesifik hanya menangkap gerakan tangan manusia saja walaupun menggunakan teknologi kamera inframerah yang sama.

Leap Motion Controller sendiri bekerja menggunakan algoritma spesifik yang masih dirahasiakan oleh Leap Motion™, namun Leap Motion™ telah membangun SDK spesifik yang dapat digunakan oleh pengembang dalam membangun algoritma atau aplikasi dengan dukungan bahasa pemrograman yang bervariasi dari Java, Python, JavaScript, Objective C, C#, dan C++. Leap Motion™ juga mendukung langsung pengembangan aplikasi menggunakan *game engine* seperti Unity, dan Unreal dengan *plugin* yang tersedia dari Leap Motion™, dan komunitas Leap Motion.

### **2.3 Struktur Permainan**

Tracy Fullerton (2008) dalam buku Game Design Workshop menyatakan bahwa dalam pembangunan permainan walau sebuah permainan memiliki tipe yang berbeda, pada dasarnya terdapat persamaan dari setiap permainan yang dibuat. Hal ini dibagi dalam dua bagian yaitu *Formal Elements* dan *Dramatic Elements*.

*Formal Elements* adalah elemen kritis sebagai penggerak permainan, tanpa adanya *Formal Elements* sebuah permainan tidak dapat dikatakan permainan. Berikut adalah *Formal Elements* menurut Fullerton.

#### *1. Players*

Elemen utama ini mendefinisikan pemain yang menggunakan aplikasi, menentukan tipe pemain, jumlah pemain, dan peran pemain dari permainan.

#### *2. Objectives*

*Objectives* mendefinisikan tujuan dari permainan sebagai alat untuk membuat pemain mencapai sesuatu yang dapat dibanggakan.

#### *3. Procedures*

*Procedures* mendefinisikan tahapan yang diperlukan dari seorang pemain untuk menyelesaikan *Objectives*.

#### 4. *Rules*

*Rules* mendefinisikan aturan dan tindakan yang diperbolehkan atau dilarang dari permainan.

#### 5. *Resources*

*Resources* mendefinisikan elemen yang ada dalam sebuah permainan yang berhubungan dengan pemain untuk digunakan.

#### 6. *Conflict*

*Conflict* mendefinisikan problem yang akan muncul dalam menyelesaikan tujuan akhir dari permainan berdasarkan *Rules*.

#### 7. *Boundaries*

*Boundaries* mendefinisikan batasan dari permainan, atau tidak. Batasan ini dibuat untuk mendefinisikan bagian mana saja yang masih termasuk sebagai permainan.

#### 8. *Outcome*

*Outcome* mendefinisikan hasil akhir dari sebuah permainan, hal ini dapat berupa kondisi akhir dari sebuah permainan, atau sebuah hasil yang menunjukkan status dari pemain saat permainan berakhir.

### 2.4 Post Study Usability Questionnaire

*Post Study Usability Questionnaire* (PSSUQ) adalah kuesioner sebanyak 19 soal yang bertujuan untuk menilai tingkat kegunaan sistem pada pengguna. Berasal dari project *internal* IBM bernama *System Usability MetricS* (SUMS) yang bertujuan untuk mendokumentasi dan memvalidasi prosedur dalam pengukuran



*system usability*, termasuk performa, masalah kegunaan, dan kepuasan pengguna (James R. Lewis, 2002).

Berdasarkan analisa faktor psikometri oleh James R. Lewis (1995), dasar dari perhitungan skala PSSUQ adalah

1. *Overall*: Rata-rata respon dari pertanyaan 1 sampai 19. Rata-rata penilaian menyeluruh dari kepuasan penggunaan.
2. *SysUse*: Rata-rata respon dari pertanyaan 1 sampai 8. Rata-rata penilaian dari kegunaan aplikasi.
3. *InfoQual*: Rata-rata respon dari pertanyaan 9 sampai 15. Rata-rata penilaian dari informasi yang disajikan dari aplikasi.
4. *IntQual*: Rata-rata respon dari pertanyaan 16 sampai 18. Rata-rata penilaian dari tampilan aplikasi.

Seluruh pertanyaan memiliki respon yang diukur berdasarkan skala 1 sampai 7, dimana 1 menandakan “Sangat setuju”, dan 7 menandakan “Sangat tidak setuju”.

Berikut adalah daftar dari *Post Study Usability Questionnaire* dalam bahasa Inggris.

1. *Overall, I am satisfied with how easy it is to use this system.*
2. *It was simple to use this system.*
3. *I could effectively complete the tasks and scenarios using this system.*
4. *I was able to complete the tasks and scenarios quickly using this system.*
5. *I was able to efficiently complete the tasks and scenarios using this system.*
6. *I felt comfortable using this system.*
7. *It was easy to learn to use this system.*
8. *I believe I could become productive quickly using this system.*

9. *The system gave error messages that clearly told me how to fix problems.*
10. *Whenever I made a mistake using the system, I could recover easily and quickly.*
11. *The information (such as on-line help, on-screen messages and other documentation) provided with this system was clear.*
12. *It was easy to find the information I needed.*
13. *The information provided for the system was easy to understand.*
14. *The information was effective in helping me complete the tasks and scenarios.*
15. *The organization of information on the system screens was clear.*
16. *The interface of this system was pleasant.*
17. *I liked using the interface of this system.*
18. *This system has all the functions and capabilities I expect it to have.*
19. *Overall, I am satisfied with this system.*

Dalam kuesioner, contoh pertanyaan yang akan disajikan sebagai berikut.

1. Overall, I am satisfied with how easy it is to use this system.								
Strongly Agree	1	2	3	4	5	6	7	Strongly Disagree
...								

Gambar 2.2 Contoh Penyajian Pertanyaan PSSUQ

## 2.5 Likert Scale

Likert Scale adalah sebuah metode pengukuran data kualitatif menjadi kuantitatif. Terdiri dari sekumpulan daftar, berupa pertanyaan dengan jawaban yang diwakili dengan sebuah nilai. Hasil dari jawaban kemudian dikumpulkan per kategori yang kemudian dianalisa secara statistik (Joshi dkk., 2015).

Untuk setiap perhitungan Likert Scale dilakukan dengan memberikan bobot pada setiap jawaban yang kemudian dikalikan dengan jumlah jawaban untuk

kemudian diperoleh skor yang kemudian digunakan untuk menghitung interpretasi skor.

Likert Scale dengan tujuh kategori memiliki pembagian bobot sebagai berikut.

1. Kategori “Sangat setuju” diberikan bobot 7.
2. Kategori “Cukup setuju” diberikan bobot 6.
3. Kategori “Setuju” diberikan bobot 5.
4. Kategori “Cukup” diberikan bobot 4.
5. Kategori “Tidak setuju” diberikan bobot 3.
6. Kategori “Cukup tidak setuju” diberikan bobot 2.
7. Kategori “Sangat tidak setuju” diberikan bobot 1.

Kriteria Interpretasi Skor yang digunakan untuk mengolah jawaban Likert Scale adalah sebagai berikut.

1. Sangat Lemah (Kategori 1) dengan syarat  $\geq 0\%$  dan  $< 20\%$
2. Lemah (Kategori 2) dengan syarat  $\geq 20\%$  dan  $< 40\%$
3. Cukup (Kategori 3) dengan syarat  $\geq 40\%$  dan  $< 60\%$
4. Kuat (Kategori 4) dengan syarat  $\geq 60\%$  dan  $< 80\%$
5. Sangat Kuat (Kategori 5) dengan syarat  $\geq 80\%$

Untuk menghitung intepretasi skor digunakan rumus sebagai berikut.

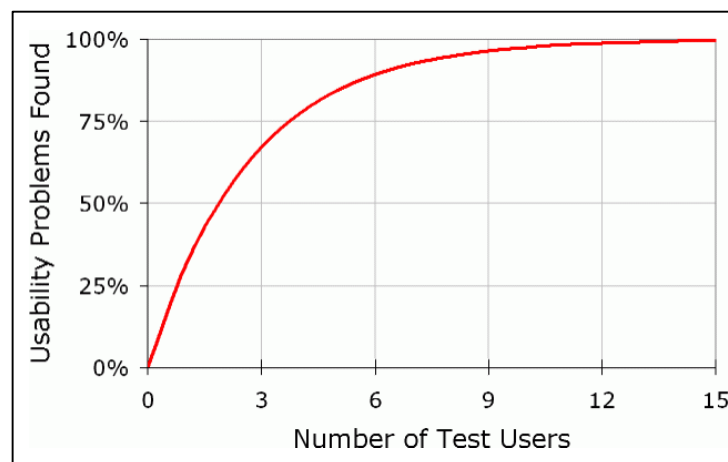
$$\begin{aligned} \text{Hasil} = & (\text{Jumlah sangat setuju} \times 7 + \text{Jumlah cukup setuju} \times 6 \\ & + \text{Jumlah setuju} \times 5 + \text{Jumlah cukup} \times 4 + \text{Jumlah kurang} \\ & \times 3 + \text{Jumlah cukup kurang} \times 2 + \text{Jumlah sangat kurang} \\ & \times 1) \div \text{Jumlah skala} \times \text{Jumlah sampel} \end{aligned}$$

## 2.6 Pencarian Problem Usability

Jakob Nielsen dan Thomas K. Landauer (1993) menjelaskan dalam penelitian yang dilakukan untuk mencari model matematika dalam pencarian problem *usability*, angka problem usability dapat ditemukan dengan menggunakan formula:

$$N(1 - (1 - L)^n) \quad \text{..( 2.1 )}$$

Di mana  $N$  adalah jumlah dari problem *usability* dalam desain, dan  $L$  adalah proporsi dalam problem *usability* yang ditemukan oleh seorang pengguna yang biasanya bernilai 31%. Penggambaran kurva dari  $L=31\%$  memberikan hasil berikut:



Gambar 2.3 Area plotting dari pencarian problem usability

(<https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/>)

Hasil menunjukkan bahwa pengujian dari 0 orang menghasilkan 0% problem. Pada saat lebih banyak pengujian dilakukan, problem yang didapatkan oleh penguji pertama meningkat hingga 31%, penguji kedua akan memberikan peningkatan yang menyerupai penguji pertama, begitu juga dengan penguji ketiga. Namun semakin bertambahnya penguji yang ada, peningkatan problem akan berkurang karena mereka akan melihat problem yang sama lagi dan lagi. Sehingga pengujian akan lebih optimal pada saat membatasi pengujian pada orang ke-5. Nielsen juga menjelaskan bahwa pengujian akan lebih optimal bila dapat dilakukan berulang kali.

Bila pengujian mampu dilakukan dengan pengguna yang lebih banyak, lakukan kembali pengujian dengan membaginya ke 5 orang per uji coba. Untuk setiap 5 orang, lakukan perbaikan dengan mengkaji desain kembali, dan menguji coba kembali (Jakob Nielsen, 2000)

## BAB III

### METODOLOGI PENELITIAN DAN PERANCANGAN SISTEM

#### 3.1 Metode Penelitian

Metode penelitian yang akan dilakukan adalah sebagai berikut

1. Studi literatur

Melakukan studi terhadap sumber-sumber literatur, jurnal, laporan, dan artikel mengenai teori yang berkaitan dengan antarmuka natural, *gesture recognition*, *puzzle game*, dan Leap Motion Controller.

2. Desain *gameplay*, antarmuka natural dan antarmuka grafis.

Melakukan desain sistem *puzzle game* dengan menentukan variasi *puzzle* yang akan dijadikan bahan permainan, kemudian melakukan desain antarmuka grafis, dan antarmuka natural berbasis gestur.

3. Pengembangan aplikasi *puzzle game*

Melakukan implementasi antarmuka natural ke dalam aplikasi *puzzle game* menggunakan Unity3D dengan bahasa pemrograman C# dan menggunakan SDK Leap Motion™ untuk menerapkan sistem kontrol Leap Motion Controller pada aplikasi.

4. Testing

Testing dilakukan dengan melakukan test aplikasi *puzzle game* pada *notebook* Asus N43S sebagai *notebook* utama untuk pengujian lapangan beserta penggunaan langsung Leap Motion Controller yang diletakkan di depan *notebook*.

5. Uji Coba Lapangan

Uji coba lapangan akan dilakukan pada 5 orang dalam rangkap usia 20-40 tahun. Peserta akan diajak untuk melakukan sejumlah tugas yang akan diobservasi oleh penguji. Pada akhir pengujian, seluruh peserta akan diberikan sebuah kuesioner berupa daftar PSSUQ.

#### 6. Evaluasi

Evaluasi akan dilakukan dengan menganalisis data yang didapat dari pengumpulan data selama uji coba. Data tersebut adalah waktu pengerjaan tugas sebelum menggunakan aplikasi, waktu pengerjaan tugas setelah menggunakan aplikasi, banyak *level* yang bisa diselesaikan dalam aplikasi *puzzle game*, dan kuesioner PSSUQ.

### 3.2 Struktur Permainan

Judul Permainan: Topia Fabricator

*Formal Elements* yang terdapat di dalam permainan dapat dijabarkan sebagai berikut.

#### 1. Players

*Single Player Game.*

#### 2. Objectives

Memindahkan dan memutar benda yang berada didalam *level* ke tempat yang ditentukan.

#### 3. Procedures

- a. Pilih tombol “*Start Game*” untuk memilih tema dari *level*.
- b. Pilih *level* yang diinginkan untuk memulai permainan.
- c. Dengan gestur tangan mencari benda yang berada dalam *level* dengan menggerakkan kamera.

- d. Dengan gestur tangan mengambil benda yang berada dalam *level* dan melepaskan atau meletakan benda ke tempat yang ditentukan.
- e. *Level* berakhir setelah pemain meletakan seluruh benda ke tempat yang ditentukan, diberikan informasi waktu yang dihabiskan untuk menyelesaikan *level*.

#### **4. Rules**

- a. Permainan dikontrol menggunakan gestur tangan untuk interaksi objek dan *gamepad* untuk mengendalikan kamera, melakukan kalibrasi, dan mengulang *level*.
- b. Permainan dibatasi oleh beberapa gestur yang sudah ditentukan, yaitu gestur menunjuk, dan gestur menjepit.
- c. Terdapat gravitasi yang memberi pengaruh terhadap objek yang pemain ambil, bila objek yang diambil dilepas oleh pemain, objek akan terpengaruh dengan efek gravitasi dan jatuh ke *area level*.
- d. Pemilihan *menu* dapat dilakukan dengan gestur menunjuk diikuti dengan menekan ke area yang diinginkan.
- e. Pengambilan barang dapat dilakukan dengan gestur menjepit dengan jari telunjuk dan ibu jari setelah mendekatkan atau menyentuh objek yang ada di dalam *level*.
- f. Rotasi kamera dapat dilakukan dengan menggerakan *analog stick* kiri ke arah kiri atau kanan.
- g. Pembesaran atau pengecilan kamera dapat dilakukan dengan menggerakan *analog stick* kiri ke arah atas atau bawah.
- h. Untuk melakukan kalibrasi posisi, pemain dapat menggunakan tombol “LB” pada *gamepad*.



- i. Untuk mengulang *level* bila terjadi kesalahan total yang pemain ingin perbaiki, dapat menggunakan tombol “*Select*” pada *gamepad*.
- j. Untuk keluar dari *level*, dapat menggunakan tombol “*Start*” pada *gamepad*.

## 5. Resources

- a. Pemain dapat memindahkan, memutar, dan menjatuhkan benda.
- b. Pemain dapat memutar, dan memperbesar kamera.
- c. *Timer* yang akan bertambah.

## 6. Conflict

Kesulitan dalam penyelesaian *level*.

## 7. Boundaries


- a. Interaksi terbatas untuk mengambil benda, meletakan benda, menjatuhkan benda, menggerakan kamera dalam *level*.
- b. Kendali terbatas menggunakan gestur dan *gamepad*.
- c. Tidak dapat menghentikan sementara permainan.
- d. Objek yang tidak dapat diambil karena jatuh atau sulit diraih tidak dapat di *reset*, dan mengharuskan pemain untuk *restart* permainan.




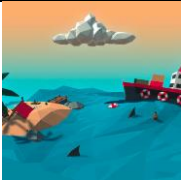

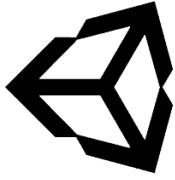

## 8. Outcome


Diinformasikan apakah pemain menyaingi waktu rekam sebelumnya.

### 3.3 Penggunaan Aset

Tabel 3.1 Daftar Aset

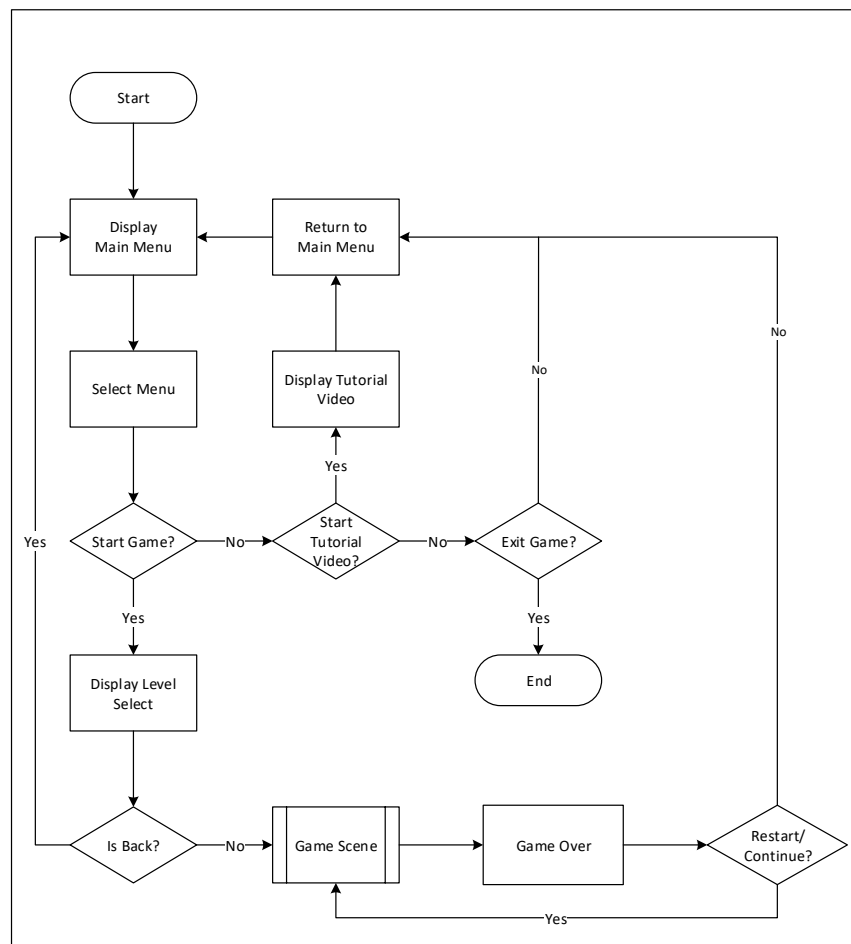
Gambar	Penjelasan	Sumber
	<i>Leap Motion Orion Plugin: plugin untuk mendapatkan data gestur dari Leap Motion Controller.</i>	Leap Motion. Inc

	<i>Leap Motion Hands Module:</i> Digunakan sebagai pemberian visualisasi tangan yang nyata.	Leap Motion. Inc
	<i>Leap Motion UI Input Module:</i> <i>script</i> untuk memberikan interaksi terhadap menu <i>virtual</i> .	Leap Motion. Inc
	<i>Low Poly Ultimate Pack:</i> Sebuah <i>Asset Pack</i> untuk pembangunan <i>level</i> .	Pavel Novák
	<i>PolyIsland - Low Poly Assets:</i> Sebuah <i>Asset Pack</i> untuk pembangunan <i>level</i> .	Ovidiu Vladut
	<i>Farland Skies – Cloudy Crown:</i> <i>Asset Pack</i> untuk memberikan visualisasi <i>skybox</i> dalam permainan.	Borodar
	<i>ArrayPrefs2:</i> <i>Script</i> yang memberikan cara untuk menyimpan dan mengambil array dari berbagai tipe yang ada di <i>Unity</i>	Eric Haines
	<i>Brighton Bossa:</i> Lagu yang digunakan sebagai tema awal permainan dan tema <i>level</i> selesai.	Jair Claudino Rodrigues Junior, Guilherme Lopes Rodrigues, Pablo Love, Campbell E Browning

	<i>Samba Flamenca</i> : Lagu yang digunakan sebagai iringan <i>level</i> .	Miguel Moreno
---	--	---------------

### 3.4 Perancangan Sistem

Sistem terlebih dahulu didesain menggunakan *flowchart* sebagai landasan pemrograman sistem. Rancangan *video game* secara umum dapat dilihat pada Gambar 3.1.



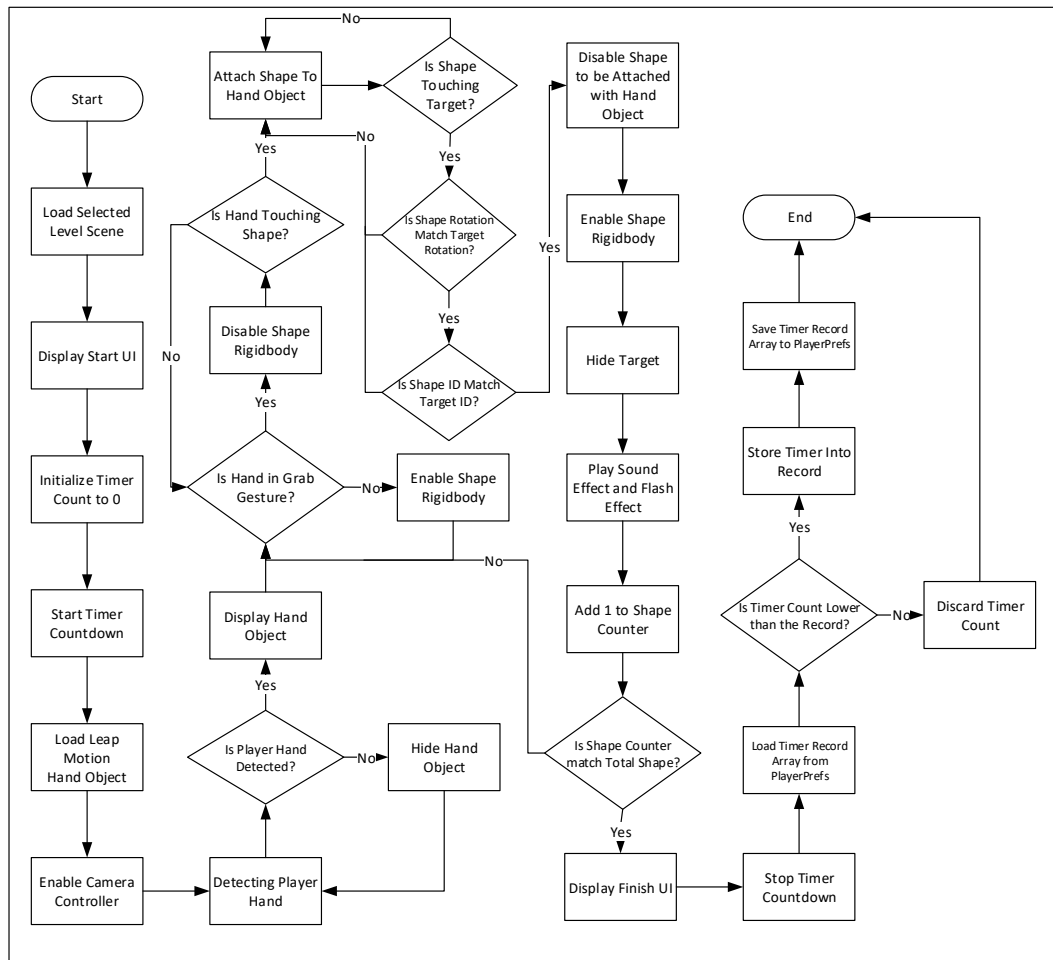
Gambar 3.1 *Flowchart* Sistem *Video Game* Secara Umum

Pada Gambar 3.1 dapat dilihat pada saat permainan dimulai, *menu* utama ditampilkan, Setelah *menu* utama ditampilkan, bila pemain memilih opsi “*Start*” maka akan ditampilkan “*Level Select*” untuk memilih *level* permainan. Saat “*Level*

*Select*” ditampilkan, bila pemain memilih opsi “*Back*” maka pemain akan dibawa kembali ke *menu* utama. Namun bila pemain memilih *level* yang diinginkan, akan ditampilkan “*Game Scene*” sesuai dengan *level* yang dipilih. Setelah permainan berakhir, pemain dapat memilih untuk melanjutkan *level* atau mengulang *level* atau mengakhiri permainan. Melanjutkan *level* dan mengulang *level* akan membawa pemain kembali ke “*Game Scene*”. Bila pemain memilih untuk mengakhiri permainan, maka pemain akan dibawa kembali ke *menu* utama.

Pada *menu* utama, pemain juga dapat memilih opsi *Tutorial* dan *Exit*. Memilih opsi *Tutorial* akan menampilkan *video tutorial* cara memainkan permainan, pada saat ini selama *video* berlangsung sampai selesai, pemain dapat menekan opsi *back* yang akan membawa pemain kembali *menu* utama. Memilih opsi *Exit* akan menutup aplikasi permainan.

“*Game Scene*” yang telah dijelaskan pada uraian sebelumnya dapat dijabarkan lebih lanjut pada Gambar 3.2



Gambar 3.2 Flowchart Proses Detail “Game Scene”

Pada Gambar 3.2 proses dimulai dengan membuka *scene* dari *level* yang dipilih, lalu kemudian menampilkan antarmuka “Start”. Kemudian menentukan ulang *Timer Count* ke 0, dan memulai *Timer Countdown*. Kemudian komponen *Leap Motion Hand Object* di masukkan, dan kontrol kamera dinyalakan.

Sistem kemudian akan melakukan deteksi tangan pemain. Bila tangan pemain dideteksi oleh *Leap Motion Controller* maka *Hand Object* akan ditampilkan. Bila *Hand Object* menyentuh benda dalam *level*, maka benda yang bersentuhan dengan *Hand Object* akan dimatikan *Rigidbodynya* agar dapat dibawa oleh *Hand Object*, namun pada saat ini *Hand Object* belum dapat mengangkat benda tersebut,

maka akan dilakukan cek apakah *Hand Object* melakukan gestur mengambil, bila gestur mengambil dilakukan benda akan dipasang di *Hand Object* dan akan bergerak bersamaan dengan *Hand Object*, bila tidak cek terhadap *Hand Object* menyentuh benda kembali dilakukan.

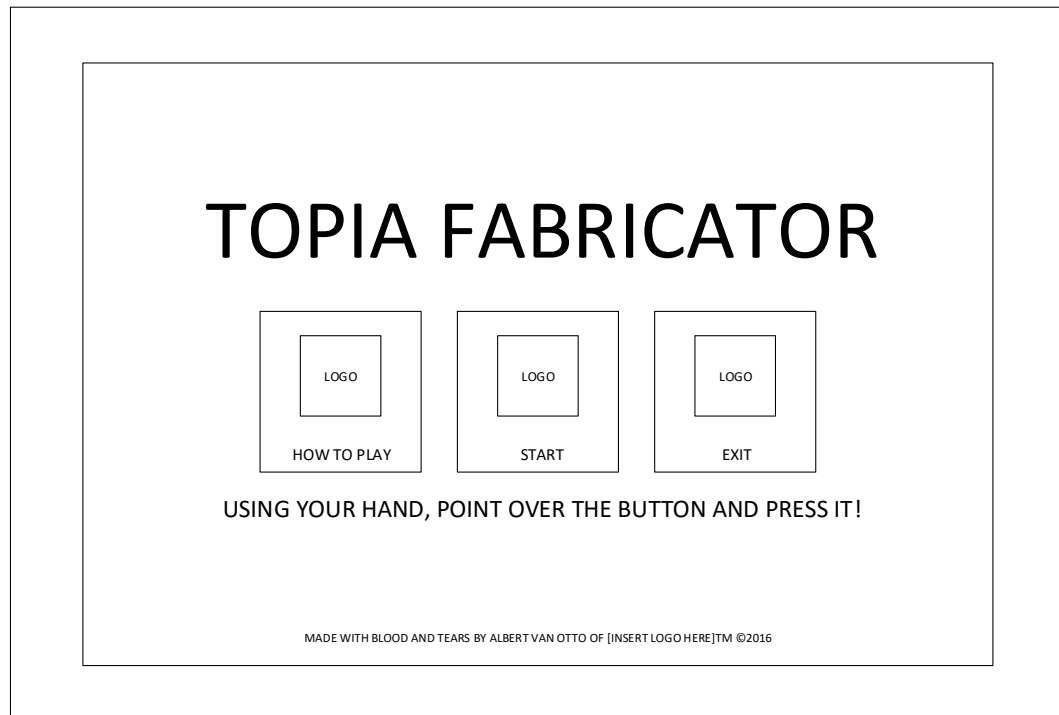
Kemudian cek terhadap posisi benda dilakukan, bila benda bersentuhan dengan *target* maka dilakukan cek apakah rotasi benda sama dengan *target* benda? Bila sama maka dilakukan cek berikutnya apakah ID dari benda sama dengan *target* benda? Bila sama maka benda akan dilepaskan dengan *Hand Object*, kemudian *Rigidbody* dari benda kembali dinyalakan, kemudian *target* akan disembunyikan, kemudian suara penanda beserta kilat cahaya akan dimainkan di antarmuka pemain, kemudian akan ditambahkan 1 poin ke *Shape Counter*. Bila rotasi benda berbeda dengan *target* benda atau bila ID dari benda berbeda dengan *target* benda maka benda akan tetap terpasang dengan *Hand Object*.

Kemudian cek terhadap *Shape Counter* dilakukan dengan jumlah *Total Shape* yang sebelumnya sudah ditentukan perlevel. Bila menyamai maka antarmuka “*Finish*” akan ditampilkan, kemudian *Timer Countdown* akan dihentikan, dan *Timer Record Array* diambil dari *PlayerPrefs*, kemudian dilakukan pengecekan apakah *Timer Count* lebih rendah dari *Timer Record Array* dari *level* yang dilakukan. Bila lebih rendah maka *Timer Count* akan disimpan kedalam *Timer Record Array* dan disimpan kembali kedalam *PlayerPrefs*. Bila tidak maka *Timer Count* akan dibuang.

### **3.5 Perancangan Tampilan Antarmuka**

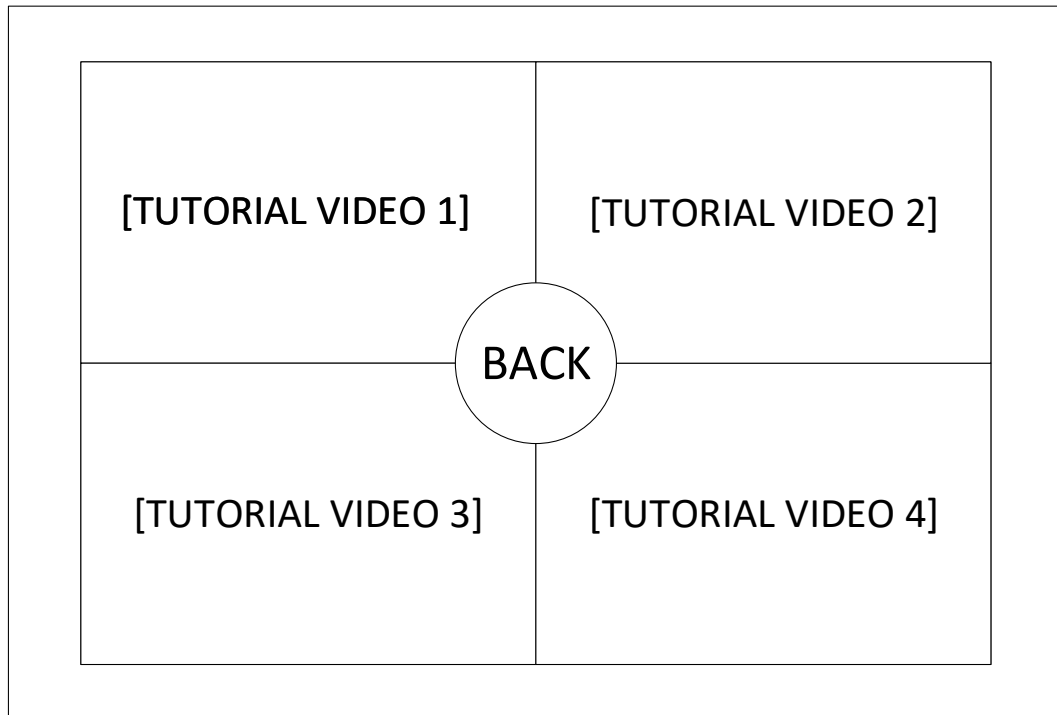
Rancangan antarmuka berikut dibuat untuk memberi panduan dalam pembuatan desain UI beserta fungsionalitas didalamnya sehingga memudahkan

proses pengembangan permainan. Berikut adalah rancangan daripada antarmuka *menu* utama yang dapat dilihat pada Gambar 3.3.



Gambar 3.3 Diagram Tampilan *Menu* Utama

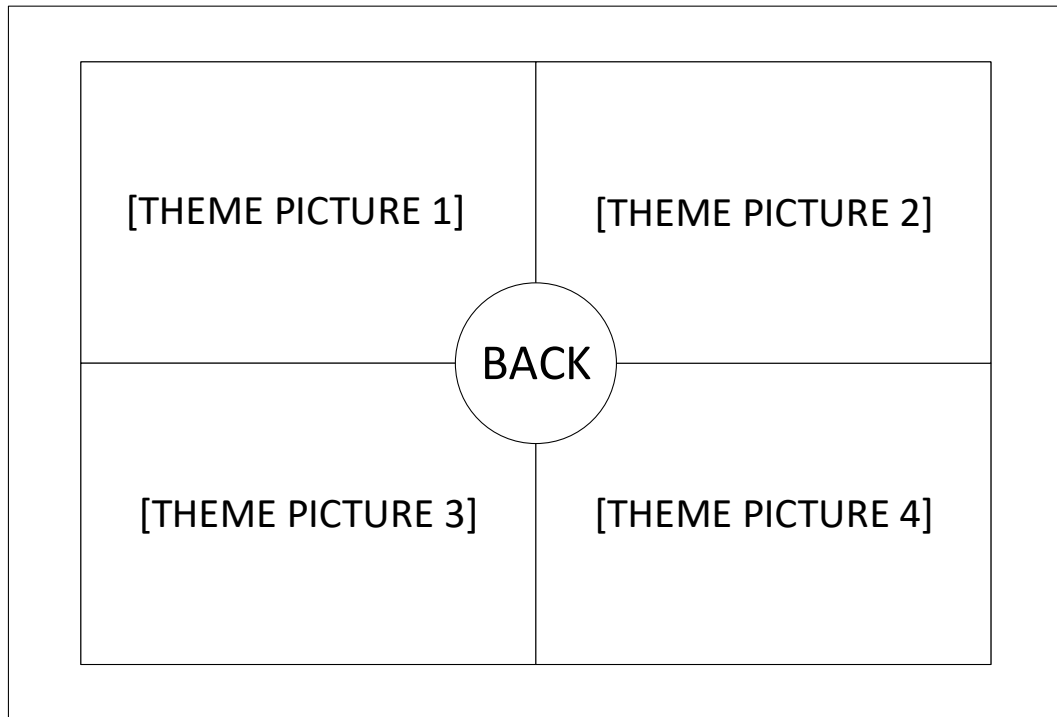
Gambar 3.3 menunjukkan diagram tampilan *menu* utama *video game*, tampilan awal ini memberikan informasi mengenai judul *video game*, tombol *menu*, informasi penggunaan *menu*, dan informasi pembuat *video game* bagian bawah. Terdapat 3 tombol *menu* untuk bernavigasi dalam *video game*, tombol “*HOW TO PLAY*” digunakan untuk menampilkan *video tutorial* untuk membantu pemain memainkan *video game*. Tombol “*START*” digunakan untuk memulai *video game* dengan berpindah antarmuka ke pemilihan level. Tombol “*EXIT*” digunakan untuk menutup aplikasi.



Gambar 3.4 Diagram Tampilan *HOW TO PLAY*

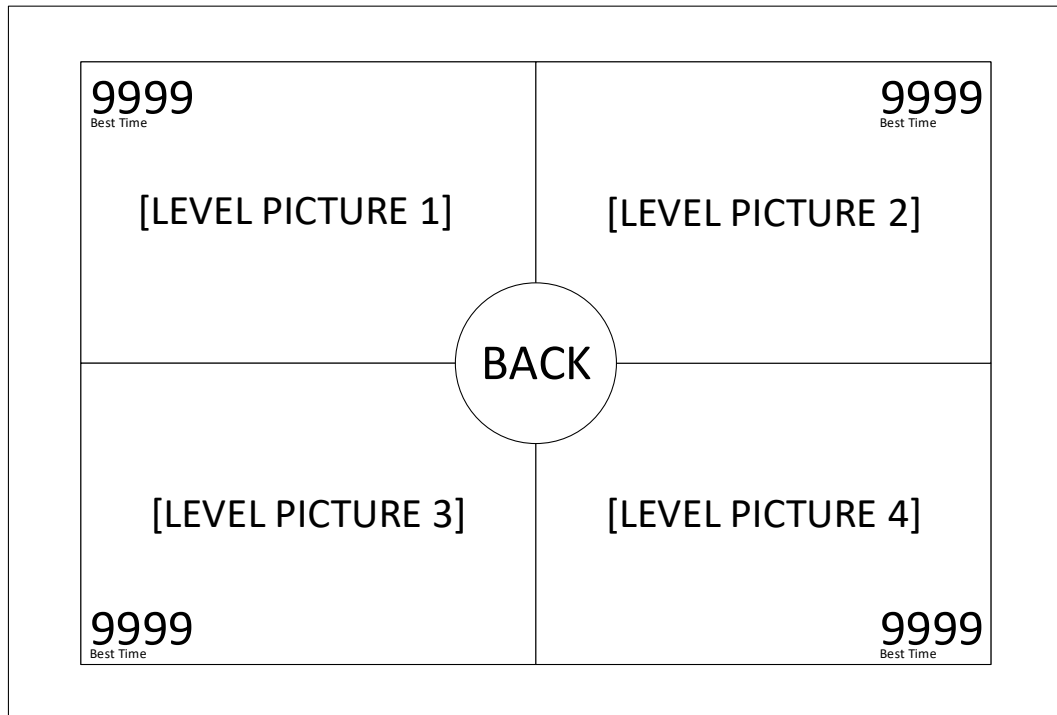
Gambar 3.4 menunjukkan diagram tampilan *HOW TO PLAY* yang tampil bila menekan tombol "*HOW TO PLAY*", tampilan ini menampilkan 4 video yang berbeda sebagai petunjuk cara bermain, video yang ditampilkan akan dimainkan secara otomatis dan akan mengulang secara otomatis. Menekan tombol "*BACK*" akan membawa pemain kembali ke *menu* utama.





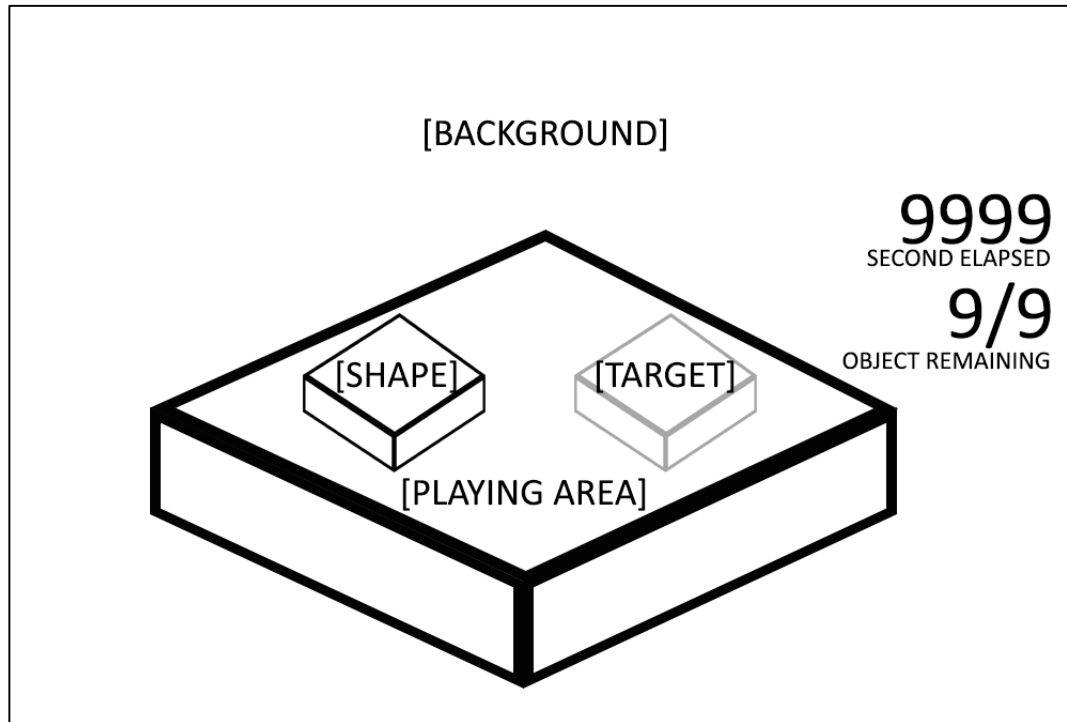
Gambar 3.5 Diagram Tampilan *THEME SELECT*

Gambar 3.5 menunjukkan diagram tampilan *THEME SELECT* yang tampil bila menekan tombol “*START*”, tampilan ini menampilkan 4 tombol yang memberikan visualisasi mengenai tema yang pemain bisa pilih. Setiap tombol akan membawa pemain menuju “*LEVEL SELECT*” dengan isi sesuai tema yang pemain pilih. Menekan tombol “*BACK*” akan membawa pemain kembali ke *menu* utama.



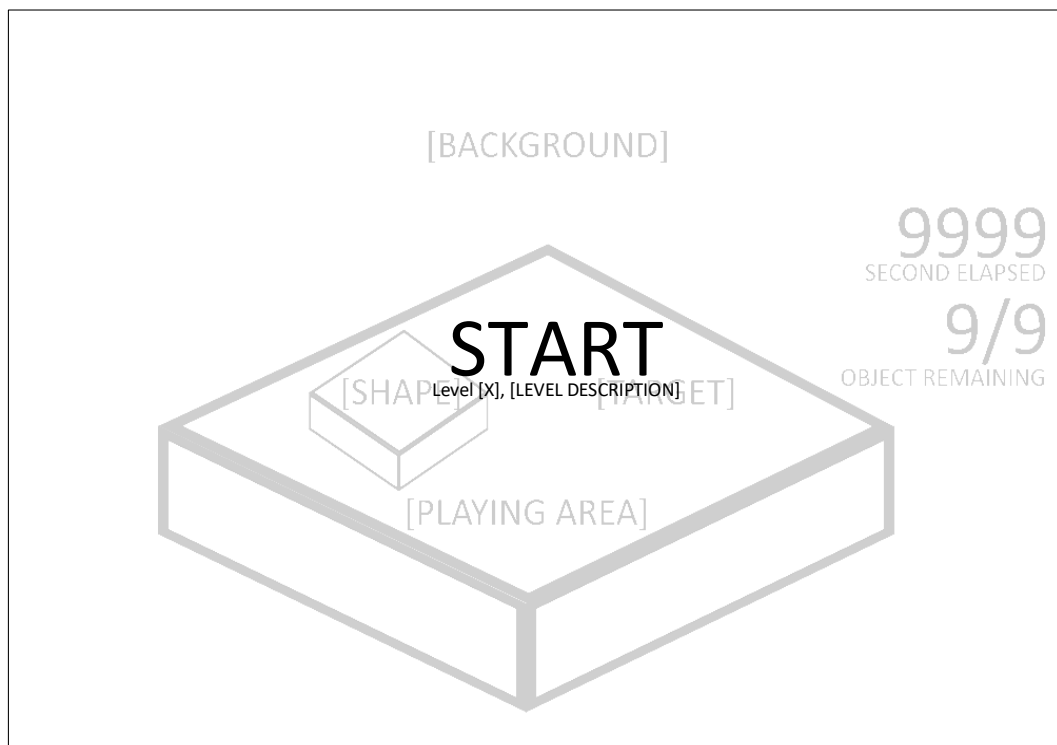
Gambar 3.6 Diagram Tampilan *LEVEL SELECT*

Gambar 3.6 menunjukkan diagram tampilan *LEVEL SELECT* yang tampil setelah memilih tema dari menu *THEME SELECT*, tampilan ini menampilkan 4 tombol yang memberikan tampilan *level* yang akan dimainkan beserta angka yang menunjukkan rekor dari setiap *level*. Setiap tombol akan membawa pemain menuju *level* yang pemain inginkan. Menekan tombol "*BACK*" akan membawa pemain kembali ke *menu* utama.



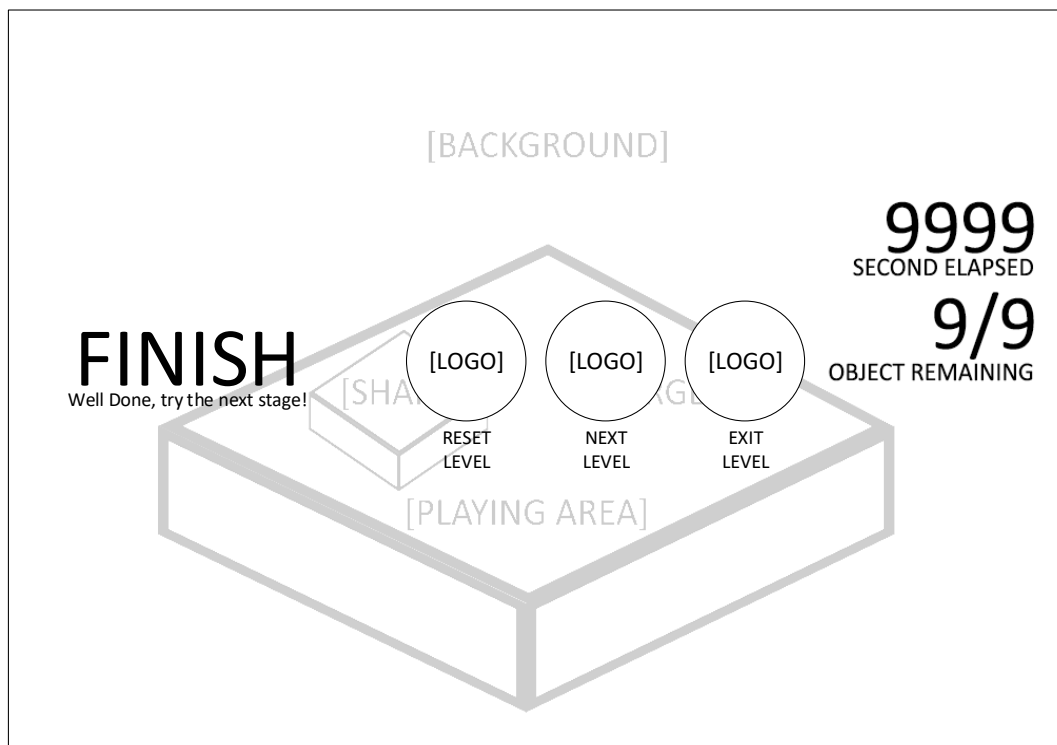
Gambar 3.7 Diagram Tampilan *Level*

Gambar 3.7 menunjukkan diagram tampilan *Level* yang tampil setelah memilih tema dari menu *LEVEL SELECT*. Tampilan ini menampilkan pandangan yang dilihat pemain saat memainkan level. Pada area tengah disebut *PLAYING AREA* adalah area dimana benda dan target akan berada. *SHAPE* adalah benda yang berwarna tidak transparan dan dapat digerakan oleh tangan. *TARGET* adalah benda berwarna transparan yang tidak dapat digerakan oleh tangan, berfungsi untuk menandakan tempat benda harus diletakkan. Tidak semua benda dapat digerakan oleh tangan, hanya benda yang menyerupai *TARGET* saja yang dapat digerakkan. Pada daerah kanan terdapat antarmuka *Timer* yang menunjukkan berapa lama waktu sudah dihabiskan untuk menyelesaikan *level* ini, notasi waktu berupa detik, dan terdapat antarmuka *Counter* yang menunjukkan jumlah objek yang sudah dicocokkan, dan yang harus dicocokkan.



Gambar 3.8 Diagram Tampilan Antarmuka *START*

Gambar 3.8 menunjukkan diagram tampilan yang tampil saat memulai level. Tampilan ini akan memberikan tulisan “*START*” beserta nomor *level* dan deskripsi *level*. Tampilan akan hilang setelah 3 detik.



Gambar 3.9 Diagram Tampilan Antarmuka *FINISH*

Gambar 3.9 menunjukkan diagram tampilan yang tampil saat menyelesaikan level. Tampilan ini akan memberikan tulisan “*FINISH*” dan “*Well done, try the next stage!*” dan menampilkan 3 tombol ditengah yang dapat dipilih oleh pemain. Tombol itu adalah “*RESET LEVEL*”, “*NEXT LEVEL*”, dan “*EXIT LEVEL*”. “*RESET LEVEL*” akan mengulang *level* yang sedang dimainkan saat ini, “*NEXT LEVEL*” akan melanjutkan *level* berikutnya dari tema yang dipilih. Bila *level* di tema yang dipilih sudah habis, tombol akan membuka *level* pertama kembali dari tema yang dipilih. “*EXIT LEVEL*” akan mengakhiri permainan, dan membawa kembali pemain ke *menu* utama.

## **BAB IV**

### **IMPLEMENTASI DAN UJI COBA**

#### **4.1 Spesifikasi Perangkat**

##### **4.1.1 Perangkat Keras**

Spesifikasi perangkat keras yang digunakan untuk menguji aplikasi *puzzle game* adalah sebagai berikut

- a. Processor: Intel® Core™ i5-2410M
- b. *Graphic Card*: NVIDIA GeForce GT 540M
- c. RAM: 4 GB
- d. *Hard Disk*: 160 GB
- e. Leap Motion Controller
- f. Sony Dualshock 4 Wireless Controller

##### **4.1.2 Perangkat Lunak**

Aplikasi yang akan digunakan dalam pengembangan aplikasi adalah sebagai berikut.

- a. Unity 5.4.2f1, *game engine* sebagai *engine* dasar *game* ini.
- b. Visual Studio 2015 Community Edition, digunakan untuk menulis kode program.

#### **4.2 Implementasi**

Aplikasi dibangun menggunakan *game engine* Unity3D versi 5.4.2f1 dengan target aplikasi untuk Windows, teknologi gestur Leap Motion Controller diterapkan menggunakan Leap Motion Orion v3.0. dengan metode *Head Mounted* untuk dapat menggunakan modul tambahan dari Leap Motion.

Metode *Head Mounted* memerlukan Leap Motion Controller untuk diposisikan pada arah yang sama pada mata, sehingga memerlukan alat untuk memasang alat pada badan agar kedua tangan tetap dapat berinteraksi dengan aplikasi. Sebuah penjepit yang dibuat menggunakan *3D Printer* yang didesain khusus untuk memasang Leap Motion Controller pada bagian leher baju digunakan untuk pengujian yang dapat dilihat pada Gambar 4.1.



Gambar 4.1 Leap Motion Controller Dengan Penjepit Baju

Pada Gambar 4.1 dapat dilihat bahwa Leap Motion Controller pertama direkatkan pada sebuah *Velcro* seukuran dengan Leap Motion Controller dan direkatkan pada *Velcro* seukuran dengan penjepit agar Leap Motion Controller dapat menempel dengan penjepit. Penjepit kemudian dapat dipasang pada pakaian pemain dibagian leher.

Pada saat permainan dimulai, Unity akan mencari Leap Motion Controller untuk mendeteksi input tangan dari pemain. Dengan SDK Leap Motion Orion, Leap Motion Controller dideteksi menggunakan library khusus yang diimplementasikan

menggunakan *prefab* untuk Unity yang dapat menkonfigurasi posisi kamera, dan model tangan untuk digambarkan dalam permainan.

Pada awal permainan, beberapa parameter akan diinisialisasi terlebih dahulu untuk memulai permainan, pertama terdapat inisialisasi *Timer*, yang selalu menset angka menjadi 0 sebelum level dimulai. *Timer* kemudian dijalankan menggunakan metode *Coroutine* yang menambahkan *variable timeElapsed* 1 per 1 detik, dan kemudian mengirimkan isi dari *timeElapsed* ke antarmuka pemain. Berikut adalah potongan kode yang mengerjakan fungsi *Timer*.

```
15 // Use this for initialization
16 void Start () {
17     timeElapsed = 0;
18     timeUIText = this.GetComponent<Text>();
19     corTime = StartCoroutine(timer());
20     isPlaying = true;
21 }
22
23 // Update is called once per frame
24 void Update () {
25
26 }
27
28 IEnumerator timer()
29 {
30     while (isPlaying == true) {
31         yield return new WaitForSeconds(1.0f);
32         timeElapsed++;
33         timeUIText.text = timeElapsed.ToString();
34         copyTimer.text = timeElapsed.ToString();
35     }
36 }
```

Gambar 4.2 Potongan Kode Untuk Inisialisasi *Timer*

Dalam sebuah *level*, selalu terdapat setidaknya sebuah *game object* bernama *TargetObject* beserta *ObjectToMove* yang menjadi persyaratan untuk menyelesaikan *level*. *TargetObject* dan *ObjectToMove* adalah sebuah kumpulan komponen yang mencakup *model* dari benda, *collider*, *rigidbody*, dan *script*. Terdapat 3 jenis *script* berbeda yang memenuhi setiap komponen. *Script LeapRTS* yang diletakan dalam *game object ObjectToMove* berfungsi untuk mengaktifkan rotasi dan translasi benda menggunakan Leap Motion Controller.



```

71 void Update() {
72     if (Input.GetKeyDown(_toggleGuiState)) {
73         _showGUI = !_showGUI;
74     }
75
76     bool didUpdate = false;
77     if(_pinchDetectorA != null)
78         didUpdate |= _pinchDetectorA.DidChangeFromLastFrame;
79     if(_pinchDetectorB != null)
80         didUpdate |= _pinchDetectorB.DidChangeFromLastFrame;
81
82     if (didUpdate) {
83         transform.SetParent(null, true);
84     }
85
86     if (_pinchDetectorA != null && _pinchDetectorA.IsActive &&
87         _pinchDetectorB != null && _pinchDetectorB.IsActive) {
88         transformDoubleAnchor();
89     } else if (_pinchDetectorA != null && _pinchDetectorA.IsActive) {
90         transformSingleAnchor(_pinchDetectorA);
91     } else if (_pinchDetectorB != null && _pinchDetectorB.IsActive) {
92         transformSingleAnchor(_pinchDetectorB);
93     }
94
95     if (didUpdate) {
96         transform.SetParent(_anchor, true);
97     }
98 }

```

Gambar 4.3 Potongan Kode *LeapRTS* Untuk Deteksi Gestur

Gambar 4.3 menampilkan potongan kode untuk pendeteksian gestur dari Leap Motion Controller. Di dalam potongan kode ini terlihat bahwa *\_pinchDetectorA* dengan *\_pinchDetectorB* adalah abstraksi dari gestur yang mengirimkan status apakah gestur sedang dilakukan atau tidak. Bila gestur dilakukan, maka translasi posisi dapat dilakukan, bila tidak maka akan diabaikan.

Dalam *game object ObjectToMove* terdapat komponen *sphere collider* yang berfungsi sebagai identifikasi untuk *TargetObject* dan memiliki sebuah *variable* yang memberikan *ID* apakah *object* memiliki *ID* yang sama dengan *TargetObject*. *TargetObject* juga memiliki sebuah *sphere collider* yang berfungsi sebagai pendeteksi untuk *ObjectToMove* yang harus bertemu dengan *sphere collider* dari *ObjectToMove*. *Sphere collider TargetObject* memiliki *script* untuk mendeteksi *ID* dari *ObjectToMove* yang bersentuhan, beserta rotasi dari *ObjectToMove*.

```

22 void OnTriggerEnter(Collider col) {
23     if (col.tag == "target") {
24         if(col.GetComponent<ObjectBehavior>().ID == shapeID) {
25             checkRotation(col.gameObject);
26
27             //For level making
28             Debug.Log("Hit!");
29             Debug.Log("X:");
30             Debug.Log(col.gameObject.transform.eulerAngles.x);
31             Debug.Log("Y:");
32             Debug.Log(col.gameObject.transform.eulerAngles.y);
33             Debug.Log("Z:");
34             Debug.Log(col.gameObject.transform.eulerAngles.z);
35         }
36     }
37 }

```

Gambar 4.4 Potongan Kode Pengecekan *ID ObjectToMove*

Gambar 4.4 menampilkan potongan kode untuk pengecekan *ID* dan rotasi dari *ObjectToMove*. Untuk setiap *TargetObject* juga disimpan sebuah *variable shapeID* yang akan digunakan untuk melakukan perbandingan dengan *ObjectToMove*. Setelah kedua *ID* cocok, dilakukan pemanggilan fungsi berikutnya *checkRotation*.

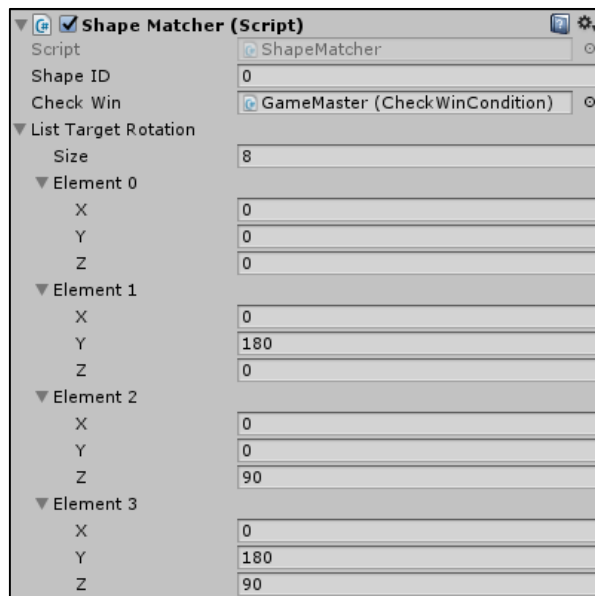
```

39
40     private float offset = 40.0f;
41     void checkRotation(GameObject target) {
42
43         float targetX = target.transform.eulerAngles.x;
44         float targetY = target.transform.eulerAngles.y;
45         float targetZ = target.transform.eulerAngles.z;
46         bool flag = false;
47
48         foreach (XYZSerializable xyz in listTargetRotation) {
49             float maxX = xyz.x + offset;
50             float minX = xyz.x - offset;
51
52             float maxY = xyz.y + offset;
53             float minY = xyz.y - offset;
54
55             float maxZ = xyz.z + offset;
56             float minZ = xyz.z - offset;
57
58             if (targetX + offset > 360) maxX += 360;
59             if (targetX - offset < 0) minX -= 360;
60
61             if (targetY + offset > 360) maxY += 360;
62             if (targetY - offset < 0) minY -= 360;
63
64             if (targetZ + offset > 360) maxZ += 360;
65             if (targetZ - offset < 0) minZ -= 360;
66
67             if (targetX < maxX && targetX > minX &&
68                 targetY < maxY && targetY > minY &&
69                 targetZ < maxZ && targetZ > minZ) {
70                 flag = true;
71             }
72         }

```

Gambar 4.5 Potongan Kode Pengecekan Rotasi *ObjectToMove*

Gambar 4.5 menampilkan potongan kode untuk pengecekan rotasi dari *ObjectToMove*. Fungsi sebelumnya akan menyimpan setiap posisi dari *ObjectToMove* sebagai *targetX*, *targetY*, dan *targetZ*. Setiap *variable* yang disimpan kemudian dibandingkan dengan *XYZSerializable* yang menyimpan seluruh kemungkinan rotasi dari *TargetObject* yang sesuai dengan permainan.



Gambar 4.6 Tampilan Pengisian Array Rotasi *TargetObject*

Gambar 4.6 menampilkan potongan dari tampilan *array* yang harus diisi sesuai kebutuhan permainan. *Size* menunjukkan berapa jumlah *array* yang dibutuhkan, dengan X, Y, dan Z adalah isi dari rotasi yang diinginkan.

```

73         if (flag) {
74
75             Debug.Log("Gotcha!");
76             _CheckWin.addScore();
77
78             target.transform.parent.GetComponent<BoxCollider>().enabled = false;
79             target.transform.parent.GetComponent<MovementBehaviour>().setUnmovable();
80             target.GetComponent<SphereCollider>().enabled = false;
81
82             GameObject.Find("MusicPlayer").GetComponent<GameMusicController>().playSFX();
83             GameObject.Find("GameMaster").GetComponent<GameMaster>().playFlash();
84
85             this.transform.parent.gameObject.SetActive(false);
86             Destroy(this.transform.parent.gameObject);
87         }

```

Gambar 4.7 Potongan Kode Yang Dijalankan Bila Pengecekan Berhasil

Gambar 4.7 menampilkan potongan kode yang dijalankan apabila pengecekan berhasil, hal yang dilakukan adalah menambah skor dengan fungsi *addScore*, kemudian *TargetObject* akan mematikan *BoxCollider*, *SphereCollider*, dan *MovementBehaviour* dari *ObjectToMove* sehingga *game object* tidak bergerak

dari posisi terakhirnya, kemudian memainkan suara dan kilat cahaya untuk memberikan tanda kepada pemain, dan terakhir mematikan *TargetObject* dan menghancurkannya.

```
24 public void addScore()
25 {
26     currentScore++;
27     checkHowManyShape();
28 }
29
30 void checkHowManyShape()
31 {
32     Debug.Log(currentScore);
33     if (howManyShape == currentScore)
34     {
35         winBox.SetBool("WinTheGame", true);
36         gameMusic.winBGMStart();
37         GameObject.Find("CameraController").GetComponent<CameraMovement>().winCameraMovement();
38         GameObject.Find("Number").GetComponent<Timer>().stopTimer();
39     }
40 }
```

Gambar 4.8 Potongan Kode Penambahan Skor dan Pengecekan Jumlah Skor

Gambar 4.8 menampilkan potongan kode untuk penambahan skor dan mengecek apakah skor sudah mencapai target jumlah benda yang ada di *level*. Fungsi *addScore* akan menambah jumlah skor atau total benda yang harus samakan dalam sebuah *level*. Setiap fungsi *addScore* dipanggil, fungsi *checkHowManyShape* dipanggil untuk langsung mengecek *currentScore* dengan *howManyShape*, bila sudah menyamai maka *UI* untuk *FINISH* dimainkan, beserta memutar lagu untuk *UI FINISH*, posisi kamera kemudian diubah ke rotasi 0 agar modul *Leap Motion UI Input* dapat berfungsi, dan memanggil fungsi untuk mematikan *Timer*.

```

38     public void stopTimer()
39     {
40         isPlaying = false;
41         StopCoroutine(corTime);
42         saveTime();
43     }
44
45     public void saveTime()
46     {
47         int[] scoreArray = PlayerPrefsX.GetIntArray("highScore",9999,16);
48
49         if (scoreArray[stageNum] > timeElapsed)
50         {
51             scoreArray[stageNum] = timeElapsed;
52             Debug.Log("Saved!");
53         }
54
55         PlayerPrefsX.SetIntArray("highScore",scoreArray);
56     }

```

Gambar 4.9 Potongan Kode Penghentian *Timer* dan Penyimpanan *highScore*

Gambar 4.9 menampilkan potongan kode untuk menghentikan *Timer* dan menyimpan *highScore*. Pada saat fungsi *stopTimer* dipanggil, *Timer* akan dihentikan, dan *Coroutine* akan dimatikan, kemudian fungsi *saveTime* akan dipanggil. Dalam fungsi *saveTime*, array untuk *highScore* akan dipanggil. Bila pemain sebelumnya belum memiliki array *highScore* yang tersimpan di *PlayerPrefs*, *ArrayPrefs2* akan membuatnya terlebih dahulu. Kemudian akan dilakukan pengecekan apakah array yang dipanggil memiliki *value* yang lebih kecil daripada *Timer*. Bila *value Timer* lebih kecil, maka array akan disimpan. Bila *value Timer* lebih besar, maka *value Timer* akan dibuang.

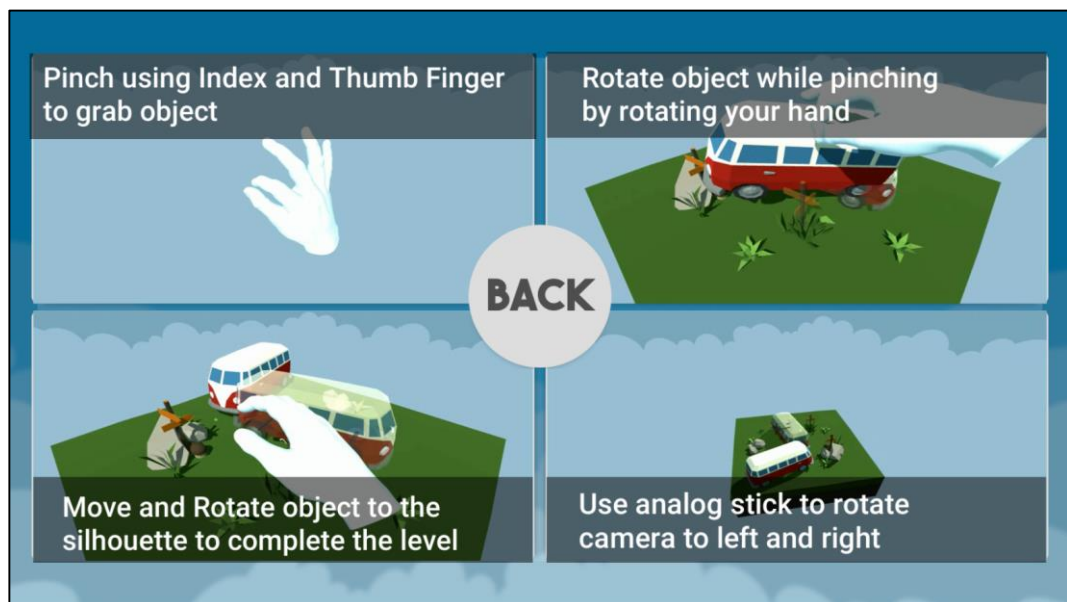
### 4.3 Hasil Implementasi

Berikut adalah hasil implementasi permainan yang memiliki tampilan pada Gambar 4.10.



Gambar 4.10 Tampilan *Menu Utama*

Pada tampilan *menu* utama permainan terdapat 3 opsi yang dapat dipilih, opsi pertama adalah “*How to Play*”, pada saat opsi ini ditekan maka tampilan cara bermain akan ditampilkan yang dapat dilihat pada Gambar 4.11.



Gambar 4.11 Tampilan “*How to Play*”

Opsi ke 2 adalah “*Start*”, pada saat opsi ini ditekan maka tampilan pemilihan tema yang dapat dilihat pada Gambar 4.12. akan ditampilkan.



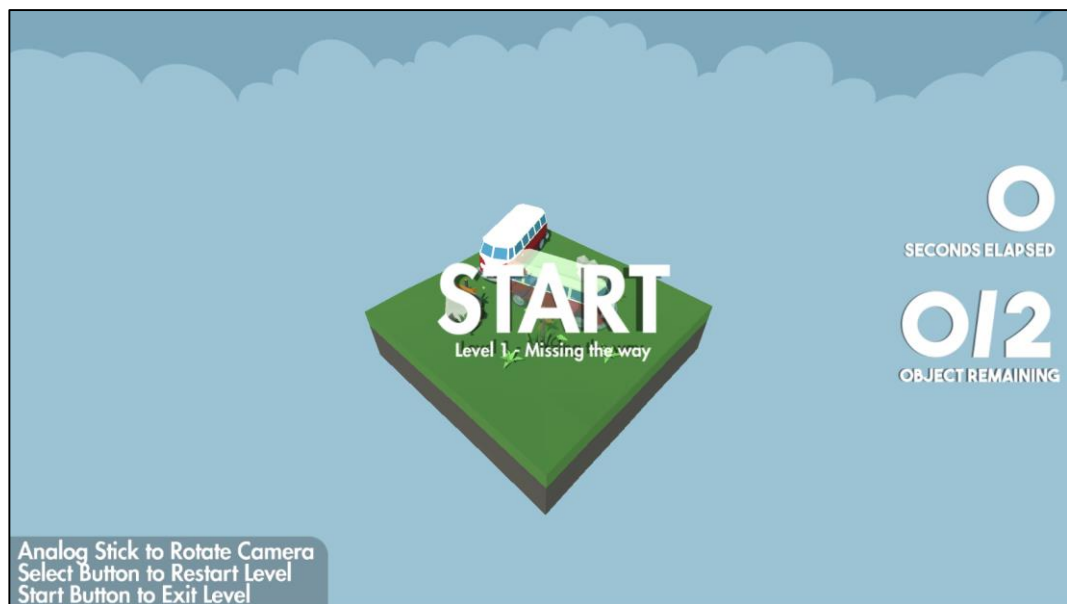
Gambar 4.12 Tampilan Pemilihan Tema

Memilih tema apapun akan membawa pemain pada tampilan selanjutnya yaitu pemilihan *level* yang dapat dilihat pada Gambar 4.13, pada tampilan ini ditampilkan gambaran *level* beserta hasil terbaik pemain disetiap *level* dalam bentuk detik.



Gambar 4.13 Tampilan Pemilihan *Level*





Gambar 4.14 Tampilan Permainan Dimulai

Memilih tema akan membawa pemain pada tampilan permainan yang dapat dilihat pada Gambar 4.14. Diawali dengan tampilan “*START*” untuk memulai permainan beserta *Timer* yang diletakkan pada sisi kanan layar.



Gambar 4.15 Tampilan Permainan Selesai

Selesai memainkan pemain akan ditampilkan “*FINISH*” yang dapat dilihat pada Gambar 4.15, beserta 3 tombol navigasi untuk mengulang *level*, melanjutkan

*level*, dan mengakhiri permainan. Menekan tombol pertama akan mengulang *level*, Menekan tombol kedua akan melanjutkan *level*, dan menekan tombol ketiga akan membawa pemain kembali ke *menu* utama.

Menekan opsi ketiga di *menu* utama akan menutup permainan.

#### **4.4 Pengujian**

Dalam penelitian ini dibutuhkan pengujian aplikasi guna mengetahui apakah aplikasi yang telah dibuat sesuai dengan kepuasan pengguna. Menurut Jakob Nielsen pengujian untuk pencarian *usability problem* cukup dilakukan ke 5 orang pengguna dan melakukan iterasi pengujian sebanyak selama memungkinkan.

Pengujian dilakukan dalam 3 iterasi. Untuk setiap iterasi terdapat 5 peserta uji coba yang akan memainkan permainan selama 10 menit, setiap peserta uji coba dimulai dengan menyimak *How to Play* untuk mempelajari cara memainkan permainan. Kemudian 10 menit setelah memainkan permainan, peserta uji coba diminta untuk mengisi kuesioner yang telah disediakan. Kuesioner yang diberikan berupa 19 soal PSSUQ yang dihitung menggunakan skala dengan 7 poin penilaian dimulai dari nilai 1 untuk sangat setuju sampai 7 untuk sangat tidak setuju. Data yang terkumpul akan diproses dan dihitung untuk mencari nilai rata-rata total sebagai hasil akhir dari setiap kategori, kemudian dihitung persentase dari nilai rata-rata total yang didapat berdasarkan 4 aspek faktor psikometri PSSUQ, yaitu *Overall*, *SysUse*, *InfoQual*, dan *IntQual*.

Kemudian dari hasil perhitungan dari setiap pertanyaan akan di hitung rata-ratanya berdasarkan aspek faktor psikometri dari PSSUQ, untuk aspek *Overall* akan dihitung rata-rata dari hasil perhitungan pertanyaan 1-19, *SysUse* akan dihitung

rata-rata dari hasil perhitungan pertanyaan 1-8, *InfoQual* akan dihitung rata-rata dari hasil perhitungan pertanyaan 9-15, dan *IntQual* akan dihitung rata-rata hasil perhitungan pertanyaan 16-18.

Untuk setiap iterasi pengujian, aplikasi akan dikaji ulang dan diperbaiki mengikuti saran dari peserta uji coba untuk kemudian dilakukan pengujian kembali dengan 5 peserta uji coba berbeda.

#### 4.4.1 Hasil Iterasi Pertama

Pada iterasi pertama, pengujian dilakukan oleh 5 orang pekerja *studio game* Toge Productions, para peserta uji coba berumur 20-30 tahun dan memiliki latar belakang pendidikan yang berbeda. Berikut adalah hasil rekapitulasi survei PSSUQ pada Tabel 4.1.

Tabel 4.1 Rekapitulasi Survei Iterasi Pertama

Pertanyaan ke-	Skala 1	Skala 2	Skala 3	Skala 4	Skala 5	Skala 6	Skala 7	Aspek
1	0	0	0	1	1	3	0	<i>SysUse</i>
2	0	0	1	2	0	2	0	
3	0	0	0	1	3	1	0	
4	0	0	0	0	3	1	1	
5	0	0	0	0	4	1	0	
6	0	0	0	1	2	2	0	
7	1	1	2	1	0	0	0	
8	0	0	1	2	1	0	1	
9	0	0	0	2	1	2	0	<i>InfoQual</i>
10	1	3	0	0	1	0	0	
11	0	2	1	1	0	1	0	
12	0	2	0	2	1	0	0	
13	0	2	1	1	1	0	0	
14	0	2	0	2	1	0	0	
15	1	2	1	1	0	0	0	
16	0	2	2	0	1	0	0	<i>IntQual</i>
17	0	2	1	2	0	0	0	
18	0	0	0	2	1	2	0	
19	0	0	1	2	1	1	0	<i>Overall</i>

Tabel 4.1 menunjukkan rekapitulasi 19 pertanyaan dari 5 peserta yang mengisi survei PSSUQ. Untuk setiap jawaban dari survei akan dilakukan perhitungan rata-rata berdasarkan aspek yang dapat dilihat pada Tabel 4.2.

Tabel 4.2 Tabel Hasil Analisis Data Pengujian Iterasi Pertama

Aspek Kategori	Persentasi Hasil Akhir (Pembulatan 2 Angka Dibelakang Koma)
<i>Overall</i>	56,54%
<i>System Usage</i>	46,07%
<i>Information Quality</i>	66,93%
<i>Interface Quality</i>	61,90%

Tabel 4.2 menunjukkan hasil rata-rata dari survei pada 5 orang peserta uji coba aplikasi. Berdasarkan kriteria intepretasi skor menggunakan Likert Scale, hasil dari *Overall*, dan *System Usage* mencapai kriteria cukup, hasil dari *Information Quality* dan *Interface Quality* mencapai kriteria kuat. Dari hasil ini peserta uji coba aplikasi memberikan saran untuk memperbaiki deteksi gestur, dan jarak *menu* agar lebih mudah dikendalikan.

#### 4.4.2 Hasil Iterasi Kedua

Pada iterasi kedua, pengujian dilakukan kepada 5 orang mahasiswa Universitas Multimedia Nusantara, para peserta uji coba berumur 22-23 tahun dan memiliki latar pendidikan yang berbeda. Berikut adalah hasil rekapitulasi survei PSSUQ pada Tabel 4.3.

Tabel 4.3 Rekapitulasi Survei Iterasi Kedua

Pertanyaan ke-	Skala 1	Skala 2	Skala 3	Skala 4	Skala 5	Skala 6	Skala 7	Aspek
1	1	0	1	1	0	1	1	<i>SysUse</i>
2	1	1	1	1	1	0	0	
3	0	1	1	1	2	0	0	



dari setiap peserta, serta memberikan petunjuk banyak objek yang harus diletakkan yang membantu pemain selama permainan berlangsung.

#### 4.4.3 Hasil Iterasi Ketiga

Pada iterasi ketiga, pengujian dilakukan oleh 5 orang mahasiswa Universitas Multimedia Nusantara, para peserta uji coba berumur 20-21 tahun dan memiliki latar pendidikan yang berbeda. Berikut adalah hasil rekapitulasi survei PSSUQ pada Tabel 4.5.

Tabel 4.5 Rekapitulasi Survei Iterasi Ketiga

Pertanyaan ke-	Skala 1	Skala 2	Skala 3	Skala 4	Skala 5	Skala 6	Skala 7	Aspek
1	0	2	3	0	0	0	0	<i>SysUse</i>
2	0	2	3	0	0	0	0	
3	1	2	0	2	0	0	0	
4	1	0	4	0	0	0	0	
5	1	1	2	1	0	0	0	
6	1	2	0	2	0	0	0	
7	2	2	1	0	0	0	0	
8	2	0	1	1	1	0	0	
9	0	0	1	2	0	2	0	<i>InfoQual</i>
10	1	1	2	1	0	0	0	
11	0	0	2	2	1	0	0	
12	1	0	3	1	0	0	0	
13	1	2	2	0	0	0	0	
14	1	2	2	0	0	0	0	
15	1	1	1	1	1	0	0	
16	2	2	1	0	0	0	0	<i>IntQual</i>
17	1	4	0	0	0	0	0	
18	2	1	2	0	0	0	0	<i>Overall</i>
19	2	3	0	0	0	0	0	

Tabel 4.5 menunjukkan rekapitulasi 19 pertanyaan dari 5 peserta yang mengisi survei PSSUQ. Untuk setiap jawaban dari survei akan dilakukan perhitungan rata-rata berdasarkan aspek yang dapat dilihat pada Tabel 4.6.

Tabel 4.6 Tabel Hasil Analisis Data Pengujian Iterasi Ketiga

Aspek Kategori	Persentasi Hasil Akhir (Pembulatan 2 Angka Dibelakang Koma)
<i>Overall</i>	77,74%
<i>System Usage</i>	78,21%
<i>Information Quality</i>	71,02%
<i>Interface Quality</i>	87,61%

Tabel 4.6 menunjukkan hasil rata-rata dari survei pada 5 orang peserta uji coba aplikasi. Berdasarkan kriteria intepretasi skor menggunakan Likert Scale, hasil dari *Overall*, *System Usage*, dan *Information Quality* mencapai kriteria kuat, hasil dari *Interface Quality* mencapai kriteria sangat kuat. Dari hasil ini peserta uji coba aplikasi memberikan saran untuk mengoptimasi deteksi pergerakan tangan, serta meningkatkan deteksi balasan *menu* pada saat *level* selesai dimainkan.

## BAB V

### KESIMPULAN DAN SARAN

#### 5.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, ditarik beberapa kesimpulan yang dapat dijabarkan sebagai berikut.

1. Aplikasi *puzzle game* “Match Shape” berbasis Leap Motion Controller telah berhasil dirancang dan dibangun. Pengujian berdasarkan PSSUQ dan metode pengujian Jakob Nielsen telah mendapatkan hasil akhir berupa 78,21% untuk *System Usage*, 71,02% untuk *Information Quality*, 87,61% untuk *Interface Quality*, dan secara *Overall* sebesar 77,74%.
2. Pengembangan aplikasi *puzzle game* dengan menggunakan metode pengujian Jakob Nielsen dapat memberikan peningkatan kualitas pada seluruh aspek yang diuji. Dapat dilihat pada iterasi pertama dalam pengujian dan iterasi kedua dalam pengujian *System Usage* terjadi peningkatan sebesar 18,21% dari 46,07% menjadi 64,28%, pengujian *Information Quality* terjadi peningkatan sebesar 5,72% dari 66,93% menjadi 72,65%, pengujian *Interface Quality* terjadi peningkatan sebesar 10,48% dari 61,90% menjadi 72,38%, dan secara *Overall* terjadi peningkatan sebesar 12% dari 56,57% menjadi 68,57%. Pada iterasi kedua dalam pengujian dan iterasi ketiga dalam pengujian *System Usage* terjadi peningkatan sebesar 13,93% dari 64,28% menjadi 78,21%, pengujian *Information Quality* terjadi penurunan sebesar 1,63% dari 72,65% menjadi 71,02%, pengujian *Interface Quality* terjadi peningkatan sebesar 15,23% dari 72,38% menjadi 87,61%, dan



secara *Overall* terjadi peningkatan sebesar 9,17% dari 68,57% menjadi 77,74%.

## **5.2 Saran**

Berdasarkan penelitian yang telah dilaksanakan, beberapa hal yang dapat diterapkan untuk penelitian serupa berikutnya adalah sebagai berikut.

1. Melakukan implementasi teknologi *Virtual Reality* dan/atau teknologi *Augmented Reality* untuk meningkatkan kesan yang lebih alami untuk pengguna.
2. Menggunakan teknologi deteksi gestur lainnya seperti Oculus Touch, atau Vive Controller untuk meningkatkan akurasi dan kenyamanan dalam penggunaan aplikasi.
3. Melakukan ekspansi permainan dalam segi desain maupun jumlah *level* yang dapat ditambahkan.

## DAFTAR PUSTAKA

- Chang, E. 2012. New Ways of Accessing Information Spaces Using 3D Multitouch Tables. *Proceedings of the International Conference on Cyberworlds (CW)*, hal. 144-150.
- Fullerton, T. 2008. *Game Design Workshop a Playcentric Approach to Creating Innovative Games*. 2<sup>nd</sup> Edition. Morgan Kaufmann, San Fransisco.
- Glonek, G., Pietruszka, M. 2012. Natural User Interfaces (NUI): Review. *Journal of Applied Computer Science* 20(2), hal. 27-45.
- Joshi, A., Kale, S., Chandel, S. dan Pal, D.K. 2015. *Likert Scale: Explored and Explained*. *British Journal of Applied Science & Technology* 7(4): 396-403.
- Leap Motion, 2014. *Gestures — Leap Motion C# SDK v2.3*. documentation. Tersedia dalam: [https://developer.leapmotion.com/documentation/csharp/devguide/Leap\\_Gestures.html](https://developer.leapmotion.com/documentation/csharp/devguide/Leap_Gestures.html) [diakses 30 Maret 2016]
- Leap Motion, 2014. *Developer Marketing Assets / Leap Motion Developers*. Tersedia dalam: <https://developer.leapmotion.com/downloads/developer-marketing-assets> [diakses 30 Maret 2016]
- Lehrman, Robert A. Video game nation: Why so many play – CSMonitor.com. Tersedia dalam: <http://www.csmonitor.com/USA/Society/2012/0318/Video-game-nation-Why-so-many-play> [diakses 3 November 2016]
- Lewis, J.R. 1995. IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use. *International Journal of Human-Computer Interaction* 7(1), hal. 57-78.
- Lewis, J.R. 2002. Psychometric Evaluation of the PSSUQ Using Data from Five Years of Usability Studies. *International Journal of Human-Computer Interaction* 14(3-4), hal. 463-488.
- McCartney, R., Yuan, J., Bischof, H.-P. 2015. Gesture Recognition with the Leap Motion Controller. Dalam *Int'l Conf. IP, Comp. Vision, and Pattern Recognition 2015*, hal. 3-9.
- Morley, R. Video Games: The New American Pastime – theTrumpet.com. Tersedia dalam: <https://www.thetrumpet.com/article/9225.3> [diakses 3 November 2016]

- Nielsen. J., Laundauer, T.K. 1993. A Mathematical Model of the Finding of Usability Problems. Dalam *INTERCHI'93 Conference*, hal. 206-213.
- Nielsen, J. 2000. *Why You Only Need to Test with 5 Users*. Tersedia dalam: <https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/> [diakses 30 Maret 2016].
- Oei, A.C., Patterson, M.D., 2014. Playing a puzzle video game with changing requirements improves executive functions. *Computers in Human Behaviour* 37, hal. 216-228.
- Russoniello, C.V., O'Brien, K., Parks, J.M. 2009. EEG, HRV and Psychological Correlates while Playing Bejeweled II: A Randomized Controller Study. *Annual Review of Cybertherapy and Telemedicine* 7, hal. 189-192.
- Spence, I., Feng, J. 2010. Video games and spatial cognition. *Review of General Psychology* 14(2), hal 92-104.
- Wigdor, D., Wixon, D. 2011. *Brave NUI World Designing Natural User Interfaces for Touch and Gesture*. Morgan Kaufmann, San Fransisco.
- Wolf, M.J.P., 2001. *The Medium of the VIDEO GAME*. University of Texas Press, Austin.

## **LAMPIRAN 1**

## LAMPIRAN 2

### BIOGRAFI PENULIS

#### A. DATA PRIBADI

Nama Lengkap	: Albert Van Otto
Jenis Kelamin	: Laki-Laki
Tempat, Tanggal Lahir	: Jakarta, 14 Januari 1993
Kewarganegaraan	: Indonesia
Alamat	: Taman Permata Sektor 7 Jalan Permata Sakti 4 Blok E6 No 16 Lippo Karawaci – Tangerang 15810
E-Mail	: vanotto@insertlogo.here

#### B. PENDIDIKAN

2011-Sekarang	Universitas Multimedia Nusantara
2009-2011	SMK PGRI 1 Tangerang
2007-2009	SMP Atisa Dipamkara Tangerang
2001-2006	SMP Atisa Dipamkara Tangerang

#### C. PENGALAMAN KERJA & ORGANISASI

Jun - Sept 2013	<b>Game Designer</b> , Gambeng Games, Tangerang – mendesain game untuk <i>smartphone</i> berjudul “Prototype”.
Apr – Jul 2014	<b>Web Development</b> , Masterpiece Magazine, Tangerang – mengatur dan membangun fitur baru untuk situs.
Sept 2013 – Sept 2014	<b>Vice President</b> , Game Development Club UMN, Tangerang – mengatur dan mewakilkan ketua dalam UKM.

Apr 2015 – Sekarang	<b><i>Motion Graphic Artist</i></b> , Studio Muné, Tangerang – mengerjakan <i>motion graphic</i> untuk klien yang dituju.
Feb 2016 – Sekarang	<b><i>Video Producer</i></b> , Dapper Penguin Studio, Spanyol – mengerjakan pembuatan <i>video</i> untuk <i>video game</i> “Project Automata”

#### **D. PRESTASI**

April 2013	<b><i>1st Place XL Gamehack 2013</i></b> , Juara 1 dalam kategori public di XL Gamehack 2013
September 2013	<b><i>1st Place Indonesia Game Show Game Developer Award 2013</i></b> , Juara 1 dalam kategori amatir di <i>Indonesia Game Show Game Developer Award</i> untuk <i>Best game for Smartphone/tablet</i>