TUGAS AKHIR - EF234801

PENGEMBANGAN INTERAKSI DAN APLIKASI REALITAS X UNTUK PERENCANAAN OPERASI PEMOTONGAN MANDIBULA

Wahyu Tri Saputro

NRP 5025201217

Dosen Pembimbing

Hadziq Fabroyir, S.Kom., Ph.D.

NIP 198602272019031006

Dosen Ko-Pembimbing

Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.

NIP 197512202001122002

Program Studi S-1 Teknik Informatika

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2024

(halaman ini sengaja dikosongkan)

TUGAS AKHIR – EF234801

PENGEMBANGAN INTERAKSI DAN APLIKASI REALITAS X UNTUK PERENCANAAN OPERASI PEMOTONGAN MANDIBULA

Wahyu Tri Saputro

NRP 5025201217

Dosen Pembimbing

Hadziq Fabroyir, S.Kom., Ph.D.

NIP 198602272019031006

Dosen Ko-pembimbing

Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.

NIP 197512202001122002

Program Studi S-1 Teknik Informatika

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2024

(halaman ini sengaja dikosongkan)

FINAL PROJECT – EF234801

DEVELOPMENT OF X REALITY INTERACTION AND APPLICATION FOR MANDIBULAR CUTTING SURGERY PLANNING

Wahyu Tri Saputro
NRP 5025201217

Advisor
Hadziq Fabroyir, S.Kom., Ph.D.
NIP 198602272019031006

Co-advisor
Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.
NIP 197512202001122002

Study Program Bachelor of Informatics
Department of Informatics
Faculty of Intelligent Electrical and Informatics Technology
Institut Teknologi Sepuluh Nopember
Surabaya
2024

(halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

PENGEMBANGAN INTERAKSI DAN APLIKASI REALITAS X UNTUK PERENCANAAN OPERASI PEMOTONGAN MANDIBULA

TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat
Memperoleh gelar Sarjana Komputer pada
Program Studi S-1 Teknik Informatika
Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember

Oleh : **WAHYU TRI SAPUTRO**
NRP. 5025201217

Disetujui oleh Tim Penguji Tugas Akhir:

1. Hadziq Fabroyir, S.Kom., Ph.D. — Pembimbing
2. Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom — Ko-pembimbing
3. Siska Arifiani, S.Kom., M.Kom. — Penguji
4. Dr.Eng. Darlis Herumurti, S.Kom., M.Kom. — Penguji

SURABAYA
Juli, 2024

(halaman ini sengaja dikosongkan)

APPROVAL SHEET

DEVELOPMENT OF X REALITY INTERACTION AND APPLICATION FOR MANDIBULAR CUTTING SURGERY PLANNING

FINAL PROJECT

Submitted to fulfill one of the requirements
for obtaining a degree Bachelor of Computer at
Undergraduate Study Program of Informatics
Department of Informatics
Faculty of Intelligent Electrical and Informatics Technology
Institut Teknologi Sepuluh Nopember

By: **WAHYU TRI SAPUTRO**
NRP. 5025201217

Approved by Final Project Examiner Team:

1. Hadziq Fabroyir, S.Kom., Ph.D. Advisor
2. Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom. Co-Advisor
3. Siska Arifiani, S.Kom., M.Kom. Examiner
4. Dr.Eng. Darlis Herumurti, S.Kom., M.Kom. Examiner

SURABAYA
July, 2024

(halaman ini sengaja dikosongkan)

PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini:

Nama mahasiswa / NRP : Wahyu Tri Saputro / 5025201217
Departemen : Teknik Informatika
Dosen Pembimbing / NIP : Hadziq Fabroyir, S.Kom., Ph.D. / 198602272019031006
Dosen Ko-pembimbing / NIP : Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom. / 197512202001122002

Dengan ini menyatakan bahwa Tugas Akhir dengan judul "Pengembangan Interaksi dan Aplikasi Realitas X Untuk Perencanaan Operasi Pemotongan Mandibula" adalah hasil karya sendiri, bersifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah. Bilamana di kemudian hari dietemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima saksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Surabaya, 10 Juli 2024

Mahasiswa

Wahyu Tri Saputro
NRP. 5025201217

Mengetahui
Dosen Pembimbing

Hadziq Fabroyir, S.Kom., Ph.D.

NIP. 198602272019031006

Mengetahui
Dosen Ko-pembimbing

Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.
NIP. 197512202001122002

(halaman ini sengaja dikosongkan)

STATEMENT OF ORIGINALITY

The undersigned below:

Name of student / NRP	: Wahyu Tri Saputro / 5025201217
Department	: Informatics
Advisor / NIP	: Hadziq Fabroyir, S.Kom., Ph.D. / 198602272019031006
Co-advisor / NIP	: Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom. / 197512202001122002

Hereby declare that Final Project with the title of "Development of X Reality Interaction and Application for Mandibular Cutting Surgery Planning" is the result of my own work, is original, and is written by following the rules of scientific writing. If in the future there is a discrepancy with this statement, then I am willing to accept sanctions in accordance with the provisions that apply at Institut Teknologi Sepuluh Nopember.

Surabaya, July 10 2024

Student

Wahyu Tri Saputro
NRP. 5025201217

Acknowledge
Advisor

Hadziq Fabroyir, S.Kom., Ph.D.

NIP. 198602272019031006

Acknowledge
Co-advisor

Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.
NIP. 197512202001122002

(halaman ini sengaja dikosongkan)

PENGEMBANGAN INTERAKSI DAN APLIKASI REALITAS X UNTUK PERENCANAAN OPERASI PEMOTONGAN MANDIBULA

Nama Mahasiswa / NRP	**:**	**Wahyu Tri Saputro / 5025201217**
Departemen	**:**	**Teknik Informatika, FTEIC-ITS**
Dosen Pembimbing	**:**	**Hadziq Fabroyir, S.Kom., Ph.D.**
		Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.

Abstrak

Bedah rahang adalah salah satu prosedur medis yang paling penting dalam pengobatan berbagai penyakit, seperti rekonstruksi rahang, pemulihan fungsi rahang, dan perbaikan penampilan wajah. Untuk menjalani prosedur bedah rahang, terutama yang melibatkan transplantasi fibula atau tulang betis, diperlukan persiapan yang cermat. Sebagai persiapan, tahap operasi bedah rahang biasanya memanfaatkan model cetak 3D atau manekin anatomi manusia. Namun, metode ini menghadapi masalah seperti biaya tinggi, waktu yang lama, dan tidak dapat diakses. Perencanaan bedah rahang dapat dilakukan dengan teknologi Realitas X (XR), yang lebih hemat biaya dan memungkinkan pengalaman bedah mendalam dalam dunia virtual. Penggunaan XR dalam perencanaan bedah rahang memungkinkan ahli bedah memanipulasi objek secara rinci dalam ruang visualisasi tiga dimensi. Dalam kasus ini, aplikasi XR membantu menentukan tempat, ukuran, dan jarak prosedur pemotongan tulang, sehingga mengurangi biaya dan waktu persiapan. Skenario pengujian aplikasi adalah partisipan akan mencoba aplikasi dan diberikan tugas-tugas yang harus diselesaikan, kemudian partisipan mengisi kuisioner untuk menilai aplikasi. Semua orang yang berpartisipasi diminta untuk mencoba tiga interaksi yang berbeda, tetapi masing-masing memiliki tujuan yang sama: melakukan proses pemotongan tulang mandibula. Jika aplikasi termasuk dalam kategori tidak dapat diterima, marjinal, atau dapat diterima, pengujian System Usability Scale akan memberikan skor nilai untuk aplikasi tersebut. Pengujian performa akan membandingkan interaksi mana yang paling cepat diselesaikan dan apakah ada perbedaan signifikan pada setiap interaksi. Mereka berharap bahwa penggunaan aplikasi XR akan sangat membantu ahli bedah merencanakan prosedur bedah rahang dengan baik, sekaligus mengurangi waktu dan biaya.

Kata kunci: Realitas X (XR), Kedokteran, Unity, Bedah Rahang.

(halaman ini sengaja dikosongkan)

DEVELOPMENT OF X REALITY INTERACTION AND APPLICATION FOR MANDIBULAR CUTTING SURGERY PLANNING

Student Name / NRP	**:**	**Wahyu Tri Saputro / 5025201217**
Department	**:**	**Informatics, FTEIC-ITS**
Advisor	**:**	**Hadziq Fabroyir, S.Kom., Ph.D.**
		Prof. Dr.Eng. Chastine Fatichah, S.Kom., M.Kom.

Abstract

Jaw surgery is one of the most important medical procedures in the treatment of various diseases, such as jaw reconstruction, restoration of jaw function and improvement of facial appearance. Undergoing a jaw surgery procedure, especially one that involves a fibula or shank bone transplant, requires careful preparation. In preparation, the operative stage of jaw surgery usually utilizes 3D printed models or human anatomy mannequins. However, this method faces problems such as high cost, long time and inaccessibility. Jaw surgery planning can be done with X Reality (XR) technology, which is more cost-effective and allows for an immersive surgical experience in a virtual world. The use of XR in jaw surgery planning allows surgeons to manipulate objects in detail in a three-dimensional visualization space. In this case, the XR application helps determine the place, size, and distance of the bone cutting procedure, thereby reducing costs and preparation time. The app testing scenario is that participants will try the app and be given tasks to complete, then participants fill out a questionnaire to rate the app. All participating people were asked to try three different interactions, but each had the same goal: to perform the mandibular bone cutting process. If the application falls into the unacceptable, marginal, or acceptable category, the System Usability Scale test will provide a value score for the application. Performance testing will compare which interaction is the fastest to complete and whether there is a significant difference in each interaction. They hope that the use of XR applications will greatly help surgeons plan jaw surgical procedures well, while reducing time and costs.

Kata kunci: Extended Reality (XR), Medical, Unity, Jaw Surgery.

(halaman ini sengaja dikosongkan)

KATA PENGANTAR

Puji syukur kepada Allah SWT atas segala karunia dan rahmat-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "PENGEMBANGAN APLIKASI REALITAS VIRTUAL UNTUK PERENCANAAN BEDAH RAHANG". Penulis menerima banyak dukungan dan bantuan dalam pelaksanaan dan pembuatan Tugas Akhir ini, Tanpa mengurangi rasa hormat penulis ingin mengucapkan terima kasih sebesar-besarnya kepada:

1. Allah SWT yang telah melancarkan dan memudahkan penulis dalam menyelesaikan Tugas Akhir ini.
2. Nabi Muhammad SAW yang telah menjadi teladan kehidupan selama penulis hidup.
3. Keluarga penulis (Ayah, Ibu dan kedua kakak) yang selalu memberikan dukungan berupa doa, semangat dan fasilitas sehingga penulis dapat menyelesaikan Tugas Akhir ini.
4. Bapak Hadziq Fabroyir, S.Kom., Ph.D. yang telah menjadi Dosen Pembimbing penulis untuk membimbing, memberikan nasihat dan memberikan saran kepada penulis selama pengerjaan Tugas Akhir.
5. Ibu Dr.Eng. Chastine Fatichah, S.Kom., M.Kom. yang telah menjadi Dosen Ko-Pembimbing penulis untuk membimbing, memberikan nasihat dan memberikan saran kepada penulis selama pengerjaan Tugas Akhir serta selaku kepala Departemen Teknik Informatika ITS.
6. Teman-teman dari Lab GIGa, Lab Alpro, dan teman-teman yang lain yang tidak bisa disebutkan satu – satu, yang selalu mendukung dan menemani penulis selama pengerjaan Tugas Akhir.
7. Muhammad Rafi Insan Fillah yang membantu penulis dalam menyelesaikan aplikasi dan memberikan masukan dan solusi atas masalah yang penulis temukan ketika mengembangkan aplikasi.
8. Untuk orang-orang yang telah membaca buku Tugas Akhir ini.

Penulis telah memberikan usaha terbaik dalam menyusun dan menyelesaikan Tugas Akhir ini. Penulis memohon maaf apabila terdapat kesalahan ataupun kekurangan yang terjadi atas ketidaksengajaan atau kekeliruan yang dilakukan. Penulis akan menerima kritik dan saran yang dimiliki oleh pembaca sebagai evaluasi kedepannya. Semoga Tugas Akhir ini dapat bermanfaat untuk penulis sendiri dan orang lain yang membacanya.

(halaman ini sengaja dikosongkan)

DAFTAR ISI

(halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

DAFTAR TABEL

(halaman ini sengaja dikosongkan)

DAFTAR KODE SUMBER

(halaman ini sengaja dikosongkan)

DAFTAR RUMUS

(halaman ini sengaja dikosongkan)

DAFTAR SINGKATAN

AR	Augmented Reality
AV	Augmented Virtuality
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
CT	Computerized Tomography
DICOM	Digital Imaging and Communications in Medicine
DSP	Digital Surgery Planning
FFF	Fibula Free Flap
HMD	Head-Mounted Display
MR	Mixed Reality
MRI	Magnetic Resonance Imaging
SDK	Software Development Kit
STL	Standard Triangle Language
SUS	System Usability Scale
UI	User Interface
USB	Universal Serial Bus
VR	Virtual Reality
XR	Extended Reality

(halaman ini sengaja dikosongkan)

BAB 1
PENDAHULUAN

1.1 Latar belakang

Operasi bedah rahang merupakan salah satu prosedur medis dalam penanganan berbagai macam kondisi mulai dari rekonstruksi rahang, pemulihan fungsi rahang, hingga perbaikan estetika wajah. Prosedur ini dilakukan dengan cara mengambil fibula atau tulang betis yang kemudian ditransplantasikan ke area rahang atau mandibula yang perlu direkonstruksi. Namun, tingkat ketelitian yang diperlukan dalam proses ini menuntut persiapan yang teliti sebelum operasi bedah rahang dilakukan.

Umumnya dalam dunia medis, persiapan untuk tahap operasi melibatkan penggunaan simulasi dengan objek cetak 3D atau manekin yang menyerupai anatomi manusia. Namun, pendekatan ini memiliki sejumlah kelemahan, termasuk biaya tinggi, waktu yang lama, dan keterbatasan aksesibilitas. Pembuatan model tersebut memerlukan alat canggih seperti CT (*Computerized Tomography*) scan atau MRI (*Magnetic Resonance Imaging*) untuk menciptakan replika yang akurat dari struktur tulang pasien (Li *et al*., 2019).

Demi mengatasi biaya dan waktu yang besar, diperlukan alternatif persiapan bedah rahang yang lebih efisien. Salah satu solusi alternatif adalah pemanfaatan teknologi VR (*Virtual Reality*), yang memungkinkan interaksi pengguna dengan lingkungan virtual. VR merupakan teknologi simulasi antarmuka antara manusia dan komputer yang menghasilkan lingkungan virtual tiga dimensi. Dengan VR, komputer mampu menciptakan pengalaman sensorik visual, pendengaran, dan sentuhan yang realistis (Li *et al*., 2019). Pengguna akan menggunakan sebuah HMD (*Head-Mounted Display*) berupa VR Headset untuk masuk ke dalam lingkungan virtual tiga dimensi dan berinteraksi di dalamnya. Seiring dengan berkembangnya tenologi VR Headset yang semakin canggih dengan aplikasi-aplikasi interaktif di dalamnya membuat popularitas VR semakin tinggi. Namun tidak hanya VR, teknologi tersebut juga memiliki tingkatan selanjutnya yaitu XR (*Extended Reality*). XR didefinisikan sebagai spektrum pengalaman realitas virtual dan berimbuh, yang menggabungkan dunia fisik dan virtual untuk menciptakan lingkungan yang menarik dan imersif, di mana pengguna dapat berinteraksi dengan elemen-elemen yang dibuat oleh komputer secara *real-time*.

Dalam konteks perencanaan bedah rahang, pengembangan aplikasi XR menjadi solusi yang efektif. Dengan bantuan XR, para ahli bedah mendapatkan pengalaman visual tiga dimensi yang memungkinkan mereka memvisualisasikan dan memanipulasi objek (Manzie *et al*., 2023). Dalam kasus sebelumnya, aplikasi serupa—jenis VR-DSP (*Digital Surgery Planning*)—telah dikembangkan pada platform Neos VR, yang tersedia di Steam. Aplikasi menyimulasikan pengangkatan kerusakan pada mandibula bagian kiri dalam lima tahapan. Setiap peserta memiliki akses ke pendamping fisik dan virtual selama bagian praktikum untuk memberikan saran dan menjawab pertanyaan. Fibula kanan pasien digunakan untuk merekonstruksi cacat ablatif. Praktik dilakukan secara duduk, tetapi peserta dapat berdiri untuk mendapatkan pemandangan yang lebih baik (Manzie *et al*., 2023).

Aplikasi yang akan dibuat akan membantu ahli bedah menentukan lokasi, ukuran, dan jarak yang diperlukan dalam prosedur pemotongan tulang. Aplikasi yang akan dibuat akan menyimulasikan pengangkatan mandibula namun dengan perbedaan pada interaksi pengangkatan yaitu dengan menggunakan *hand tracking* pada VR yang fleksibilitas dan aksesibilitas yang ditawarkan oleh penggunaan VR Headset menjadikan teknologi ini sebagai

solusi yang efisien dan potensial dalam mengurangi biaya dan waktu yang diperlukan dalam persiapan bedah rahang. Tidak hanya sekedar VR saja, namun aplikasi ini juga akan mengimplementasikan XR agar untuk melebur dunia fisik dan virtual agar menjadikan pengalaman pengguna menjadi lebih imersif. Dengan digunakannya teknologi XR ini, diharapkan dapat membantu ahli bedah untuk menyiapkan prosedur bedah rahang dan dapat menekan biaya besar dan waktu panjang dalam proses persiapan bedah rahang.

1.2 Rumusan Permasalahan

Rumusan masalah yang diajukan dalam Tugas Akhir ini adalah sebagai berikut.

1. Bagaimana perancangan aplikasi dan implementasi XR yang dapat digunakan untuk mengatasi keterbatasan aksesibilitas, biaya tinggi, dan waktu yang lama dalam proses persiapan bedah rahang?
2. Bagaimana evaluasi penggunaan dan pemahaman proses bedah rahang sebelum dan sesudah menggunakan aplikasi persiapan bedah rahang berbasis XR?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan dalam perancangannya, di antaranya sebagai berikut.

1. Pengembangan awal yang berfokus pada pengembangan XR.
2. Perancangan berfokus kepada pembuatan aplikasi saja tidak termasuk pembuatan aset-aset aplikasi, sehingga pengembangan akan menggunakan aset-aset model, UI (*User Interface*) gratis/berbayar yang dapat digunakan secara legal yang telah tersedia di Unity Asset Store atau internet.
3. Pengembangan aplikasi XR memanfaatkan *package* resmi Oculus Integration SDK yang telah tersedia dalam Unity Registry.
4. Bagian tulang yang dimanfaatkan dalam pengembangan aplikasi adalah tulang tengkorak.

1.4 Tujuan

Tujuan yang diajukan dalam menyelesaikan rumusan masalah yang diajukan dalam Tugas Akhir ini adalah sebagai berikut.

1. Mengatasi keterbatasan aksesibilitas, biaya tinggi, dan waktu yang lama dalam proses persiapan bedah rahang.
2. Mengetahui penggunaan dan pemahaman proses bedah rahang sebelum dan sesudah menggunakan aplikasi persiapan bedah rahang berbasis XR.

1.5 Manfaat

Tugas Akhir ini diharapkan dapat membantu ahli bedah untuk menyiapkan prosedur bedah rahang dan dapat menekan biaya besar dan waktu panjang dalam proses persiapan bedah rahang. Pengalaman imersif saat menggunakan aplikasi XR diharapkan mampu membantu ahli bedah untuk memvisualisasikan proses bedah rahang.

BAB 2
TINJAUAN PUSTAKA

2.1 Penelitian Terkait

2.1.1. Combined Application of Virtual Surgery and 3D Printing Technology in Postoperative Reconstruction of Head and Neck Cancers

Chao Li, Yongchong Cai, Wei Wang, Yan Sun, Guojun Li, Amy L. Dimachkieh, Weidong Tian, dan Ronghao Sun telah menghasilkan sebuah laporan studi berdasarkan lima kasus pasien yang menjalani rekonstruksi kepala dan leher akibat tumor kanker dengan menggunakan tekologi bedah virtual dan pencetakan 3D seperti VR, CAD (*Computer Aided Design*), dan CAM (*Computer Aided Manufacturing*). Mereka menjelaskan bahwa teknologi tersebut memiliki potensi untuk meningkatkan akurasi, efisiensi, dan fungsi dari hasil prosedur bedah. Metode penelitian yang digunakan adalah analisis literatur serta analisis konseptual yang bertujuan untuk menjelaskan dan mendiskusikan penerapan teknologi bedah virtual dan pencetakan 3D berdasarkan kasus-kasus yang mereka teliti. Hasil dari penelitian ini menyimpulkan bahwa pendekatan menggunakan bantuan komputer untuk rekonstruksi kepala dan leher secara personal dapat membantu dalam memvisualisasikan dengan lebih jelas anatomi tumor, melakukan rekonstruksi pada cacat tulang dan jaringan lunak yang kompleks, serta menentukan lesi vaskular. Studi ini mendukung kajian tentang rekonstruksi rahang dan dapat menjadi inspirasi untuk Tugas Akhir ini.

2.1.2. Virtual Reality Digital Surgical Planning for Jaw Reconstruction: A Usability Study

Timothy Manzie, Hamish MacDougall, Kai Cheng, Rebecca Venchiarutti, Richard Fox, Ashleigh Sharman, Emma Charters, Doruk Seyfi, Masako Dunn, Payal Mukherjee, dan Jonathan Clark telah melakukan sebuah studi yang bertujuan untuk mengembangkan sebuah platform DSP (*Digital Surgery Planning*) menggunakan teknologi VR. Platform VR-DSP ini difokuskan pada perencanaan ablasi dan rekonstruksi patologi pada area kepala dan leher. Dalam uji coba prospektif, para peneliti melibatkan ahli bedah rekonstruksi sebagai partisipan. Para peserta melakukan simulasi sesi perencanaan menggunakan VR-DSP. Selain itu, mereka diminta untuk mengisi kuesioner sebelum dan sesudah menggunakan VR-DSP, serta rekaman audio yang digunakan untuk analisis kualitatif. Studi ini menjadi inspirasi pengembangan aplikasi VR untuk topik Tugas Akhir ini.

2.1.3. Development and Application of a Surgical Process Simulation System Using VR Technology

Lang Zhou dan Reika Sato telah melakukan sebuah simulasi pelatihan operasi bedah dengan memanfaatkan teknologi VR sebagai pengganti metode pelatihan tradisional yang melibatkan observasi langsung. Penulis berpendapat bahwa pelatihan secara tradisional memiliki kelemahan, di mana tidak semua aspek anatomi tubuh dapat terlihat dengan jelas, sehingga mempersulit mahasiswa dalam memahami detail prosedur operasi bedah. Oleh karena itu, mereka mengusulkan penggunaan VR sebagai alternatif pelatihan untuk prosedur operasi, karena VR memungkinkan interaksi dan demonstrasi secara *real-time*. Hasilnya menunjukkan peningkatan aksesibilitas yang dapat memfasilitasi pemahaman yang lebih baik terhadap proses operasi. Studi ini menjadi inspirasi penerapan VR untuk topik Tugas Akhir ini.

2.1.4. Augmented and Virtual Reality in Surgery

Mathilde R. Desselle, Ross A. Brown, Allan R. James, Mark J. Midwinter, Sean K. Powell, dan Maria A. Woodruff telah melakukan studi terkait kemunculan teknologi AR dan VR dalam

konteks prosedur bedah, seperti pelatihan dan perencanaan. Penulis mengulas bagaimana AR dan VR memiliki potensi untuk meningkatkan integrasi, visualisasi, interaksivitas, pertukaran informasi kesehatan, peningkatan kinerja, serta keamanan dalam prosedur operasi bedah. Selain itu, teknologi ini juga memungkinkan konsultasi dan kolaborasi jarak jauh dalam bidang medis. Studi ini menjadi inspirasi untuk topik Tugas Akhir ini.

2.2 Dasar Teori

2.2.1. Bedah Rahang

Kedokteran, sebagai ilmu dan praktik dalam melakukan diagnosis, terapi, dan pencegahan serta pengobatan penyakit, meliputi berbagai praktik kesehatan dan perawatan untuk mempertahankan dan memulihkan kesehatan. Menurut Kamus Besar Bahasa Indonesia, kedokteran adalah segala sesuatu yang berhubungan dengan dokter atau pengobatan penyakit. Dalam konteks kedokteran kontemporer, ilmu biomedis, penelitian biomedis, genetika, dan teknologi medis digunakan untuk mendiagnosis, mengobati, dan mencegah cedera dan penyakit melalui obat-obatan atau prosedur operasi bedah, serta melalui terapi yang beragam, antara lain, psikoterapi, belat dan traksi eksternal, peralatan medis, biofarmasi dan radioterapi.

Salah satu contoh aplikasi praktik kedokteran ini adalah bedah rahang, tindakan operasi bedah yang bertujuan untuk memperbaiki ketidakaturan atau kelainan pada struktur rahang, baik kecacatan atau penyakit yang memengaruhi fungsi dan bentuk rahang. Prosedur bedah ini seringkali bertujuan untuk mengembalikan fungsi dan estetika rahang yang terganggu, sehingga memerlukan rekonstruksi sensitif terhadap teknik dan waktu yang panjang (J. J. *et al*., 2022). Salah satu metode bedah rahang adalah FFF (*Fibula Free Flap*) untuk merekonstruksi bentuk rahang. Prosedur ini melibatkan anestesi, pembukaan rongga mulut, pengangkatan tulang yang nekrotik, insisi pada kaki untuk mengambil fibula beserta pembuluh darahnya, pembentukan tulang yang diambil agar cocok dengan bagian rahang yang telah dihilangkan, penghubungan pembuluh darah ke bagian kepala dan leher, pemasangan tulang rahang baru dengan pelat dan sekrup, dan penjahitan bagian wajah dan leher yang sebelumnya terbuka. Setelah operasi selesai, pasien akan dirawat di unit perawatan intensif selama sehari untuk memantau pernapasan dan akan dipindahkan ke ruang perawatan umum setelah ventilator dilepas. Ini adalah contoh nyata dari bagaimana kedokteran kontemporer dapat mengubah dan memperbaiki kehidupan manusia (Oh *et al*., 2022).

2.2.2. Unity

Unity merupakan salah satu *game engine cross-patform* populer dan gratis, digunakan oleh berbagai kalangan mulai dari individu pemula, pelajar, pengembang *indie*, hingga perusahaan-perusahaan besar. Dikembangkan oleh Unity Technologies, platform ini menggunakan bahasa pemrograman C# sebagai basis untuk *scripting*. Unity mampu membuat video gim 3D dan 2D. Selain itu, Unity juga mendukung pembuatan simulasi interaktif, pembuatan film dan animasi, serta memenuhi kebutuhan industri untuk mengubah format objek 3D menjadi aplikasi yang imersif. Kemudahan yang diberikan Unity membuat game engine ini banyak dipakai mulai dari pembuatan video gim hingga aplikasi-aplikasi untuk riset. Unity menawarkan kemudahan bagi pengguna lewat beragam fitur-fitur yang diberikan mulai dari pembuatan *game* lintas platform, UI yang ramah, *scripting* yang mudah, banyaknya dukungan dari komunitas maupun resmi, dan kemudahan membuat aplikasi XR (Dealessandri, 2020).

2.2.3. Virtual Reality (VR)

VR (*Virtual Reality)* merupakan sebuah lingkungan virtual 3D yang memungkinkan pengguna untuk menjelajah dan berinteraksi dalam suatu dunia maya secara *real-time*. Pengalaman dalam

VR memungkinkan pengguna untuk merasakan interaksi layaknya di dunia nyata, seperti melakukan sentuhan, memegang objek, melihat sekitar, mendengar suara, berjalan, dan sebagainya. Pengguna dapat mengakses VR melalui perangkat keras seperti helm atau kacamata khusus. Tingkat kedalaman pengalaman VR menentukan sejauh mana pengguna dapat tenggelam ke dalam dunia virtual, sehingga mengurangi persepsi lingkungan fisik mereka dan memungkinkan pengguna untuk menerima pengalaman tersebut sebagai pengalaman nyata meskipun bersifat virtual. Lingkungan virtual dalam VR dapat bersifat *non-immersive*, dimana interaksi terjadi melalui perangkat input standar seperti mouse dan keyboard. Selain itu, ada juga jenis VR *semi-immersive* yang berfokus pada aspek visual 3D dan menggunakan kontrol khusus seperti pada simulasi pesawat terbang. Dan terakhir adalah *fully-immersive*, dimana pengguna melakukan interaksi langsung dengan lingkungan virtual melaui gerakan fisik. Penggunaan VR tidak hanya sebatas sebagai hiburan, namun juga telah meluas ke berbagai bidang seperti pelatihan, edukasi, kesehatan, dan sebagainya (Sheldon, 2022).

2.2.4. Augmented Reality (AR)

AR (*Augmented Reality*) adalah teknologi yang meng-*overlay* informasi digital, seperti gambar, video, atau data, ke dalam dunia nyata melalui perangkat seperti *smartphone*, tablet, atau HMD. Teknologi ini memungkinkan pengguna untuk melihat dan berinteraksi dengan elemen digital yang ditempatkan secara virtual dalam lingkungan fisik mereka. Ada dua metode umum yang digunakan untuk mencapai realitas augmentasi. Sistem *Video-Seethrough* memungkinkan pengguna melihat dunia nyata melalui kamera dan objek virtual ditempatkan pada gambar yang diambil, sedangkan sistem *Optical-Seethrough* memungkinkan pengguna melihat dunia luar melalui permukaan proyeksi transparan yang menampilkan konten AR (Schäfer *et al*., 2022). AR dapat figambarkan sebagai teknologi yang memungkinkan superposisi objek digital 2D dan 3D dalam pengaturan kehidupan nyata. Teknologi ini bekerja dengan menggunakan perangkat yang memiliki fungsionalitas pelacakan posisi pengguna dan penanda yang ditempatkan dalam skenario nyata untuk memproyeksikan informasi digital tambahan, sehingga meningkatkan realitas. AR dapat didefinisikan secara sederhana dengan tiga karakteristik utama. Pertama, AR adalah campuran dari dunia nyata dan virtual yang berdampingan dalam satu skenario. Kedua, kedua konteks, baik nyata maupun virtual, berinteraksi secara bersamaan dalam waktu nyata. Ketiga, terdapat implementasi gambar dan video digital 2D atau 3D dalam AR (Villagran-Vizcarra *et al*., 2023).

2.2.5. Mixed Reality (MR)

MR (*Mixed Reality*) adalah perpaduan antara dunia fisik dan digital yang memungkinkan interaksi antara manusia, komputer, dan lingkungan tiga dimensi yang alami dan intuitif. Realitas baru ini bergantung pada kemajuan dalam pemrosesan grafis, visi komputer, teknologi tampilan, sistem input, dan komputasi awan. Teknologi ini merupakan bagian dari spektrum realitas yang dikenal sebagai *Reality-Virtuality Continuum*, yang mencakup realitas fisik sepenuhnya di satu ujung dan realitas virtual sepenuhnya di ujung lainnya, dengan *Augmented Reality* (AR) dan *Augmented Virtuality* (AV) sebagai bentuk-bentuk perantara. Realitas campuran telah melampaui tampilan untuk mencakup banyak elemen penting. Ini mencakup pemahaman manusia melalui input ucapan, pelacakan tangan, dan mata, dan pemetaan spasial dan jangkar. Selain itu, realitas campuran juga mencakup suara yang berada di ruang fisik dan virtual, serta suara yang berada di lokasi dan di ruang. Dengan teknologi ini, orang dapat bekerja sama dengan aset 3D di ruang realitas campuran. Hal ini membuat realitas campuran dapat digunakan dalam banyak aplikasi (Microsoft, 2021).

2.2.6. Extended Reality (XR)

XR (Extended Reality) adalah istilah umum yang mencakup semua teknologi imersif yang

dihasilkan computer. Teknologi seperti VR, AR, dan MR dapat memperluas realitas yang dialami dengan menggabungkan lingkungan virtual dengan dunia nyata atau dengan membuat pengalaman realitas yang sepenuhnya imersif. XR tidak terbatas pada tiga teknologi tersebut; itu juga dapat mencakup berbagai teknologi dan variasi lainnya. Ide utama XR adalah menggabungkan pengalaman digital dan fisik untuk membuat lingkungan menjadi imersif dan interaktif. Aplikasi permainan, hiburan, pendidikan, perawatan kesehatan, arsitektur, dan teknik adalah beberapa industri di mana aplikasi XR telah atau dapat dipakai. XR akan memiliki kemampuan untuk mengubah banyak hal, termasuk hiburan, pendidikan, perawatan kesehatan, dan bisnis. Tetapi masalah seperti privasi, keamanan, dan aksesibilitas masih ada. Selain itu, pertimbangan moral yang berkaitan dengan XR, seperti risiko menjadi kecanduan dan kehilangan hubungan dengan dunia nyata, harus dipertimbangkan. Masa depan interaksi dan inovasi manusia dapat dibentuk oleh XR, baik melalui eksplorasi dunia virtual atau melalui peningkatan dunia fisik dengan lapisan digital (Willing, 2024).

2.2.7. Meta Quest 3

Meta Quest 3 adalah salah satu jenis perangkat XR yang sangat populer. Perangkat HMD (*Head Mounted Display*) Meta Quest awalnya dirancang oleh Oculus VR, tetapi sekarang diubah menjadi Meta Quest setelah Facebook atau Meta membeli perusahaan tersebut (TechCrunch, 2014). Meta Quest memiliki dua buah *controller* dan sebuah HMD dengan kamera dan sensor untuk fitur pengawasan dan *tracking* tangan, sehingga pengguna dapat menggunakan VR Headset tanpa memegang controller. Meta Quest 3 memiliki fitur *Passthrough* yang memungkinkan pengguna melihat keadaan sekitar secara *real time* dengan warna penuh, tidak seperti perangkat pendahulunya. Karena tampilan akan diganti dengan keadaan nyata di luar realitas virtual, fitur ini mematikan fitur VR untuk sementara dan menyalakan kamera yang ada pada HMD (Meta, 2022). Dengan memanfaatkan kemampuan *Passthrough* ini, lapisan realitas maya baru ditambahkan ke realitas nyata. Teknologi ini dapat digunakan dalam pengembangan AR dan MR. *Passthrough* Meta Quest 3 sangat berbeda dari versi sebelumnya, seperti Meta Quest 2, dalam hal warna. Di Meta Quest 2, *passthrough* masih menampilkan lingkungan dengan warna hitam putih, tetapi di Meta Quest 3, dapat menampilkan lingkungan secara penuh warna. Pengembangan aplikasi XR perencanaan bedah rahang akan menghasilkan *file* .APK yang dapat dipasang pada Meta Quest 3. Perangkat lunak yang dikenal sebagai SideQuest, salah satu portal aplikasi VR yang tersedia di pasar *virtual reality*, berfungsi untuk mengirimkan data *build* .APK dari Unity ke Meta Quest 3. Ini dapat dilakukan dengan mencolokkan kabel USB ke Meta Quest 3, kemudian menunggu sinyal koneksi dari SideQuest, setelah itu Unity dapat digunakan untuk *debugging* datagram. Perangkat lunak ini juga mendukung pengembangan aplikasi XR untuk Meta Quest 3.

2.2.8. Jumlah Partisipan

Teori bahwa lima partisipan tidak cukup dalam pengujian pengalaman pengguna (UX) telah didukung oleh berbagai penelitian dan praktik terbaru. Meskipun prinsip klasik menyatakan bahwa lima partisipan dapat menemukan sekitar 85% masalah kegunaan, metode ini memiliki keterbatasan yang signifikan terutama dalam menangani variasi pengguna dan kompleksitas tugas yang diuji. Strategi yang kuat untuk mengidentifikasi pengaruh urutan tugas terhadap kegunaan adalah membandingkan dua kelompok peserta yang melakukan tugas yang sama dalam urutan yang berbeda. Dalam konteks lima partisipan, metode ini tidak cukup karena variasi data, validitas hasil, dan kemampuan untuk menarik kesimpulan yang lebih mendalam dan akurat tentang pengalaman pengguna harus ditingkatkan. Hasil penelitian menjadi lebih valid dan dapat digeneralisasi dengan menggunakan lebih banyak partisipan dan menguji urutan tugas yang berbeda. Ini juga memberikan gambaran yang lebih luas tentang bagaimana

pengguna berinteraksi dengan system (Albert dan Tullis, 2013).

2.2.9. Metode Evaluasi

Salah satu metode evaluasi yang digunakan adalah menggunakan Likert Scale. Skala tersebut sering digunakan untuk mengumpulkan data pengguna yang melaporkan sendiri, memungkinkan penilaian subjektif atas berbagai bagian pengalaman mereka. Dengan beberapa titik tengah, skala ini biasanya berkisar dari "sangat setuju" hingga "sangat tidak setuju". Keunggulan dari Likert Scale adalah bahwa pertanyaannya mudah dipahami oleh responden dan memberikan konsistensi dalam pengumpulan data karena setiap pertanyaan memiliki opsi respons yang sama. Selain itu, dapat dilakukan analisis statistik untuk menemukan pola dan tren dalam persepsi pengguna dalam data yang dikumpulkan dengan menggunakan skala ini. Peneliti dapat mengukur tingkat kepuasan, kegunaan, dan elemen lain dari pengalaman pengguna secara kuantitatif dengan menggunakan skala ini. Namun, kekurangan Likert Scale adalah bahwa peserta mengisi skala secara subjektif dan beberapa cenderung memilih opsi tengah untuk menghindari ekstrem (Albert dan Tullis, 2013).

2.2.10. Uji System Usability Scale (SUS)

Sistem Usability Scale (SUS) adalah salah satu alat evaluasi yang paling banyak digunakan untuk mengukur kegunaan aplikasi. Pengguna diminta untuk menjawab sepuluh pertanyaan dan memberikan nilai pada setiap pertanyaan menggunakan Likert Scale yaitu skala satu sampai dengan lima. Skala bernilai satu menandakan pengguna sangat tidak setuju dengan pertanyaan sedangkan skala bernilai lima menandakan pengguna sangat setuju dengan pertanyaan. Skala ini dirancang untuk memberikan skor kegunaan aplikasi secara keseluruhan dengan nilai 0 hingga 100. SUS bermanfaat karena mudah digunakan dan dipahami oleh peneliti, dapat diterapkan pada berbagai jenis produk dan sistem, dan memiliki proses pengisian yang cepat. Pertanyaan-pertanyaan uji SUS adalah sebagai berikut:

1. Saya pikir saya akan sering menggunakan sistem ini.
2. Saya merasa sistem ini terlaalu rumit untuk digunakan.
3. Saya merasa sistem ini mudah digunakan.
4. Saya pikir saya memerlukan bantuan teknis untuk dapat menggunakan sistem ini.
5. Saya merasa fungsi-fungsi dalam sistem ini terintegrasi dengan baik.
6. Saya merasa terdapat terlalu banyak inkonsistensi dalam sistem ini.
7. Saya bisa mempelajari sistem ini dengan cepat.
8. Saya merasa sistem ini sangat rumit untuk digunakan.
9. Saya merasa sangat percaya diri saat menggunakan sistem ini.
10. Saya perlu mempelajari banyak hal sebelum bisa mulai menggunakan sistem ini.

Untuk menghitung nilai SUS, pertama-tama ubah skala dari sepuluh pertanyaan menjadi mulai dari "sangat tidak setuju", yang menerima 1 poin, hingga "sangat setuju", yang menerima 5 poin. Selanjutnya, hitung nilai dengan X, yang merupakan jumlah poin pada semua pertanyaan dengan nomor ganjil dikurangi 5, dan kemudian hitung nilai Y, yang merupakan jumlah poin pada semua pertanyaan dengan nomor genap dikurangi 5 lalu dihitung dengan menggunakan Rumus 2.1.

$$SUS\ Score = (X + Y) \times 2.5$$

Rumus 2.1 Perhitungan nilai SUS

Interpretasi dari hasil pengujian SUS adalah sebagai berikut:

- <50: Tidak dapat diterima
- 50 – 70: Marjinal

- >70: Dapat diterima.

(Albert dan Tullis, 2013).

2.2.11. Uji Shapiro-Wilk

Uji Shapiro-Wilk adalah statistik yang digunakan untuk menentukan apakah set data yang diberikan mengikuti distribusi normal atau tidak. Metode ini adalah metode uji normalitas yang efektif digunakan untuk sampel berjumlah kecil. Jika nilai *p-value* rendah, maka dapat menolak hipotesis nol tersebut dan mengatakan bahwa sampel belum dihasilkan dari distribusi normal (Malato, 2023). Rumus 2.2 adalah rumus untuk uji Shapiro-Wilk dengan a_i adalah koefisien uji, $x_{(i)}$ adalah angka ke $n-1+1$ pada data, x_i adalah angka ke i pada data, dan $\bar{x}$ adalah rata-rata.

$$W = \frac{(\sum_{i=1}^{n} a_i x_{(i)})^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Rumus 2.2 Uji Shapiro-Wilk

(Statistic How To, n.d).

Perhitungan hasil uji coba akan menggunakan bahasa pemrograman Python dengan *library* SciPy agar mendapatkan hasil uji statistik dan *p-value* yang lebih akurat. Interpretasi dari hasil perhitungan pengujian Kruskal Wallis adalah sebagai berikut:

- Jika (*p-value* < 0,05), ada perbedaan yang signifikan di seluruh interaksi.
- Jika (*p-value* ≥ 0,05), tidak ada perbedaan yang signifikan di seluruh interaksi.

2.2.12. Uji Kruskal Wallis

Uji Kruskal Wallis adalah tes hipotesis nonparametrik yang membandingkan tiga atau lebih kelompok independen. Ini juga disebut sebagai ANOVA satu arah pada peringkat oleh para ahli statistik (Frost, 2024). Uji Kruskal Wallis menawarkan alternatif yang kuat untuk ANOVA satu arah ketika asumsi normalitas tidak terpenuhi dan data rentan terhadap *outliers*. Misalnya, percobaan pertumbuhan tanaman yang melibatkan benih yang terpapar pada berbagai tingkat paparan sinar matahari menghasilkan data yang signifikan tentang jumlah sinar matahari yang masuk ke dalam tanaman. Contoh kedua, penggunaan uji Kruskal Wallis digunakan untuk menentukan apakah ada perbedaan yang signifikan dalam nilai rata-rata di antara tiga kelompok siswa yang nilainya tidak terdistribusi secara normal (Bobbit, 2019). Uji ini akan membandingkan tiga jenis interaksi dari dua kelompok dengan distribusi sampel yang tidak normal dan ukuran sampel yang kecil., sehingga analisis dari hasil uji coba aplikasi XR bedah rahang akan menggunakan rangkaian Kruskal Wallis untuk membandingkan performa partisipan dalam menggunakan aplikasi. Di bawah ini adalah Rumus 2.3 untuk penggunaan uji Kruskal Wallis (Statisticseasily, 2024). Dengan N adalah jumlah total observasi di semua grup, sedangkan k adalah jumlah grup. R adalah jumlah peringkat untuk kelompok *j*, sedangkan n_j adalah jumlah observasi pada kelompok *j*

$$H = \frac{12}{N(N+1)} \sum_{j=1}^{k} \frac{R_j^2}{n_j} - 3(N+1)$$

Rumus 2.3 Uji Kruskal Wallis

Perhitungan hasil uji coba akan menggunakan bahasa pemrograman Python dengan *library*

SciPy agar mendapatkan hasil uji statistik dan *p-value* yang lebih akurat. Interpretasi dari hasil perhitungan pengujian Kruskal Wallis adalah sebagai berikut:

- Jika (*p-value* < 0,05), ada perbedaan yang signifikan di seluruh interaksi.
- Jika (*p-value* ≥ 0,05), tidak ada perbedaan yang signifikan di seluruh interaksi.

2.2.13. Uji Wilcoxon Signed-Rank

Uji Wilcoxon diajukan oleh Frank Wilcoxon dalam makalah risetnya pada tahun 1945, dan digunakan untuk menganalisis perbedaan antara pasangan dalam dua kelompok data. Hipotesis statistik nonparametik digunakan untuk menguji data yang dapat dirangking tanpa perlu memiliki nilai numerik; ini adalah dasar dari uji Wilcoxon (Hayes, 2023). Ada dua jenis tes Wilcoxon: Wilcoxon Rank Sum dan Wilcoxon Signed-Rank. Untuk membandingkan pasangan efektivitas penggunaan, analisis dari hasil uji coba aplikasi XR bedah rahang akan menggunakan Wilcoxon Signed-Rank. Uji ini memiliki asumsi bahwa pasangan data yang diobservasi memiliki informasi tentang selisih dan perbedaan tanda. Di bawah ini adalah Rumus 2.4 untuk penggunaan uji Wilcoxon Signed-Rank (Educational Research Techniques, 2023). Dengan W adalah hasil uji statistik yang dicari. sgn adalah fungsi sign. x_{i1} dan x_{i2} adalah pasangan data ke-i. R_i adalah rangking atau peringkat dari i.

$$W = \sum_{i=1}^{N} [sgn(x_{i2} - x_{i1}).R_i]$$

Rumus 2.4 Uji Wilcoxon Signed-Rank

Perhitungan hasil uji coba akan menggunakan bahasa pemrograman Python dengan *library* SciPy agar mendapatkan hasil uji statistik dan *p-value* yang lebih akurat. Interpretasi dari hasil perhitungan pengujian Wilcoxon Signed-Rank adalah sebagai berikut:

- Jika (*p-value* < 0,05), ada perbedaan yang signifikan di seluruh interaksi.
- Jika (*p-value* ≥ 0,05), tidak ada perbedaan yang signifikan di seluruh interaksi.

(halaman ini sengaja dikosongkan)

BAB 3
METODOLOGI

3.1 Metode Penelitian

Urutan metode yang digunakan untuk menyelesaikan Tugas Akhir adalah sebagai berikut:

1. Penyusunan Kerangka Tugas Akhir
2. Studi Literatur
3. Perancangan Sistem Aplikasi Realitas Virtual
4. Persiapan Perangkat Pendukung
5. Pengembangan Aplikasi
6. Pengujian dan Evaluasi
7. Penyusunan Buku Tugas Akhir

3.2 Peralatan Pendukung

Perangkat - perangkat pendukung yang digunakan untuk proses pengembangan Tugas Akhir ini adalah sebagai berikut:

1. Laptop atau Komputer
2. Meta Quest 3
3. Kabel USB Type C
4. Unity
5. SideQuest
6. 3D Slicer
7. Visual Studio 2022

3.3 Rencana Implementasi dan Uji Coba

Tahap pertama sebelum merancang aplikasi adalah menyusun kerangka Tugas Akhir untuk memastikan bahwa pengembangan aplikasi XR memiliki landasan yang kuat. Latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat yang telah ditetapkan akan menjadi landasan utama dalam pengembangan aplikasi. Batasan masalah juga berperan dalam fokus perancangan aplikasi untuk menyelesaikan permasalahan yang telah dirumuskan.

Tahap kedua adalah melakukan studi literatur mengenai hal-hal yang perlu dipersiapkan dalam perancangan dan pengembangan aplikasi. Langkah-langkah dalam persiapan pengembangan aplikasi mencakup:

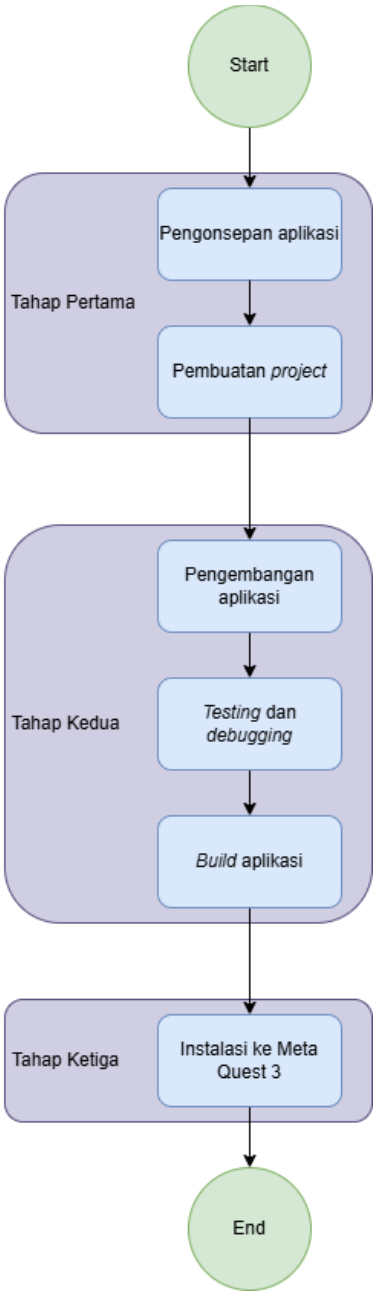
1. Mengidentifikasi jurnal atau penelitian terkait penggunaan teknologi VR atau XR dalam kedokteran, terutama pada prosedur operasi bedah dan rekonstruksi rahang.
2. Memahami *library* atau *package* yang digunakan dalam pengembangan aplikasi XR. Tugas Akhir ini akan menggunakan Oculus Integration SDK sebagai *package* utama untuk mengembangkan aplikasi di Unity untuk perangkat keras Meta Quest 3.
3. Menemukan aset yang digunakan seperti model 3D untuk fibula dan mandibula, UI, atau *audio* yang tersedia di Unity Asset Store dan internet yang dapat digunakan secara legal. Untuk aset berupa model tulang, dapat mengonversi dari file DICOM menjadi STL menggunakan aplikasi *converter* 3D Slicer atau dapat diunduh dari website www.embodi3d.com.
4. Pengembangan aplikasi XR memanfaatkan *package open source* pada Github bernama EzySlice.[1]

[1] https://github.com/DavidArayan/ezy-slice

Tahap ketiga adalah perancangan sistem aplikasi XR yang dimulai dengan merancang *gameplay* atau *interaction* terlebih dahulu karena *gameplay* atau *interaction* menjadi fokus utama dalam pengerjaan Tugas Akhir. Karena Tugas Akhir ini bertujuan untuk mengembangkan aplikasi untuk persiapan bedah rahang, pengalaman pengguna dari aplikasi XR harus disempurnakan terlebih dahulu sebelum melanjutkan ke perancangan antarmuka pengguna.

Tahap keempat adalah mempersiapkan peralatan pendukung dalam pengembangan aplikasi. Peralatan ini mencakup perangkat keras seperti laptop atau komputer sebagai media pengembangan aplikasi, kabel USB Type C untuk menghubungkan prototipe aplikasi ke dalam VR Headset guna proses *debugging*, dan Meta Quest 3 sebagai VR Headset yang digunakan. Perangkat lunak yang digunakan mencakup Unity sebagai *engine* pengembangan aplikasi, SideQuest untuk instalasi *file* aplikasi pada VR Headset, 3D Slicer untuk mengonversi *file* DICOM menjadi STL, dan Visual Studio 2022 untuk menulis kode program. Pada tahap ini juga dilakukan uji coba dan pengecekan untuk memastikan semua perangkat beroperasi dengan semestinya.

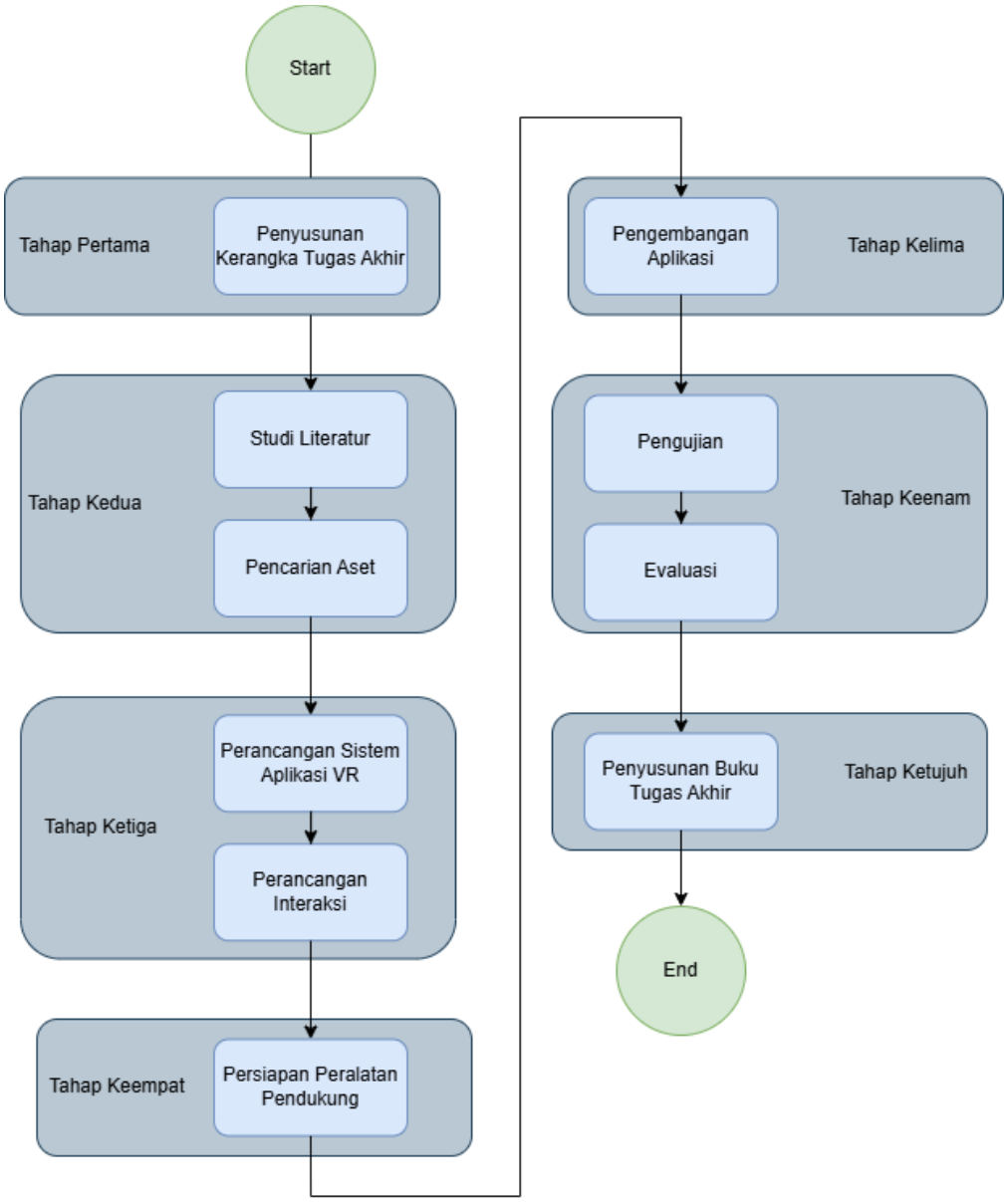
Tahap kelima adalah pengembangan sistem aplikasi yang dimulai pengonsepan bagaimana *gameplay* atau interaksi yang ada di dalam aplikasi XR. Setelah konsep dirasa sudah matang, maka selanjutnya adalah inisiasi *project* pada Unity. Selanjutnya adalah mencari aset-aset dan *packages* atau *library* yang akan digunakan. Setelah pencarian dan inisiasi *project*, maka selanjutnya masuk ke proses pengembangan aplikasi. Diawali dengan memasang *packages* yang akan digunakan di dalam lingkungan Unity. Selanjutnya adalah membuka sebuah *sample scene* yang tersedia di dalam *package* Oculus Interaction SDK, terutama yang memuat interaksi menggunakan *hand tracking* untuk mempermudah proses pengembangan aplikasi dengan menghindari pembuatan interaksi dari awal. Selanjutnya adalah menyalin objek-objek yang ada pada *hierarchy* pada *sample scene* ke dalam *scene* baru yang telah dibuat yang berfokus pada pengembangan *gameplay* seperti kamera yang sudah tersedia pada *package*. Setelah kamera sudah tersedia pada *scenes*, selanjutnya adalah mulai memasukkan objek-objek lain yang akan digunakan dalam interaksi seperti aset-aset. Kemudian adalah pengembangan interaksi tambahan yang tidak ada pada *package* Oculus Interaction SDK seperti interaksi memotong. Setelah melakukan pengembangan, kemudian adalah melakukan *debug* dan juga pengujian apakah aplikasi bekerja seperti yang diharapkan dan adakah penambahan atau perbaikan dalam aplikasi. Setelah semua dirasa cukup, selanjutnya adalah mencoba membangun (*build*) *scene* tersebut menjadi sebuah *file* .APK. Setelah berhasil dibangun, langkah berikutnya adalah mencoba memasang *file* .APK ke dalam Meta Quest 3 menggunakan aplikasi SideQuest. Jika *file* .APK dapat dipasang dan dapat dimainkan, hal tersebut menandakan bahwa proses percobaan pengembangan telah berhasil. Dengan demikian, dapat dilanjutkan pengembangan dan penyempurnaan *gameplay* sesuai dengan rancangan awal. Pengembangan aplikasi dianggap selesai ketika sebuah prototipe telah tercipta dan dapat dimainkan. Untuk memudahkan dalam memahami urutan tahapan pengembangan aplikasi, maka dapat dilihat diagram alur pada Gambar 3.1.



Gambar 3.1 Diagram alur tahapan pengembangan aplikasi

Tahap keenam adalah pengujian dan evaluasi aplikasi yang telah dikembangkan. Setiap hasil yang diperoleh dari proses pengujian dan evaluasi akan menjadi acuan untuk menilai keberhasilan aplikasi serta untuk melakukan perbaikan yang diperlukan. Pengujian dilakukan kepada beberapa partisipan untuk mendapatkan beragam data untuk kemudian dievaluasi beserta pemberian saran.

Tahap ketujuh adalah penyusunan buku Tugas Akhir. Setiap kegiatan dan proses pengembangan akan didokumentasikan dalam jurnal sementara yang akan dijadikan referensi dalam penyusunan buku. Buku yang disusun akan menjelaskan dasar teori, metode yang digunakan selama melaksanakan Tugas Akhir, rancangan sistem aplikasi XR, proses pengembangan dari rancangan sistem aplikasi XR, serta hasil pengujian dan evaluasi dari aplikasi XR yang telah dikembangkan. Untuk lebih memahami tahapan penelitian, maka diagram alur dapat dilihat pada Gambar 3.2.



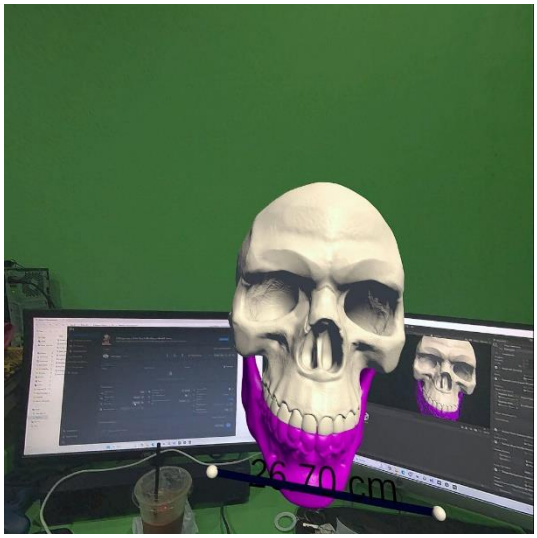
Gambar 3.2 Diagram alur tahapan penelitian

3.4 Rancangan Aplikasi XR

Aplikasi XR dibuat dengan tujuan untuk membantu ahli bedah dalam mengatasi keterbatasan aksesibilitas, biaya tinggi, dan waktu yang lama dalam proses persiapan bedah rahang. Desain aplikasi dibuat untuk membantu pengguna dalam memvisualisasikan bentuk tulang dan bagian mana saja yang harus dilakukan prosedur operasi bedah. Pengguna akan dibawa ke sebuah ruang di mana terdapat sebuah tulang tengkorak dan penggaris. Pengguna bisa berinteraksi dengan objek tersebut menggunakan tangan seperti memperbesar atau memperkecil, merotasi, dan menggeser objek. Kemudian pengguna akan diberikan sebuah opsi untuk memunculkan *plane* lewat interaksi dengan tengkorak seperti menyentuh atau memosisikan sebuah *plane* transparan yang berada di jari telunjuk pengguna. *Plane* yang muncul akan digunakan sebagai pemotong. Pengguna dapat mengurungkan hasil potongan jika dirasa tidak sesuai dan dapat mengulang proses memotong lagi. Karena pengembangan aplikasi XR mirip seperti pengembangan aplikasi gim, rancangan sistem aplikasi akan dipaparkan dengan taksonomi mekanik yang tertera pada buku “The Art of Game Design: A Book of Lenses” yang ditulis oleh desainer gim Jesse Schell (Schell, 2008).

3.4.1. Space

Ruang dalam aplikasi disebut sebagai *space*, yang merujuk pada lingkungan tempat adegan berlangsung. Aplikasi ini menggunakan Continuous Space, yang memungkinkan pengguna untuk bergerak dalam segala arah di ruang terbatas. Dalam peran sebagai ahli bedah, pengguna dihadapkan pada tugas memotong mandibula dengan cara memunculkan bidang (*plane*). *Plane* tersebut kemudian disisipkan ke dalam mandibula, dengan bantuan penggaris yang disediakan. Setelah pengguna merasa jarak ukuran di mandibula sesuai, pengguna dapat memunculkan *plane* di titik yang telah ditentukan. Dalam aplikasi, pengguna dapat duduk atau berdiri diam dan berinteraksi dengan model tengkorak, seperti merotasi, memindah, dan sebagainya. Namun, pengguna juga dapat bergerak bebas, misalnya berjalan mengitari tengkorak. Contoh perspektif pengguna dalam aplikasi XR dapat dilihat pada Gambar 3.3 yang menunjukkan apa yang dilihat pengguna ketika mengenakan VR Headset dan membuka aplikasi. Aplikasi ini juga memiliki mode *Passthrough* milik Oculus, yang memungkinkan pengguna melihat lingkungan sekitar di luar lingkungan virtual.



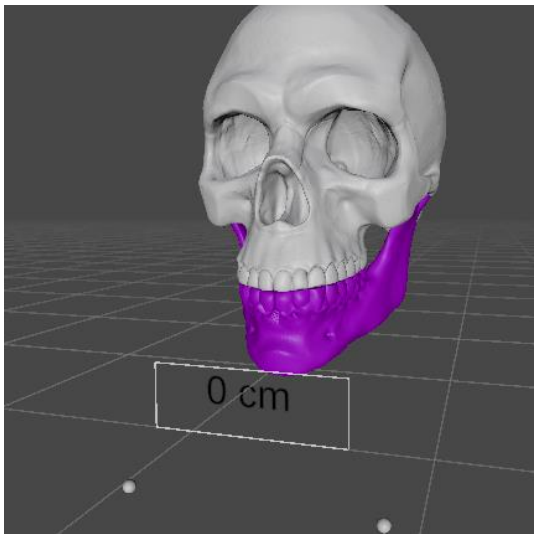
Gambar 3.3 Tampilan *space* dalam aplikasi

3.4.2. Object, Attributes, and States

Object merupakan entitas atau elemen diskrit yang hadir di dalam dunia gim. *Object* dapat berupa elemen interaktif di dalam aplikasi. *Objects* tersebut disebut GameObject di dalam *project* Unity. *Object* dalam aplikasi memiliki beberapa properti di dalamnya, mulai dari transformasi hingga *script* yang memungkinkan *object* berinteraksi. Properti-properti tersebut termasuk sebagai atribut (*attributes*) dari *object*, seperti ukuran *object* tulang dan sebuah *script* agar tulang dapat berinteraksi dengan pengguna. Kemudian, *states* suatu *object* merujuk pada kondisi saat ini di dalam aplikasi. *States* dapat mencerminkan nilai atau atribut (*attributes*) *states*. *States* dapat berubah berdasarkan tindakan dan interaksi pengguna, misalnya, *states* tulang sebelum dan setelah dipotong. Berikut adalah *object*, *attributes*, dan *states* yang ada pada aplikasi ini.

3.4.3. Tulang

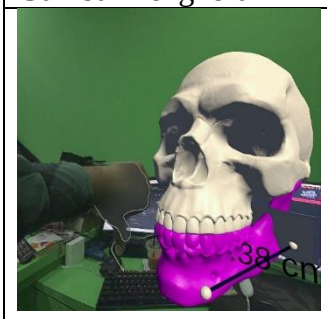
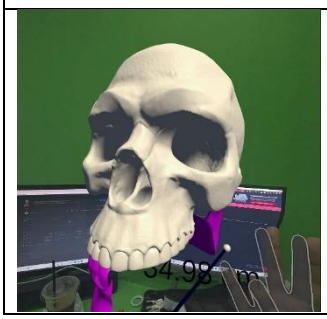
Tentunya, *object* utama yang terdapat dalam aplikasi ini adalah tulang, dengan jenis tulang meliputi tulang tengkorak. Tulang tersebut akan mengalami proses pemotongan yang dapat dilakukan oleh pengguna. Sebagai contoh, Gambar 3.4 adalah gambar dari model tulang tengkorak yang ada di dalam aplikasi. Atribut utama yang dimiliki oleh tulang adalah *script* yang memungkinkan pengguna untuk memegang dan mentransformasi tulang, seperti mengubah ukuran atau melakukan rotasi.



Gambar 3.4 Model 3D untuk tulang tengkorak

Model tulang tengkorak memiliki dua *states*, yaitu *original states* yang berarti keadaan tengkorak masih dalam kondisi semula sebelum pemotongan, dan *sliced states* yang berarti keadaan tengkorak telah terpotong. Contoh keadaan tulang sebelum dan sesudah terpotong dapat dilihat pada Tabel 3.1.

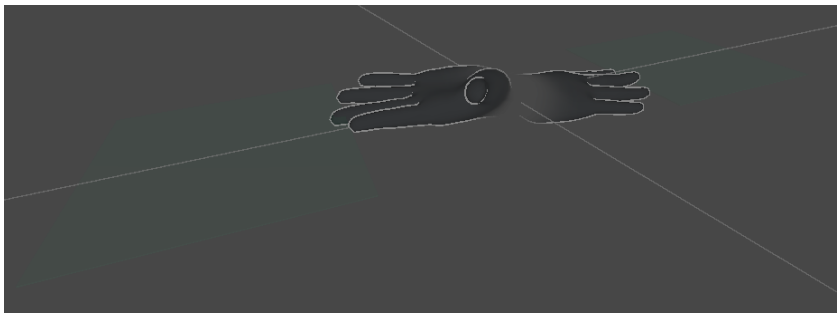
Tabel 3.1 Perbandingan State Tengkorak Sebelum dan Sesudah Terpotong

Gambar Tengkorak	*States*	Deskripsi
	*Original*	Keadaan atau *states* pertama adalah *Original States* dimana model tengkorak masih belum terpotong, sehingga pengguna dapat melakukan tindakan seperti memunculkan *plane*, menaruh penggaris, dan lain – lain.
	*Sliced*	Keadaan atau *states* kedua adalah *Sliced States* dimana model tengkorak sudah terpotong yang dapat dilihat pada gambar, terdapat rongga di antara mandibula.

3.4.4. Hand Tracking

Aplikasi XR ini menggunakan tangan sebagai *controller* utama. Pengguna akan seolah-olah berinteraksi langsung dengan GameObject lain yang ada di dalam lingkungan virtual. Pengguna dapat memegang tulang, memunculkan *pointer* dan *plane* transparan, dan melakukan pose untuk memotong tulang. Gambar 3.5 dan Gambar 3.6 adalah gambar tangan yang ada di dalam aplikasi.

Gambar 3.5 Tangan dengan pointer

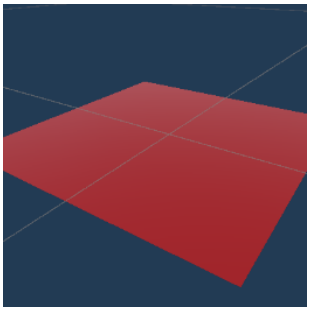


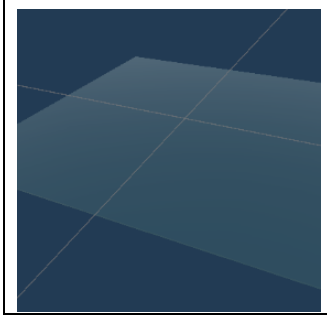
Gambar 3.6 Tangan dengan plane transparan penanda

3.4.5. Planes

Plane digunakan sebagai instrumen pemotong di dalam aplikasi ini, namun ada satu *plane* khusus yang digunakan sebagai penanda untuk memunculkan *plane* pemotong. Pengguna dapat menampilkan *plane* pemotong dengan berbagai macam cara, seperti menyentuh mandibula dengan jari telunjuk atau menggunakan *plane* penanda transparan. Setelah *plane* pemotong muncul, pengguna dapat berinteraksi dengan *plane* tersebut, seperti memindah, merotasi, dan memperbesar sesuai dengan keinginan pengguna. Proses pemotongan hanya dapat dilakukan sekali. Namun, jika pengguna merasa kurang puas dengan hasil potongan atau melakukan kesalahan saat pemotongan, mereka dapat membatalkan proses pemotongan sebelumnya dan mencoba memosisikan kembali *plane*. Untuk mengetahui perbedaan kedua *plane*, dapat dilihat pada Tabel 3.2.

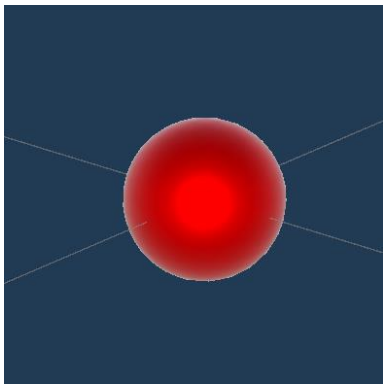
Tabel 3.2 Perbandingan Plane Pemotong dan Plane Penanda

Gambar *Plane*	Jenis Plane	Deskripsi
	*Plane* pemotong	*Plane* berwarna merah yang digunakan sebagai alat pemotong mandibula. Diperlukan dua *plane*, yang dimunculkan oleh pengguna untuk dapat memotong tulang. *Plane* pemotong dapat muncul dari jarak antara dua *point*, tepat dimana *point* itu sendiri berada, atau dari lokasi dimana *plane* penanda berada. Metode pemunculan *plane* tergantung jenis *scene* yang dimainkan

		pengguna.
	*Plane* penanda	*Plane* penanda memiliki material transparan yang digunakan untuk menandakan dimana *plane* pemotong akan muncul. *Plane* ini akan muncul di ujung jari pengguna.

3.4.6. Pointer

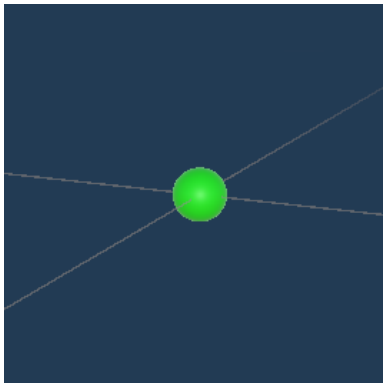
Pointer adalah sebuah bola kecil berwarna merah yang terdapat pada ujung jari telunjuk dari *hand tracking* aplikasi. Fungsi dari pointer adalah sebagai alat untuk memunculkan bola kecil lain berwarna hijau yang disebut *point*. *Point* tersebut nantinya akan digunakan sebagai penanda di mana *plane* akan muncul. Gambar 3.7 adalah model dari *object pointer*.



Gambar 3.7 Tampilan model pointer

3.4.7. Point

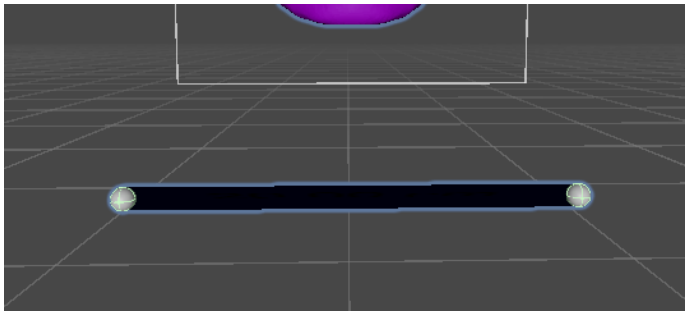
Point adalah sebuah bola hijau kecil yang berfungsi sebagai pin untuk memunculkan *plane*. Tergantung pada *scene* di mana pengguna berada, kegunaan *point* ini juga berbeda. Di salah satu *scene*, dua buah *point* digunakan sebagai pin di mana *plane* akan muncul. *Point* pertama digunakan sebagai titik awal, sementara *point* kedua digunakan sebagai titik akhir. Di antara kedua *point* tersebut, *plane* dapat muncul dengan mengalkulasi dua jarak *point* dan menemukan titik tengah di mana *plane* akan muncul. Di *scene* lain, *point* digunakan sebagai pin untuk memunculkan *plane* tepat di mana *point* berada, tanpa memerlukan dua buah *point*. Gambar 3.8 adalah model dari *object point*.



Gambar 3.8 Tampilan model point

3.4.8. Penggaris

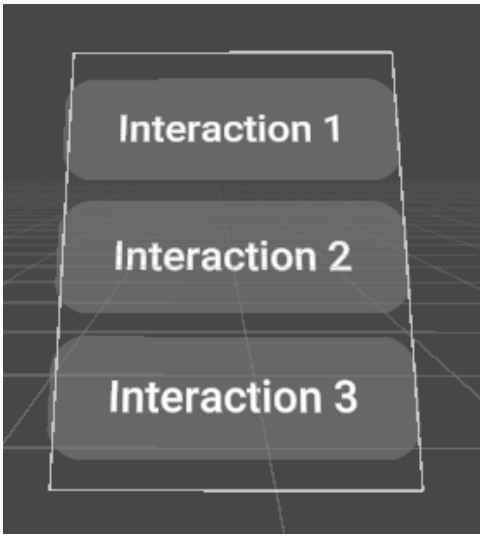
Penggunaan penggaris dalam aplikasi ini menjadi alat bantu untuk membantu pengguna mengukur dengan presisi panjang tulang yang akan dipotong. Selain itu, penggaris digunakan untuk menentukan lokasi untuk memunculkan *plane* sesuai dengan kebutuhan. Fungsinya juga melibatkan pengamatan ukuran tulang yang ingin dipotong. Pengaturan penggaris dapat disesuaikan sesuai preferensi pengguna dengan cara memegang kedua ujung penggaris dan menempatkannya pada bagian tulang sebagai acuan. Gambar 3.9 adalah gambar model untuk penggaris.



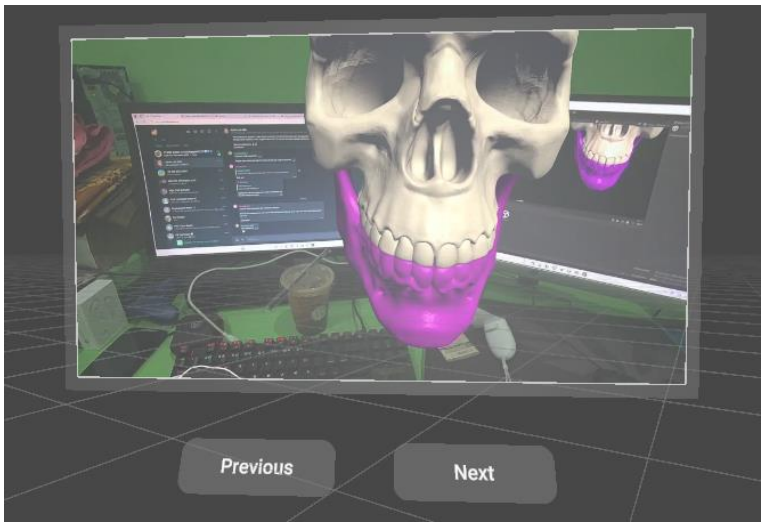
Gambar 3.9 Tampilan penggaris

3.4.9. Antarmuka

Antarmuka atau UI aplikasi XR persiapan bedah rahang menggunakan aset bawaan yang ada pada Unity dan menggunakan aset sampel dari *package* Oculus Interaction SDK. Antarmuka di aplikasi XR ini hanya berfungsi sebagai pemindah *scene*, serta sebuah panel dengan dua tombol untuk memainkan video panduan tentang cara dan urutan memotong tulang. Antarmuka aplikasi XR ini mencakup sejumlah tombol yang bertujuan untuk memindah *scene* karena di dalam aplikasi terdapat beberapa *scene* berbeda dengan interaksi pemotongan yang berbeda pula. Lalu antarmuka kedua adalah sebuah *video player* dan juga tombol *next* dan *previous* untuk berpindah ke video selanjutnya atau sebelumnya. Antarmuka berlokasi di samping pengguna sehingga pengguna dapat fokus dengan tengkorak dan tidak terganggu dengan adanya antarmuka di sekitarnya, kecuali dengan *video player* yang dapat diambil dan ditempatkan sesuai keinginan pengguna. Gambar 3.10 adalah antarmuka untuk memindah *scene*, sedangkan Gambar 3.11 adalah antarmuka untuk memainkan video bantuan.



Gambar 3.10 Tampilan antarmuka menu scenes



Gambar 3.11 Tampilan antarmuka video player

3.4.10. Actions

Actions mencakup semua tindakan yang dapat dilakukan oleh pengguna di dalam aplikasi. Terdapat dua jenis tindakan yang terjadi selama pengguna menggunakan aplikasi, yaitu *operative actions* dan *resultant actions*. Berikut penjelasan lebih lanjut mengenai kedua tindakan yang ada di aplikasi XR persiapan bedah rahang.

3.4.11 Operative Actions

Operative actions merupakan tindakan utama yang dapat dilakukan pengguna di dalam aplikasi. Beberapa tindakan tersebut meliputi:

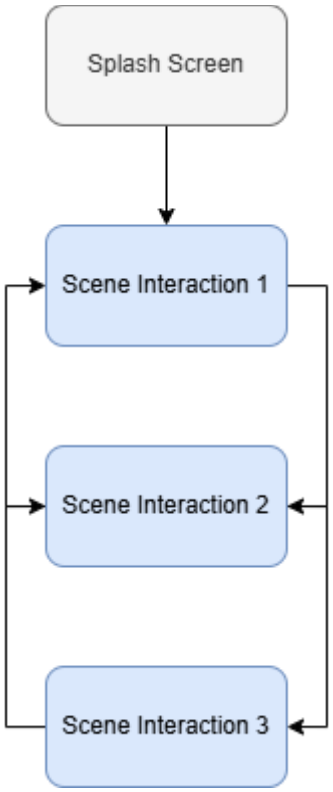
- Pengguna memiliki kemampuan untuk bergerak bebas di lingkungan virtual yang disediakan.
- Pengguna dapat memegang objek yang ada di dalam aplikasi.
- Pengguna dapat mengatur skala objek, baik memperbesar maupun memperkecilnya.
- Pengguna dapat mengatur posisi dan rotasi objek ke segala arah.

3.4.12 Resultant Actions

Dalam konteks aplikasi XR persiapan bedah rahang, tindakan tersebut meliputi kemampuan pengguna untuk melakukan *hand pose* untuk memunculkan objek *pointer* atau *plane*, mengurungkan pemunculan *plane*, dan juga untuk memotong dan mengurungkan hasil potongan. Pengguna juga dapat menekan tombol-tombol antarmuka untuk berpindah *scene*, mengaktifkan atau menonaktifkan mode *passthrough*, atau berpindah ke video selanjutnya atau sebelumnya.

3.4.13. Scenes

Sebuah *scene* mengacu pada segmen tertentu di dalam aplikasi. Segmen-segmen tersebut merupakan lingkungan dimana pengguna akan berinteraksi. Di dalam *scene*, terdapat beberapa *object* yang akan pengguna gunakan sebagai mendia interaksi sehingga menciptakan pengalaman yang imersif. Dalam kasus aplikasi ini, *objects* tersebut adalah *object* yang digunakan sebagai interaksi pengguna seperti model tengkorak hingga antarmuka. Alur *scene* pada aplikasi XR ini dapat dilihat pada Gambar 3.12.



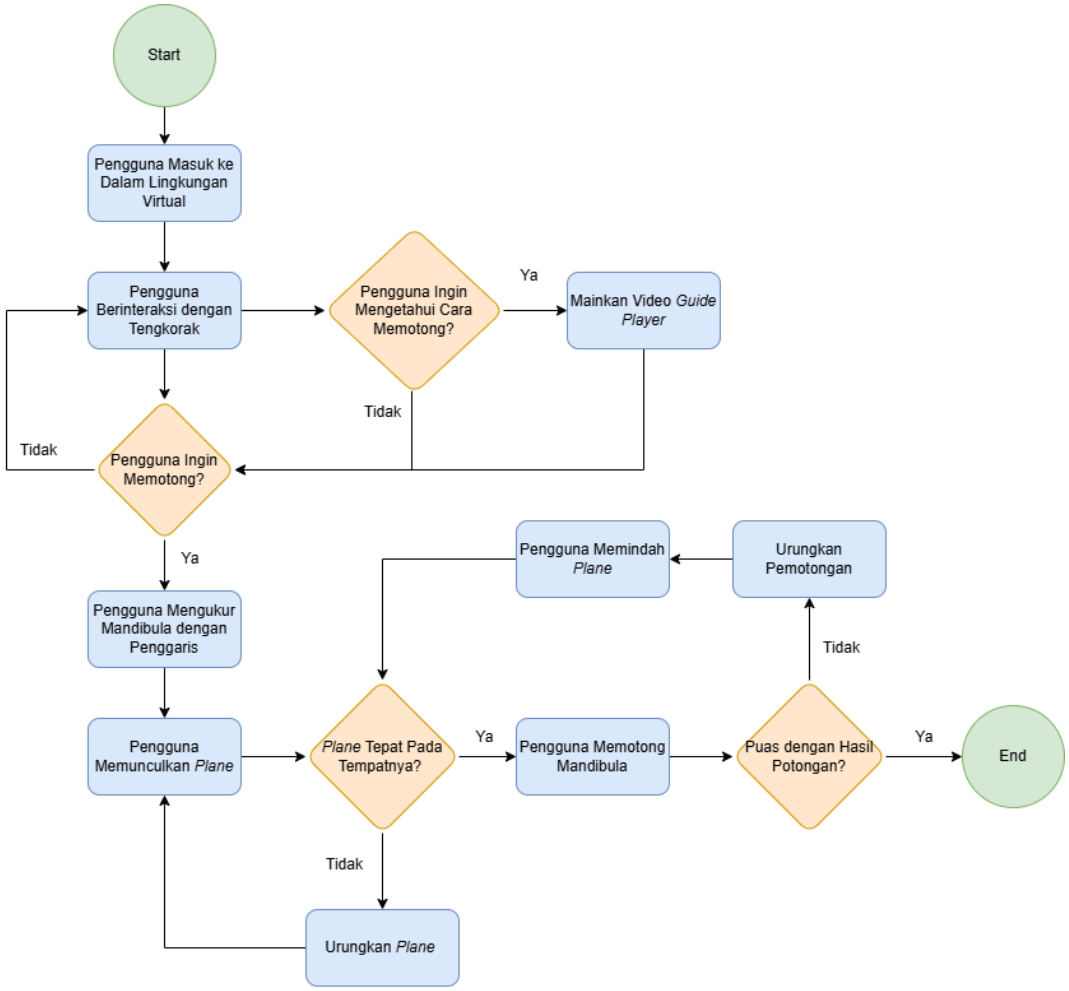
Gambar 3.12 Diagram alur perpindahan *scenes*

Di dalam *scenes* yang ada pada aplikasi, pengguna akan menjalankan sebuah interaksi dengan tujuan memotong mandibula. Yang membedakan antara satu *scene* dengan yang lain adalah jenis interaksi pengguna dalam menuju proses pemotongan. Pada *scene* interaksi pertama, pengguna berinteraksi dengan tulang tengkorak menggunakan *pointer* yang ada di ujung jari telunjuk untuk memunculkan dua buah *point* yang kemudian digunakan untuk memunculkan *plane*. *Scene* interaksi kedua kurang lebih sama, namun pengguna hanya perlu memunculkan satu buah *point* untuk memunculkan *plane*. Pada *scene* interaksi ketiga, pengguna menggunakan *plane* penanda transparan yang ada di ujung jari telunjuk untuk memunculkan *plane*. Jika pengguna kebingungan tentang cara berinteraksi di dalam *scenes* tersebut, mereka dapat melihat video panduan yang ada di setiap *scene*. Tujuan di semua *scene* adalah sama, yaitu memotong mandibula, namun jenis interaksinya berbeda, terutama dalam cara memunculkan *plane*. Pengguna nantinya akan membandingkan mana interaksi yang menurut mereka paling nyaman sesuai dengan preferensi mereka dan mana yang kurang. Pengguna dapat berpindah dari satu *scene* ke *scene* yang lain sewaktu-waktu dengan menekan antarmuka *menu scene* yang berada di samping kiri pengguna.

3.5. Alur

Alur dari aplikasi XR ini dimulai dengan pengguna membuka aplikasi. Kemudian, pengguna mengenakan VR Headset dan memasuki lingkungan virtual. Di dalam lingkungan virtual, pengguna disajikan dengan berbagai macam *object* seperti model tengkorak, penggaris, dan antarmuka. Pengguna dapat mencoba berinteraksi dengan tengkorak dengan cara memegang, memperbesar, memperkecil, dan merotasi tengkorak. Jika pengguna ingin memotong, maka pertama kali, pengguna akan memasang penggaris untuk mengukur mandibula sesuai keinginan. Jika pengguna tidak tahu cara memotong, mereka dapat memainkan video panduan proses pemotongan mandibula. Setelah memahami caranya, pengguna dapat memunculkan

plane pemotong. Setelah pengguna merasa lokasi *plane* tepat, pengguna dapat melanjutkan dengan memotong mandibula. Namun, jika dirasa lokasi *plane* belum tepat, pengguna dapat mengurungkannya. Setelah pemotongan, mandibula akan memiliki rongga sebagai hasil dari bagian yang terpotong. Jika pengguna merasa kurang puas atau melakukan kesalahan dengan hasil potongan, mereka dapat mengurungkan potongan, memindah *plane* sesuai keinginan, lalu kembali memotong mandibula. Untuk mempermudah pemahaman alur tersebut, dapat dilihat pada Gambar 3.13 untuk diagram alur aplikasi XR persiapan bedah rahang.



Gambar 3.13 Diagram alur penggunaan aplikasi

3.6. Pengujian

Salah satu cara untuk mendapatkan umpan balik tentang aspek positif dan negatif dari aplikasi XR persiapan bedah rahang adalah dengan melakukan sebuah pengujian yaitu pengujian performa pengguna dan pengujian pengalaman pengguna. Pengujian performa pengguna adalah pengujian yang akan membandingkan mana interaksi yang paling cepat dan nyaman berdasarkan dari waktu partisipan menggunakan aplikasi. Pengujian pengalaman pengguna adalah pengujian yang akan menilai mana jenis interaksi yang paling sesuai bagi partisipan. Partisipan diprioritaskan bagi mereka yang belum pernah mencoba aplikasi VR atau XR sebelumnya.

Pada awal pengujian, partisipan akan diberikan tentang pengetahuan dan informasi umum tentang aplikasi XR bedah rahang. Kemudian penguji akan menjelaskan fitur-fitur dan tujuan

dari aplikasi. Selanjutnya, penguji akan memberikan contoh demonstrasi bagaimana cara menggunakan aplikasi. Jika partisipan merasa siap untuk mencoba menggunakan aplikasi, maka penguji akan mempersilakan partisipan menyelesaikan beberapa *task*.

Pengujian performa pengguna akan membandingkan performa dua grup yang masing-masing berisi tiga orang partisipan. Grup pertama akan diminta untuk mencoba aplikasi dengan urutan interaksi pertama, kedua, dan ketiga. Waktu yang dibutuhkan oleh setiap partisipan dalam menggunakan aplikasi akan dihitung. Selanjutnya, grup kedua akan diminta untuk mencoba aplikasi dengan urutan interaksi ketiga, kedua, dan pertama, dan waktu yang dibutuhkan oleh partisipan juga akan dihitung. Hasil pengujian akan membandingkan mana interaksi yang paling cepat dan mudah digunakan oleh kedua grup. Perhitungan akan dilakukan per individu dari setiap grup, kemudian waktu yang dihabiskan akan dirata-rata untuk setiap grup. Selanjutnya, hasil rata-rata waktu dari kedua grup akan dibandingkan untuk menentukan interaksi mana yang memberikan kinerja terbaik.

Pengujian pengalaman pengguna akan mengevaluasi aplikasi dan membandingkan interaksi mana yang menjadi preferensi pengguna. Ketika partisipan telah selesai melakukan beberapa *task* pada pengujian performa pengguna, maka penguji akan memberikan beberapa pertanyaan mengenai umpan balik dari interaksi-interaksi yang partisipan telah lakukan ketika mencoba aplikasi dan partisipan diminta untuk memberikan skala dari satu sampai lima pada tiap pertanyaan. Berikut merupakan beberapa pertanyaan untuk mendapatkan umpan balik dari partisipan.

11. Saya pikir saya akan sering menggunakan sistem ini.
12. Saya merasa sistem ini terlalu rumit untuk digunakan.
13. Saya merasa sistem ini mudah digunakan.
14. Saya pikir saya memerlukan bantuan teknis untuk dapat menggunakan sistem ini.
15. Saya merasa fungsi-fungsi dalam sistem ini terintegrasi dengan baik.
16. Saya merasa terdapat terlalu banyak inkonsistensi dalam sistem ini.
17. Saya bisa mempelajari sistem ini dengan cepat.
18. Saya merasa sistem ini sangat rumit untuk digunakan.
19. Saya merasa sangat percaya diri saat menggunakan sistem ini.
20. Saya perlu mempelajari banyak hal sebelum bisa mulai menggunakan sistem ini.

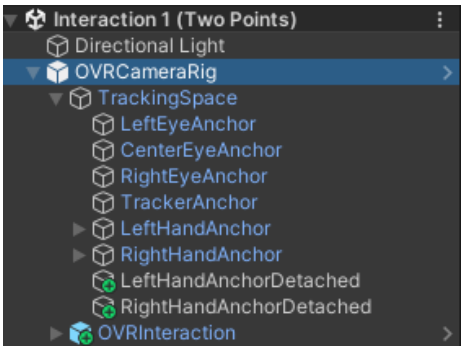
Setelah peserta menjawab semua pertanyaan pada formulir dan menyerahkannya kepada penguji, pengujian dianggap selesai. Pengujian yang direncanakan akan diberikan kepada enam peserta, dan hasilnya akan disimpulkan dan dipertimbangkan setelah ujian selesai secara keseluruhan.

(halaman ini sengaja dikosongkan)

BAB 4
IMPLEMENTASI

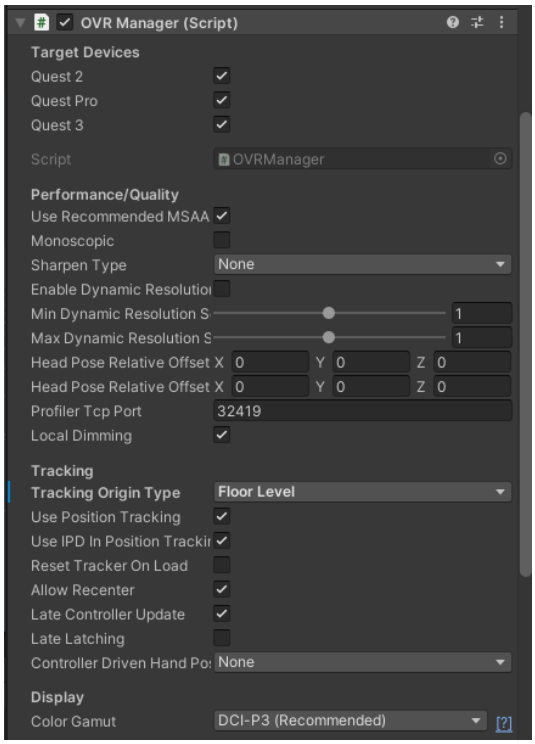
4.1 Implementasi Aplikasi XR

Dalam pengembangan aplikasi XR, hal paling mendasar untuk dapat mewujudkan lingkungan XR itu sendiri adalah dengan dibuatnya sebuah peralatan XR di dalam proyek Unity. *Package* milik Oculus Interaction SDK sudah menyediakan sebuah *prefab* untuk menggantikan kamera biasa pada *Unity* dengan kamera VR. *Prefab* tersebut bernama OVRCameraRig yang menyediakan objek transformasi untuk mewakili *tracking space* Oculus. *Prefab* ini berisi *tracking space* GameObject untuk menyempurnakan relasi antara kerangka referensi pelacakan kepala dan dunia virtual. Di bawah *tracking space* GameObject, dapat ditemukan sebuah *anchor* di *center eye*, yang merupakan kamera utama Unity, dua *anchor object* untuk setiap mata kiri dan kanan, serta *anchor* kiri dan kanan untuk *controller*. Gambar 4.1 adalah gambar untuk *hierarchy* dari *prefab* OVRCameraRig.

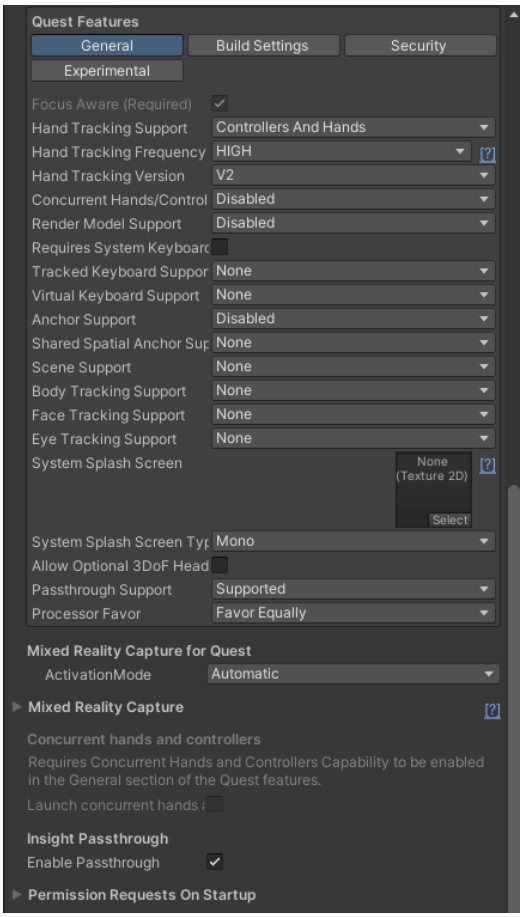


Gambar 4.1 Tampilan *hierarchy prefab* OVRCameraRig

Di dalam *prefab* ini, terdapat *script* utama sebagai pengontrol yang bernama OVRManager. Gambar 4.2 dan Gambar 4.3 adalah gambar dari *script* tersebut.



Gambar 4.2 Komponen *script* OVRManager



Gambar 4.3 Lanjutan komponen *script* OVRManager

Kegunaan dari *script* tersebut adalah untuk mengatur segala yang diperlukan dalam pelacakan menggunakan VR Headset milik Meta. Dalam proyek aplikasi XR bedah rahang, komponen-komponen utama yang mengatur pelacakan antara lain adalah *target devices*, di mana Meta Quest 3 dipilih sebagai VR Headset yang digunakan selama pengembangan.

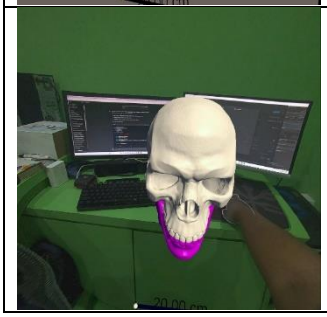
Selanjutnya, terdapat Tracking Origin Type yang berfungsi untuk mengatur ketinggian dan ruang pelacakan ketika menggunakan VR Headset, baik berdasarkan mata, dari lantai, atau dari panggung tempat pengguna berada. Pada Gambar 4.3 di bagian Hand Tracking Support, terdapat tiga jenis *controller* yang didukung, yaitu:

1. **Controller only**, yang hanya menggunakan *controller* milik Meta sebagai media interaksi dan *tracking*.
2. **Hands only**, yang hanya menggunakan *hand tracking* sebagai media interaksi dan *tracking*.
3. **Controller and hands**, yang memungkinkan pengguna untuk beralih antara *controller* dan *hand tracking* sebagai media interaksi dan *tracking*.

Komponen lain yang memungkinkan pengguna untuk mengaktifkan dan menonaktifkan mode *passthrough* terletak di Passthrough Support. Dalam proyek ini, opsi *supported* telah diatur untuk memungkinkan pengguna mengaktifkan dan menonaktifkan mode tersebut. Selain itu, terdapat pilihan *required*, di mana lingkungan virtual harus berada dalam mode tersebut, dan pilihan *none*, di mana lingkungan virtual tidak dapat berada dalam mode tersebut. Namun, untuk menggunakan *passthrough*, diperlukan *scipt* tambahan yang dapat dipasang di *prefab* ini

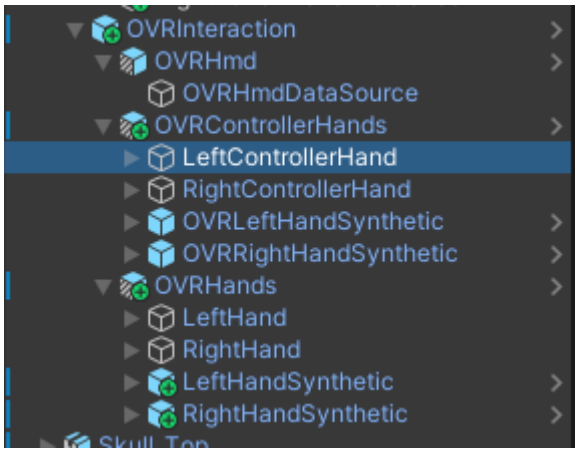
bernama OVRPassthroughLayer, yang digunakan sebagai *layering* objek ketika berada dalam mode *passthrough*. Tabel 4.1 merupakan perbandingan dari tampilan aplikasi dengan menggunakan *passthrough* dan tidak menggunakan *passthrough*.

Tabel 4.1 Perbandingan Tampilan dalam Mode Passthrough dan Non-Passthrough

Gambar	Mode	Deskripsi
	*Non-passthrough*	Tampilan di dalam VR Headset tertutup sepenuhnya dengan lingkungan virtual.
	*Passthrough*	Tampilan di dalam VR Headset tidak tertutup oleh lingkungan virtual, namun hanya menampilkan *object* penting saja.

4.1.1 Implementasi Hand Tracking

Di dalam *hierarchy* OVRCameraRig, terdapat *prefab* lain bernama OVRInteraction sebagai *child*. Tugas dari *prefab* tersebut adalah untuk mengurus semua yang berkaitan dengan interaksi. Dapat dilihat pada Gambar 4.4 adalah gambar *hierarchy* OVRInteraction.

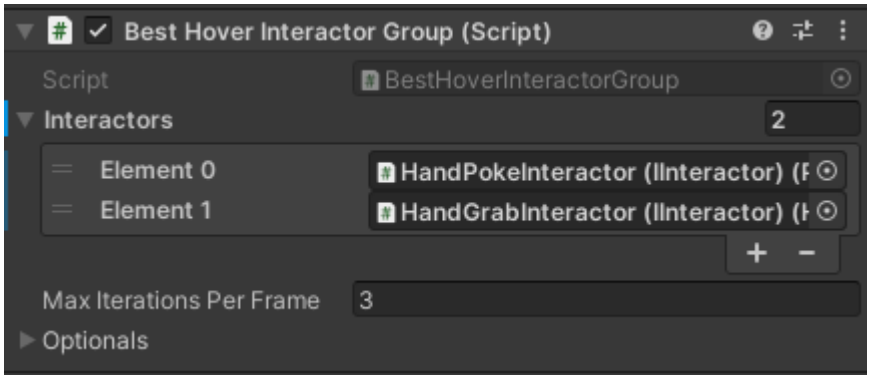


Gambar 4.4 Tampilan *hierarchy* dari OVRInteraction

Di bawah OVRInteraction, terdapat masing-masing *prefab* juga bernama OVRControllerHands dan OVRHands. Di masing-masing *prefab* tersebut, terdapat sebuah GameObject kosong yang menagani berbagai jenis interaksi tangan kanan dan kiri untuk *controller* dan *hand tracking* mulai dari *grab, pinch, ray, poke,* dan lain-lain. Karena aplikasi XR ini berfokus pada penggunaan *hand tracking*, maka di dalam LeftHand dan RightHand, terdapat GameObject bernama HandInteractorLeft dan HandInteractorRight yang dapat dilihat pada Gambar 4.5.

Gambar 4.5 Terdapat komponen *interactors* di masing - masing tangan

Di dalam HandInteractors, terdapat sebuah *scipt* untuk menangani interaksi apa saja yang bisa digunakan. *Script* tersebut bernama BestHoverInteractionGroup, dimana *script* tersebut akan mengelompokkan semua interaksi yang dipasang di tangan. Contoh *script* terdapat pada Gambar 4.6.



Gambar 4.6 Komponen *script* untuk mengelompokkan jenis *interactors*

Pada gambar tersebut, terdapat dua jenis interaksi yang dikelompokkan yaitu *poke* dan *grab*. Interaksi-interaksi tersebut terlebih dahulu dipasang di *script* tersebut. Untuk mendapatkan interaksi-interaksi yang diinginkan, maka digunakan *prefabs* interaksi yang telah disediakan oleh Oculus Interaction SDK. *Prefab* interaksi tersebut kemudian dijadikan sebagai *child* dari GameObject baik HandInteractorLeft maupun HandInteractorRight. Sebagai contoh, pada Gambar 4.7, terdapat dua *prefab* interaksi yaitu HandPokeInteractor dan HandGrabInteractor pada HandInteractorLeft.

Gambar 4.7 Dua *interactors* yang terdapat pada HandInteractorLeft

Sesuai dengan nama mereka, masing-masing *interactor* berfungsi untuk *poke* pada HandPokeInteractor dan *grab* pada HandGrabInteractor.

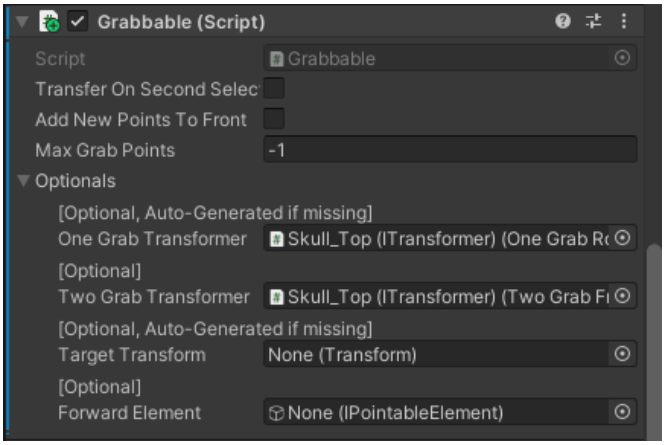
4.1.2 Implementasi Grabbable

Jika pada tangan memiliki komponen *interactor*, maka pada objek yang ingin bisa dilakukan interaksi diberi komponen *interactable*. Untuk membuat objek tulang, penggaris, dan plane dapat dipegang, diperlukan komponen *interactable* bernama *grabbable*. Komponen *grabbable*

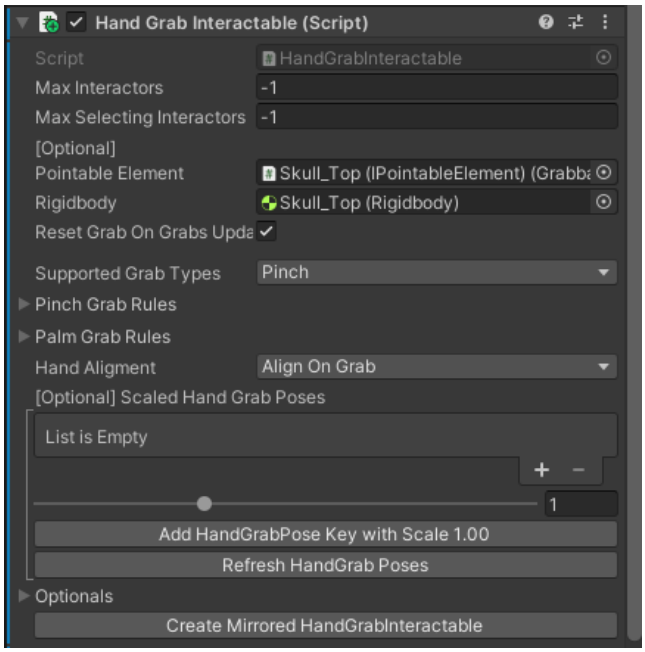
memungkinkan GameObject berputar, diskalakan, atau ditransformasi saat pengguna berinteraksi dengannya. *grabbable* juga menangani perhitungan kecepatan jika pengguna melempar suatu benda. Untuk menentukan bagaimana GameObject harus diputar, diskalakan, atau diubah, *grabbable* menggunakan komponen transformator. Komponen transformator dibagi menjadi dua kategori utama:

1. **One-hand Grab Transformers**, yang hanya menggunakan satu tangan. Kategori ini memiliki beberapa tipe transformasi, antara lain:
    a. **OneGrabFreeTransformer**, yaitu *transformer* dasar yang memungkinkan pengguna merotasi dan mentranslasi objek secara bebas dengan satu tangan.
    b. **OneGrabTranslateTransformer**, yaitu *transformer* yang memungkinkan objek hanya digeser (translasi) pada sumbu x, y, atau z.
    c. **OneGrabRotateTransformer**, yaitu *transformer* yang memungkinkan objek hanya dirotasi berdasarkan pivot point yang diatur. *Pivot point* dapat berupa GameObject kosong, dengan sumbu rotasi berdasarkan *pivot point* tersebut, tergantung di mana GameObject dipasang. Rotasi dapat berdasarkan *world transform* atau *local transform*.
2. **Two-hand Grab Transformers**, yang membutuhkan dua tangan. Kategori ini memiliki beberapa tipe transformasi, antara lain:
    a. **TwoGrabFreeTransformer**, yaitu *transformer* dasar yang memungkinkan pengguna merotasi, mentranslasi, dan menskalakan objek secara bebas dengan kedua tangan.
    b. **TwoGrabRotateTransformer**, yaitu *transformer* yang memungkinkan objek hanya dirotasi berdasarkan *pivot point* yang diatur. Perbedaannya dengan OneGrabRotateTransformer adalah pengguna memegang objek dengan kedua tangan untuk kemudian dapat dirotasi.

Dalam aplikasi XR ini, misalnya, GameObject tulang tengkorak menggunakan dua *transform grabbable*, yaitu OneGrabRotateTransformer dan TwoGrabFreeTransformer. Namun, ada komponen *grabbable* lain yang harus dipasang pada GameObject tulang, yaitu skrip Grabbable dan HandGrabInteractable. *Script* Grabbable mengatur jenis *grab* yang ada pada sebuah GameObject dan akan mengatur OneGrabRotateTransformer serta TwoGrabFreeTransformer. Sementara itu, *script* HandGrabInteractable memungkinkan GameObject dapat dipegang. Komponen ini juga dapat mengatur jenis *grab* yang didukung, seperti *pinch*, *palm*, atau *everything*.

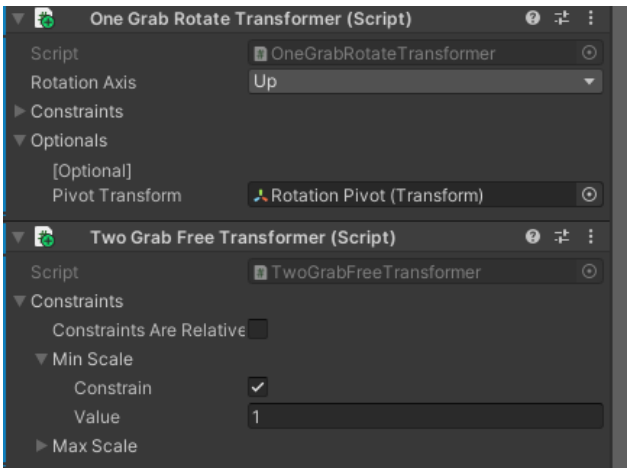


Gambar 4.8 Komponen script Grabbable pada tulang tengkorak



Gambar 4.9 Komponen *script* HandGrabInteractable pada tulang tengkorak

Pada Gambar 4.9 komponen Grabbable telah dipasangi dua *transformer*. Pada *field* One Grab Transformer, terpasang komponen OneGrabRotateTransformer, sedangkan pada *field* Two Grab Transformer, terpasang komponen TwoGrabFreeTransformer. Jadi, ketika pengguna memegang tengkorak dengan satu tangan, mereka hanya dapat merotasi tengkorak. Namun, ketika pengguna memegang tengkorak dengan kedua tangan, mereka bebas untuk merotasi, menggeser, serta memperbesar atau memperkecil tengkorak. Pada Gambar 4.9 komponen HandGrabInteractable, perlu diperhatikan bahwa GameObject tengkorak harus memiliki komponen RigidBody dan *script* Grabbable agar dapat dipegang. Selain itu, GameObject tengkorak hanya dapat dipegang ketika pengguna mencubitnya, karena tipe *grab* yang diatur adalah *pinch*.



Gambar 4.10 Komponen Grab Transformers pada tulang tengkorak

Pada Gambar 4.10 OneGrabRotateTransformer memiliki Rotation Axis Up, yaitu rotasi akan berdasarkan sumbu y, sehingga objek hanya berotasi secara horizontal terhadap sumbu y. Selain itu, *pivot point* dipasang pada *field* Pivot Transform. Pada komponen TwoGrabFreeTransform, tengkorak memiliki batasan bahwa skala minimum hanya bernilai 1 dan tidak dapat dikurangi menjadi, misalnya, 0.5.

4.1.3 Implementasi Hand Pose

Interaksi yang paling banyak digunakan di aplikasi ini salah satunya adalah *hand pose*. Untuk menggunakan *hand pose*, maka perlu dibuat sebuah *empty* GameObject di dalam Unity sebagai wadah untuk menangani *hand pose* dengan memasang berbagai macam komponen di dalam *inspector*. *Hand pose* dari Oculus menggunakan sebuah *scriptable object* untuk membuat berbagai macam pose yang diinginkan dengan cara merekognisi sebuah *shape* yang dibuat di dalam pengaturan *scriptable object* tersebut. Di dalam *scriptable object* tersebut, terdapat konfigurasi untuk masing-masing jari. Masing-masing jari memiliki empat fitur yang dapat digunakan untuk menggambarkan bentuknya: *curl*, *flexion*, *abduction*, dan *opposition*. Keempat fitur ini memiliki tiga *state* seperti *open*, *neutral*, dan *closed*, namun ada juga *states* berbeda tergantung jenis fitur yang digunakan. Berikut ini penjelasan lebih detail mengenai bentuk-bentuk atau fitur jari beserta *state* yang dimiliki.

1. **Curl**, merepresentasikan seberapa bengkoknya dua sendi teratas jari tangan atau ibu jari. Fitur ini tidak mempertimbangkan sambungan proksimal (*knuckle*). *States* dari fitur *curl* antara lain:
    a. **Open**, jari tidak tertutup atau menekuk sama sekali atau lurus sepenuhnya.
    b. **Neutral**, jari sedikit tertutup atau menekuk kedalam, seperti sedang memegang cangkir kopi.
    c. **Closed**, jari sepenuhnya tertutup atau menekuk sampai ujung jari seperti hampir menyentuh telapak tangan.

    Contoh pose dalam kondisi *closed* dapat dilihat pada Gambar 4.11.

Gambar 4.11 Contoh *curl* dalam keadaan *closed*, sendi menekuk hingga kedalam

2. **Flexion**, sejauh mana sendi proksimal (*knuckle*) tertekuk relatif terhadap telapak tangan. *Flexion* hanya dapat diandalkan pada 4 jari saja. Hal ini dapat ibu jari kebebasan dalam menentukan fitur atau bentuk yang digunakan. *States* dari fitur *flexion* antara lain:
    a. **Open**, tulang pertama pada jari terentang penuh dan sejajar dengan telapak tangan.
    b. **Neutral**, jari – jari sedikit menekuk.
    c. **Closed**, sendi *knuckle* sepenuhnya tertekuk.

Contoh pose dalam kondisi *closed* dapat dilihat pada Gambar 4.12. Catatan, *flexion* hanya dapat diandalkan pada empat jari saja selain ibu jari, sehingga ibu jari dapat bebas diberi fitur apapun tanpa memengaruhi *flexion*.

Gambar 4.12 Contoh *flexion* dalam keadaan *closed*, sendi menekuk sepenuhnya

3. **Abduction**, sudut antara dua jari yang berdekatan, diukur pada pangkal kedua jari tersebut. *Abduction* mengukur sudut antara jari tertentu dan jari di sebelahnya yang lebih dekat ke kelingking. Misalnya, *abduction* jari telunjuk adalah sudut antara jari telunjuk dan jari tengah. Untuk memahami *states* dari fitur *abduction*, dapat dilihat pada penjelasan di bawah beserta Gambar 4.13.
    a. **Open**, dua jari terpisah (digambarkan oleh jari telunjuk).
    b. **Closed**, dua jari menyentuh secara rapat (digambarkan oleh ibu jari, jari tengah, dan jari manis).
    c. **None**, saat ini tidak digunakan.

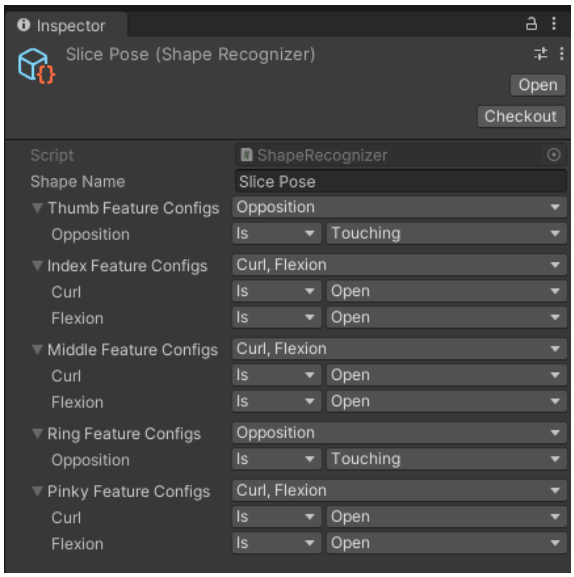
Gambar 4.13 Contoh *abduction*, jari manis pada keadaan *open* sementara jari lain pada keadaan *closed*

Catatan, fitur *abduction* tidak dapat diaplikasikan pada jari kelingking.

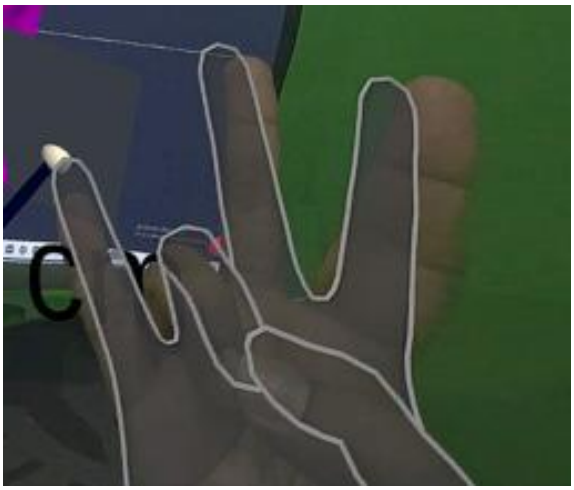
4. **Opposition**, seberapa dekat ujung jari tertentu dengan ujung ibu jari. Hanya bisa digunakan pada jari telunjuk, jari tengah, jari manis, dan jari kelingking. Untuk memahami *states* dari fitur *opposition*, dapat dilihat pada penjelasan di bawah beserta Gambar 4.14.
    a. **Touching**, ujung jari yang menyentuh berada dalam jarak ~1,5 cm (digambarkan sebagai indeks).
    b. **Near**, ujung jari yang menyentuh berada dalam jarak ~1,5 cm dan ~15 cm.
    c. **None**, ujung jari berada dalam jarak >15 cm.

Gambar 4.14 Contoh *opposition*, jari telunjuk dalam keadaan *touching*

Contoh *scriptable object* yang digunakan dalam aplikasi XR bedah rahang adalah pose untuk *Slice*. Pada Gambar 4.15 adalah beberapa konfigurasi yang digunakan dan Gambar 4.16 adalah tampilan dari pose yang dibuat.



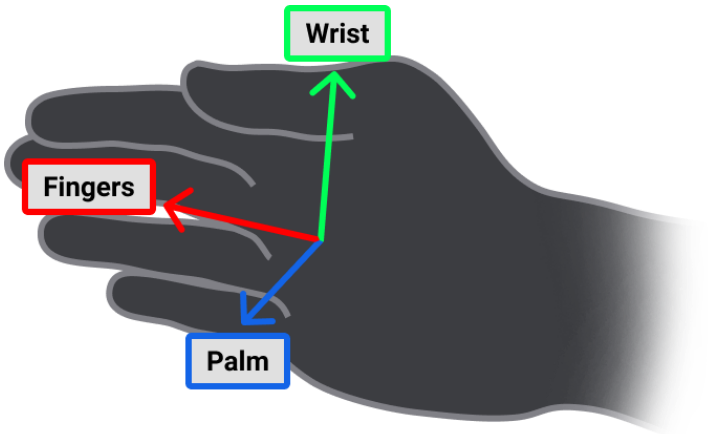
Gambar 4.15 Konfigurasi komponen *scriptable object shape recognizer* Slice Pose



Gambar 4.16 Tampilan dari pose *Slice*

Pose tersebut dibuat berdasarkan konfigurasi seperti ibu jari dan jari manis menggunakan fitur *opposition* dengan *state touching*. Kemudian jari telunjuk, jari tengah, dan jari kelingking menggunakan dua fitur yaitu *curl* dan *flexion* dalam kondisi *open* di kedua fitur, yang menunjukkan bahwa jari-jari tersebut harus dalam keadaan lurus dan tidak tertekuk sedikitpun.

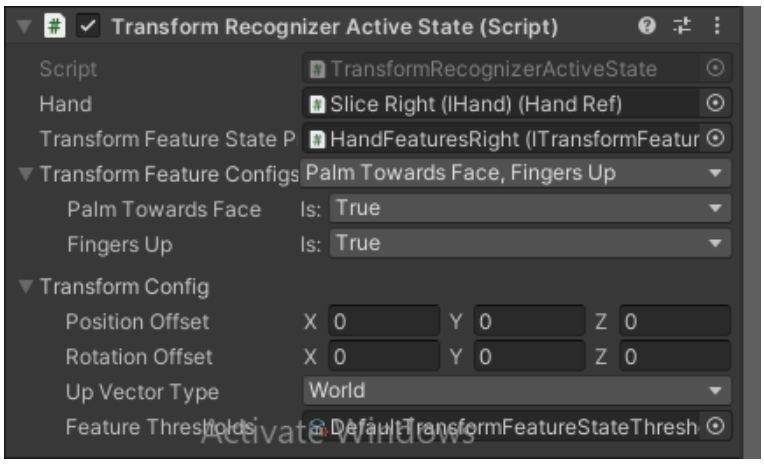
Selanjutnya adalah Transform Recognition, yaitu fitur atau komponen yang digunakan untuk mengidentifikasi transformasi pose. Di dalam Transform Recognition, terdapat komponen lain bernama TransformRecognizerActiveState yang berfungsi mencatat transformasi yang diperlukan untuk pose tertentu; transformasi pada tangan hanya mewakili orientasi dan posisi. Evaluasi orientasi hanya dilakukan terhadap pose Wrist Up, Wrist Down, Palm Up, Palm Towards Face, Palms Away From Face, Finger Up, Finger Down, dan Pinch Clear. Dalam pelacakan tangan, transformasi tangan dan pose berbeda. Pose terdeteksi jika kedua set transformasi cocok dan semua bentuk dalam pose aktif. Gambar 4.17 adalah gambar sumbu yang mendefinisikan transformasi untuk tangan.



Gambar 4.17 Sumbu yang mendefinisikan transformasi *Wrist*, *Fingers*, dan *Palm*

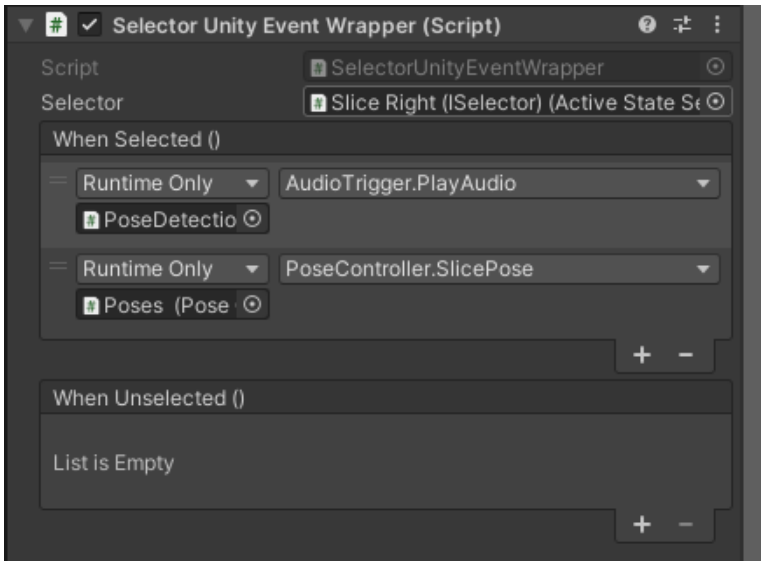
Salah satu penggunaan komponen TransformRecognizerActiveState adalah pada pose *slice* di tangan kanan. Seperti terlihat pada Gambar 4.18 Transform Feature Configs adalah konfigurasi yang mengatur transformasi tangan. Pada pose *slice*, konfigurasi diatur menjadi Palm Towards Face dan Fingers Up. Palm Towards Face diatur menjadi *true* karena, seperti terlihat pada Gambar 4.16 telapak tangan menghadap ke arah wajah pengguna. Kemudian, Fingers Up diatur menjadi *true* karena pada Gambar 4.16, semua jari yang tidak melakukan fitur *opposition*

menghadap ke atas. Transformasi ini bertujuan untuk membuat batasan agar pengguna tidak secara tidak sengaja mengeksekusi perintah saat menggerakkan tangan mereka, dan hanya dapat mengeksekusi perintah ketika memenuhi kriteria batasan tersebut. Untuk membuat Transform Feature Configs bekerja, maka dibutuhkan Feature Thresholds yang dapat dimasukkan langsung ke dalam TransformRecognizerActiveState karena telah disediakan oleh Oculus Interaction SDK.



Gambar 4.18 Komponen Transform Recognizer Active State

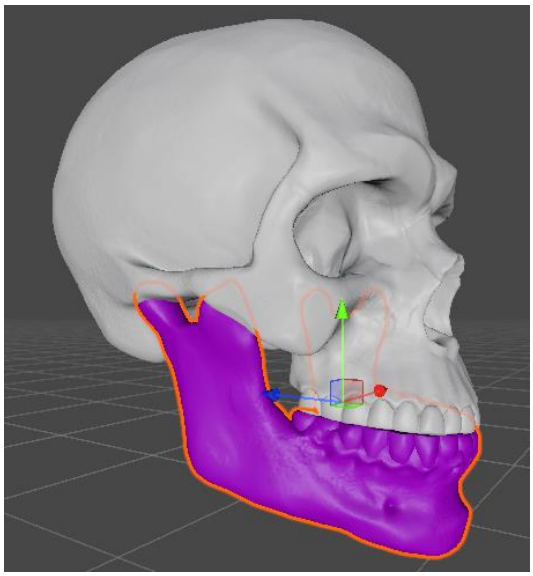
Untuk mengeksekusi perintah seperti *slice*, memunculkan *plane*, dan lain-lain, maka diperlukan komponen bernama SelectorUnityEventWrapper. Ketika pengguna memeragakan pose, maka komponen tersebut akan memanggil *script* yang dipasang di komponen tersebut. Contoh pada Gambar 4.19 komponen SelectorUnityEventWrapper memiliki dua buah *script* yang akan dieksekusi ketika pose diperagakan yaitu untuk memainkan *audio* dan mengeksekusi *script slice*.



Gambar 4.19 Komponen Selector Unity Event Wrapper

4.2 Implementasi Tulang Tengkorak

Tulang tengkorak yang digunakan dalam aplikasi XR bedah rahang terdiri dari dua model 3D. Model pertama bernama Skull_Top, yaitu model seluruh tengkorak namun hanya sampai rahang atas. Model kedua bernama Skull_Mandible, yaitu model rahang bawah. Gambar 4.20 menunjukkan model Skull_Top dan Skull_Mandible.



Gambar 4.20 Model tengkorak dengan dua model terpisah yaitu Skull_Top dan Skull_Mandible

Pada gambar tersebut, Skull_Mandible ditandai dengan garis highlight berwarna oranye yang menandakan bahwa GameObject tersebut terpisah dari GameObject Skull_Top. Skull_Mandible diatur sebagai *child* dari Skull_Top karena interaksi *grabbable* dipasang pada GameObject ini, dengan tujuan membuat kedua objek menjadi satu kesatuan. Jika *grabbable* dipasang pada Skull_Mandible, kedua GameObject akan berperilaku seolah – olah terpisah. Misalnya, jika pengguna merotasi Skull_Top, Skull_Mandible tidak akan ikut berotasi, dan sebaliknya. Model Skull_Mandible diberi *tag* "Mandible" untuk pengecekan tumbukan atau *collider* dengan *pointer*. Sedangkan model Skull_Top diberi *tag* “Skull” untuk memudahkan menjadikan hasil potongan menjadi *child* dari Skull_Top. Selain itu, ada penggaris yang diatur sebagai *child* dari tengkorak, sehingga penggaris akan tetap bersama tengkorak jika tengkorak digerakkan.

4.3 Implementasi Penggaris

Penggaris digunakan sebagai alat ukur untuk mengukur di mana pengguna akan memunculkan *plane*. GameObject penggaris terdiri dari dua buah *sphere* dan sebuah *line renderer* yang menghubungkan kedua ujung. Pada kedua *sphere*, diberikan komponen *script* Grabbable dan HandGrabInteractable sehingga kedua ujung dapat dipegang dan ditaruh pada mandibula sesuai keinginan pengguna. Untuk tampilan teks panjang penggaris menggunakan UI dari Unity yaitu Text Mesh Pro. Implementasi penggaris dapat dilihat pada Kode Sumber 4.1 berikut.

Ruler.cs

```
01. Method Update():
02.     Set the start position of lineRenderer to startSphere.position
03.     Set the end position of lineRenderer to endSphere.position
04.
05.     distance = Calculate the distance between startSphere.position and
endSphere.position multiplied by 100
06.      Set distanceText.text to distance formatted to 2 decimal places
followed by " cm"
07.
08.     midPoint = Calculate the midpoint between startSphere.position and
endSphere.position
09.     Set distanceText.transform.position to midPoint
10.
11.     // Optionally, make the text face the camera
```

```
12.          Set distanceText.transform.rotation to Look rotation from
distanceText.transform.position to Camera.main.transform.position
13. End Method
```

Kode Sumber 4.1 *Pseudocode* implementasi penggaris

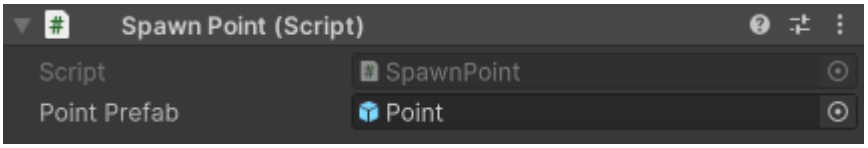
Script akan mengatur posisi awal pada posisi `startSphere` dan posisi akhir pada `endSphere`. Kemudian *script* akan mengalkulasi jarak antara *spehere* dan dikali 100 untuk mendapatkan jarak dalam sentimeter. Kemudian jarak akan dimasukkan ke dalam Text Mesh Pro dengan format dua desimal dan dibelakang diberi label cm. GameObject penggaris dijadikan sebagai *child* dari mandibula sehingga akan tetap menempel, bahkan berubah ketika tengkorak diperbesar atau diperkecil, namun skala dari jarak akan ikut berubah juga.

4.4 Implementasi Pointer

Pointer digunakan sebagai penanda dimana pengguna akan memunculkan GameObject *point*, lalu kemudian dari *point* tersebut, muncullah *plane* yang digunakan untuk *slice* mandibula. Di dalam *hiearchy project* Unity, *pointer* ditempatkan sebagai *child* dari *prefab* HandsSynthetic yang sudah ada pada OVRCameraRig baik itu di tangan kiri dan kanan. HandsSynthetic berisi *joints* dan jari-jari dari *hand tracking*. *Pointer* sendiri ditempatkan sebagai *child* dari GameObject *r_index_finger_tip_marker* karena GameObject tersebut adalah sebagai ujung jari telunjuk seperti yang terlihat pada Gambar 4.21.

Gambar 4.21 Tampilan *hierarchy* Right Hand Synthetic beserta *pointer* sebagai *child index finger*

Komponen-komponen pada *pointer* adalah SphereCollider, RigidBody, dan sebuah *script* bernama `SpawnPoint.cs`. SphereCollider dan RigidBody digunakan dalam *script* `SpawnPoint.cs` untuk mengecek apakah *pointer* bertumbukan dengan objek yang memiliki *tag* "Mandible", yaitu Skull_Mandible dalam aplikasi ini, dan kemudian memunculkan *point*. *Prefab point* dimasukkan ke dalam *field* Point Prefab, seperti terlihat pada Gambar 4.22, agar GameObject tersebut dapat dipanggil dan dimunculkan.



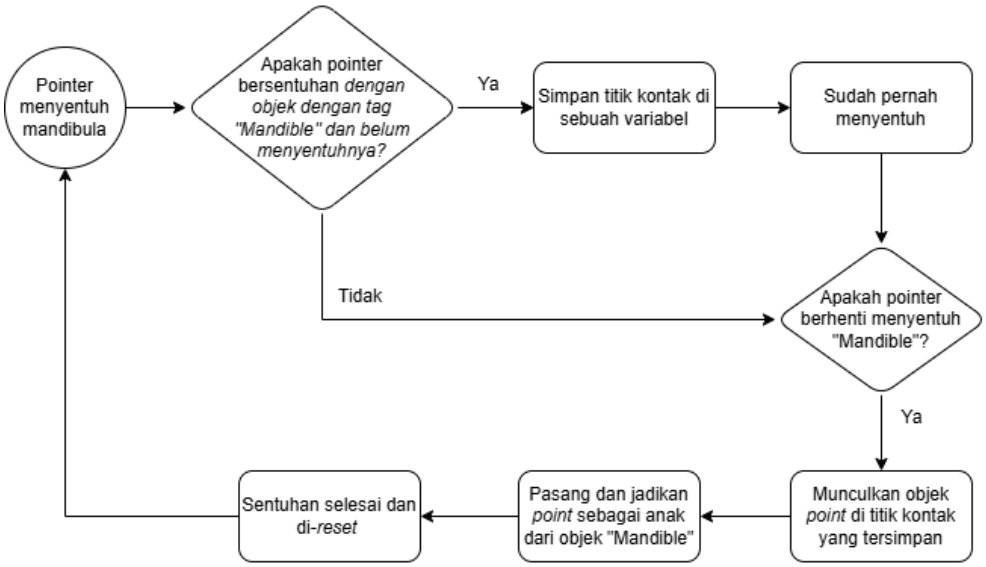
Gambar 4.22 Komponen *script* SpawnPoint pada *pointer*

Cara kerja *script* `SpawnPoint.cs` dapat dilihat pada Kode Sumber 4.2 dan Gambar 4.23 berikut.

SpawnPoint.cs

```
01. Class SpawnPoint extends MonoBehaviour:
02.     Public Attributes:
03.         pointPrefab: GameObject
04.     Private Attributes:
05.         contactPoint: Vector3
06.         storedPoint: GameObject
07.         hasCollided: Boolean = false
08.
09.     Method OnCollisionEnter(collision: Collision):
10.         If hasCollided is False AND collision.gameObject has tag
"Mandible":
11.             contact: ContactPoint = collision.contacts[0]
12.             contactPoint = contact.point
13.             hasCollided = True
14.         End If
15.     End Method
16.
17.     Method OnCollisionExit(collision: Collision):
18.         If hasCollided is True AND collision.gameObject has tag
"Mandible":
19.             storedPoint = Instantiate pointPrefab at contactPoint
with no rotation
20.             Set storedPoint's parent to
collision.gameObject.transform
21.             hasCollided = False
22.         End If
23.     End Method
24. End Class
```

Kode Sumber 4.2 *Pseudocode* implementasi *script* SpawnPoint



Gambar 4.23 Flowchart implementasi script SpawnPoint

Pada *method* `OnCollisionEnter()`, ketika *pointer* menyentuh atau *collide* dengan "Mandible" dan belum pernah menyentuh sebelumnya, titik kontak dari sentuhan akan disimpan di sebuah variabel. Selanjutnya, *flag boolean* `hasCollided` yang awalnya bernilai

false diubah menjadi *true*. Pada *method* `OnCollisionExit()`, ketika *pointer* berhenti menyentuh atau berhenti bertumbukan dengan "Mandible", *point* akan dimunculkan di lokasi titik kontak tersebut. *Point* yang muncul akan dijadikan *child* dari "Mandible" dan kemudian *flag* `hasCollided` diatur ulang menjadi *false*. Proses ini dapat diulang kembali.

Untuk memunculkan *pointer* di ujung jari pengguna, pengguna harus melakukan *hand pose* seperti membentuk huruf L kapital baik dengan tangan kiri maupun tangan kanan. Gambar 4.24 menunjukkan *hand pose* untuk memunculkan *pointer* di ujung jari. Kondisi tangan saat aplikasi dijalankan berada dalam kondisi normal tanpa *pointer*. Ketika pengguna melakukan *hand pose*, *pointer* akan muncul. Jika pengguna melakukan *hand pose* kembali, *pointer* akan hilang. Pose ini dapat dilakukan secara terus-menerus karena berfungsi sebagai *toggle* untuk mengaktifkan dan menonaktifkan *pointer*.

Gambar 4.24 Pose untuk *toggle* pointer

4.5 Implementasi Point

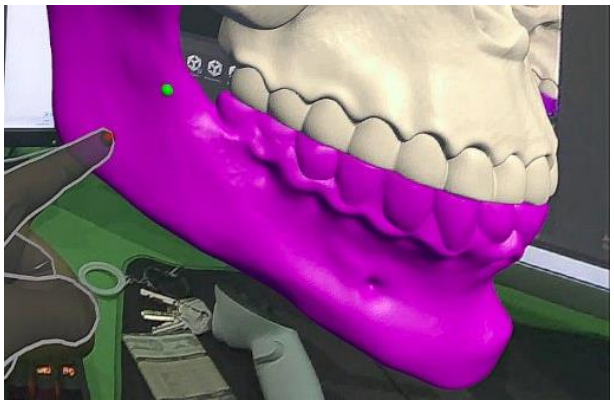
Point adalah GameObject *sphere* berwarna hijau yang digunakan untuk memunculkan *slicing plane* di lokasi tertentu yang ditentukan oleh pengguna melalui sentuhan jari telunjuk pengguna ke mandibula. GameObject *point* akan dijadikan *prefab* karena digunakan secara terus-menerus. Komponen yang ada di dalam *point* hanya mencakup komponen dasar seperti RigidBody dan SphereCollider. Namun, GameObject *point* diberi *tag* "Point" yang akan digunakan dalam sebuah *script* bernama `SpawnPlane.cs`. Kode Sumber 4.3 adalah bagian kode yang memungkinkan penggunaan *point* tersebut.

SpawnPlane.cs

```
01. Method Update():
02.     points = Find all game objects with tag "Point"
03.
04.      If points length is greater than or equal to 2 AND lineDrawn is
false:
05.         startPoint = Get the first point from points
06.         endPoint = Get the second point from points
07.
08.         Call SpawnPlaneBetweenPoints
09.         Destroy startPoint
10.         Destroy endPoint
11.     End If
12. End Method
```

Kode Sumber 4.3 *Pseudocode method* yang menangani *point*

Fungsi dari potongan kode tersebut adalah sebagai berikut, pada *method* `Update()`, kode akan mencari semua GameObject yang memiliki *tag* "Point" setiap *frame*. Jika ada dua GameObject dengan *tag* "Point" di *scene*, maka *point* pertama akan dimasukkan ke dalam sebuah *array* dan menjadi urutan pertama dengan nama `startPoint`. Selanjutnya, *point* kedua akan dimasukkan ke dalam *array* dan menjadi urutan kedua serta terakhir dengan nama `endPoint`. Gambar 4.25 adalah contoh tampilan ketika satu buah *point* telah muncul dan menjadi *child* dari GameObject dengan *tag* “Mandible”.



Gambar 4.25 Tampilan mandibula dengan satu buah *point* terpasang

Ketika kedua *point* telah muncul, maka akan memanggil *method* yang ada di *script* `SpawnPlane`.cs yaitu `SpawnPlaneBetweenPoints()` yang akan memunculkan *plane* di tengah-tengah dua ujung *point*. Berikut adalah *method* `SpawnPlaneBetweenPoints()` yang dapat dilihat pada Kode Sumber 4.4.

SpawnPlane.cs

```
01. Method Update():
02.     points = Find all GameObjects with tag "Point"
03.
04.     If points.Length >= 2:
05.         startPoint = points[0]
06.         endPoint = points[1]
07.
08.         Call SpawnPlaneBetweenPoints()
09.         Destroy startPoint
10.         Destroy endPoint
11.     End If
12. End Method
```

Kode Sumber 4.4 *Pseudocode method* untuk memunculkan *plane* pada interaksi pertama

Pada interaksi pertama, fungsi *point* yang dimunculkan oleh *pointer* adalah untuk memunculkan *slicing plane* di antara dua *point*. *Method* `Update()` tersebut berfungsi untuk memunculkan *slicing plane* di tengah-tengah jarak antara dua *point* di tiap *frame*. *Point*s yang dimunculkan memiliki *tag* “Point” untuk mengecek jarak antar objek dengan *tag* tersebut. Jika jarak *point* lebih dari dua, maka *point* yang muncul pertama akan ditetapkan sebagai `startPoint` dan dimasukkan ke dalam *array* pertama. Kemudian *point* yang muncul kedua akan ditetapkan sebagai `endPoint` dan dimasukkan ke dalam *array* kedua. Setelah kedua *point* muncul, panggil *method* `SpawnPlaneBetweenPoints()` untuk memunculkan *slicing plane* di tengah-tengah jarak kedua *point*. Ketika *slicing plane* telah muncul, maka hancurkan semua *point* sehingga pengguna dapat memunculkan dua *point* yang lain untuk

memunculkan *slicing plane* selanjutnya.

Pada interaksi kedua, perbedaan penggunaan *point* dari interaksi pertama adalah *point* yang digunakan hanya satu untuk memunculkan *slicing plane*. *Plane* yang muncul akan berada tepat dimana *point* juga dimunculkan. Berikut ini adalah perbedaan mendasar isi *script* pada interaksi kedua yang digunakan pada untuk memunculkan *point* yang dapat dilihat pada Kode Sumber 4.5.

SpawnPlane_V2.cs

```
01. Method Update():
02.     points = Find all GameObjects with tag "Point"
03.
04.     For each point in points:
05.         If spawnedPoints does not contain point:
06.             Call SpawnPlaneAtPoint with point.transform.position
07.             Add point to spawnedPoints
08.             Destroy point
09.         End If
10.     End For
11. End Method
```

Kode Sumber 4.5 *Pseudocode method* untuk memunculkan *plane* pada interaksi kedua

Jika pada *script* `SpawnPlane.cs` terdapat perhitungan untuk menghitung jarak antara *point* untuk memunculkan *plane* di tengah-tengah, maka pada *script* `SpawnPlane_V2`.cs, yang digunakan pada interaksi kedua, tidak terdapat perhitungan serupa. *Method* `Update()` tetap memberikan objek *point tag* "Point", namun gunanya adalah untuk memunculkan *plane* pada GameObject yang memiliki *tag* tersebut. Untuk setiap *point* yang muncul, jika setiap *point* yang dimunculkan belum ada pada *lsit point* yang dimunculkan, maka munculkan *slicing plane* di posisi *point* berada dan masukkan *point* yang muncul ke dalam *list* agar *point* yang telah muncul tidak diproses lagi kedepannya. Namun, ketika *slicing plane* telah muncul, hancurkan *point* pada *scene*. *List* yang telah dibuat berguna agar *point* dapat dimunculkan berkali-kali.

Interaksi ketiga tidak menggunakan *point* untuk memunculkan *slicing plane*, namun menggunakan *plane* lain bernama *ghost plane* yang terdapat pada ujung jari telunjuk pengguna.

4.6 Implementasi Plane

Plane di dalam aplikasi XR bedah rahang memiliki dua fungsi antara lain *plane* sebagai alat *slice* dan *plane* transparan sebagai penanda dan "hantu" untuk mendefinisikan dimana lokasi *plane* pemotong nanti akan dimunculkan. *Plane* pemotong akan dinamai *Slicing Plane* sedangkan *plane* penanda akan dinamai *Ghost Plane*.

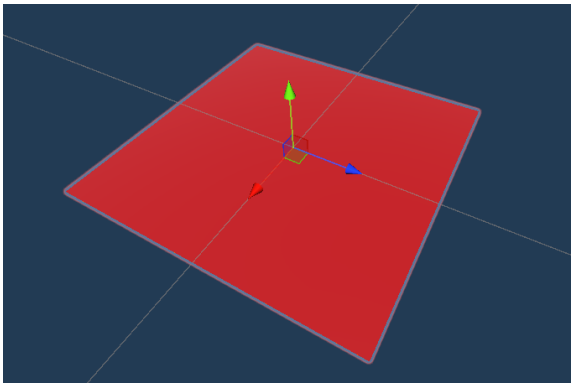
4.6.1 Implementasi Slicing Plane

Slicing plane, sesuai namanya, adalah *prefab* yang digunakan sebagai alat pemotong pada mandibula. Dalam aplikasi ini, diperlukan dua buah *slicing plane* untuk melakukan pemotongan pada mandibula. Komponen-komponen yang digunakan di dalam *plane* ini meliputi MeshCollider, RigidBody, Grabbable, HandGrabInteractable, dan sebuah *script* buatan bernama `PlaneSlice_EzySlice.cs`. Terdapat dua jenis *slicing plane* berdasarkan jenis transformasinya antara lain:

1. **Normal Slicing Plane**, dimana *plane* ini dapat dipegang dan dipindahkan atau dirotasi kemana saja. *Plane* jenis ini digunakan pada interaksi pertama dan interaksi ketiga.

2. **Fixed Slicing Plane**, dimana perbedaan dari normal plane hanya pada kemampuan untuk dipindahkan. *fixed slicing plane* tidak dapat dipindahkan namun hanya bisa dirotasi terhadap sumbu z. *Plane* jenis ini hanya digunakan pada interaksi kedua.

Grabbable dan HandGrabInteractable digunakan dengan tujuan yang sama seperti sebelumnya, agar *plane* dapat dipegang dan dipindahkan ke mana saja. Namun, *fixed slicing plane* hanya bisa dirotasi karena komponen OneGrabRotateTransformer terpasang di sana. Hal ini dapat dicapai dengan menambahkan komponen OneGrabRotateTransformer pada *plane* tersebut. Gambar 4.26 adalah tampilan dari *slicing plane* yang digunakan dalam aplikasi.



Gambar 4.26 Slicing Plane

Selanjutnya *method* untuk memunculkan *plane* terdapat pada *script* `SpawnPlane.cs` yang dapat dilihat pada Kode Sumber 4.6 berikut.

SpawnPlane.cs

```
01. Method SpawnPlaneBetweenPoints():
02.     planePosition = (startPoint.position + endPoint.position) / 2f
03.     direction = endPoint.position - startPoint.position
04.     planeRotation = Quaternion.LookRotation(direction)
05.
06.        planeObject = Instantiate planePrefab at planePosition with
planeRotation
07.     slicingPlane = Get PlaneSlice_EzySlice component from planeObject
08.     planeTransform = Get transform from planeObject
09.
10.     planeTransform.rotation = Look rotation based on direction and up
vector
11.     planeTransform.SetParent(target.transform)
12.     slicingPlane.target = target
13.
14.     If spawnedPlanes count is 0:
15.         Set slicingPlane's firstPlane to planeTransform
16.     Else If spawnedPlanes count is 1:
17.         Rotate planeObject by (180, 0, 0)
18.            Set slicingPlane's firstPlane to firstPlane of the first
slicingPlane
19.         Set slicingPlane's secondPlane to planeTransform
20.         Set secondPlane of the first slicingPlane to planeTransform
21.     End If
22.
23.     Add planeObject to spawnedPlanes
24.     Add slicingPlane to slicingPlanes
25. End Method
```

Kode Sumber 4.6 *Pseudocode method* untuk memunculkan *normal slicing plane*

Slicing plane akan mendapatkan komponen-komponen yang ada pada *script* `PlaneSlice_EzySlice.cs`. Untuk menyinkronkan objek *plane* dengan *script* `PlaneSlice_EzySlice.cs`, yang membutuhkan nilai *transform* dari *slicing plane*, jika *plane* pertama yang muncul, nilai *transform* dari *slicing plane* pertama dimasukkan ke *field* First Plane pada *script* `PlaneSlice_EzySlice.cs`. Jika *plane* kedua telah muncul, nilai *transform* dari objek *plane* tersebut dimasukkan ke dalam *field* Second Plane pada *script* `PlaneSlice_EzySlice.cs`.

Pada interaksi pertama, *plane* yang dimunculkan adalah *normal slicing plane*. Dengan menggunakan dua *point*, *method* `SpawnPlaneAlongLine()` akan menghitung posisi *slicing plane* berdasarkan titik tengah dari kedua *point*. Orientasi *slicing plane* didapat dari selisih antara `startPoint` dan `endPoint`. Selanjutnya, *slicing plane* dimunculkan dengan posisi dan rotasi yang telah ditetapkan dari perhitungan sebelumnya dan dimasukkan ke dalam sebuah *list*. *Slicing plane* kemudian diatur menjadi *child* dari *target*.

Pada interaksi kedua, *plane* yang dimunculkan adalah *fixed slicing plane*, perbedaannya hanya pada proses memunculkan *plane* bahwa alih-alih harus menggunakan dua *point* terlebih dahulu, interaksi hanya menggunakan satu *point* dan *plane* akan dimunculkan tepat dimana *point* tersebut muncul. Yang membedakan adalah orientasi dari *plane*, dimana pada *script* SpawnPlane yang menggunakan direction, *fixed rotation plane* menggunakan rotasi aslinya yaitu `Quaternion planeRotation = Quaternion.identity`. Kemudian *method* yang digunakan untuk memunculkan *fixed rotation plane* juga berbeda, yang dapat dilihat pada Kode Sumber 4.7 berikut.

SpawnPlane_V2.cs

```
01. Method SpawnPlaneAtPoint(pointPosition):
02.     planeRotation = Quaternion.identity // Default rotation
03.
04.        planeObject = Instantiate planePrefab at pointPosition with
planeRotation
05.     slicingPlane = Get PlaneSlice_EzySlice component from planeObject
06.     planeTransform = Get transform from planeObject
07.
08.     planeTransform.rotation = Quaternion.identity // Adjust as needed
09.     planeTransform.SetParent(target.transform)
10.     slicingPlane.target = target
11.
12.     If spawnedPlanes count is 0:
13.         // First plane
14.         slicingPlane.firstPlane = planeTransform
15.     Else If spawnedPlanes count is 1:
16.         // Second plane
17.         planeObject.transform.Rotate(180, 0, 0)
18.         slicingPlane.firstPlane = slicingPlanes[0].firstPlane
19.         slicingPlane.secondPlane = planeTransform
20.         slicingPlanes[0].secondPlane = planeTransform
21.     End If
22.
23.     Add planeObject to spawnedPlanes
24.     Add slicingPlane to slicingPlanes
25. End Method
```

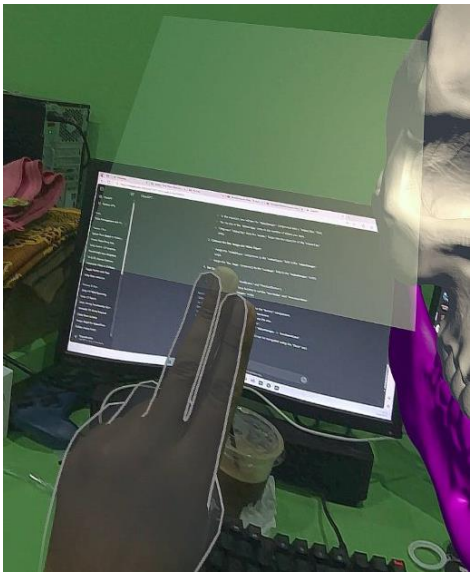
Kode Sumber 4.7 *Pseudocode method* untuk memuncukan *fixed slicing plane*

Jika pada *script* `SpawnPlane.cs` terdapat perhitungan untuk mencari *planePosition* dan

planeRotation, maka pada *script* `SpawnPlane_V2.cs`, yang digunakan pada interaksi kedua, tidak terdapat perhitungan serupa. *Method* `SpawnPlane(pointPosition)` langsung memunculkan *slicing plane* dengan nilai *transform position* yang diambil dari *position point* yang dimunculkan. Kemudian, sisa dari *script* sama dengan SpawnPlane dimana *slicing plane* yang muncul akan dimasukkan ke *transform* pada *script* `PlaneSlice_EzySlice.cs` dan akan dimasukkan ke dalam *list* untuk dapat diurungkan dalam memunculkan *slicing plane*. Interaksi ketiga tidak menggunakan *point* untuk memunculkan *slicing plane*, namun menggunakan sebuah *ghost plane* untuk memunculkannya dengan nilai *transform* sama dengan *ghost plane* tersebut.

4.6.2 Implementasi Ghost Plane

Seperti yang dibahas sebelumnya, pada interaksi ketiga tidak menggunakan *point* untuk memunculkan *slicing plane* melainkan menggunakan sebuah *plane* transparan bernama *ghost plane*. Pengguna akan memunculkan *ghost plane* dengan melakukan *pose* jari telunjuk dan tengah ke atas dan rapat, seperti pada Gambar 4.27**.**



Gambar 4.27 Pose untuk memunculkan *ghost plane*

Sama seperti *pointer*, *plane* ini ditempatkan sebagai *child* dari *prefab* HandsSynthetic yang sudah ada pada OVRCameraRig baik itu di tangan kiri dan kanan. Di dalam objek *ghost plane*, terdapat beberapa komponen yang digunakan untuk dapat memunculkan *slicing plane* antara lain BoxCollider, RigidBody, dan sebuah *script* bernama `SpawnPlane_V3.cs`. RigidBody dan BoxCollider digunakan untuk mengecek apakah objek *ghost plane collide* dengan mandibula atau tidak, apakah *collider* masih bertahan di target, atau *collider* sudah keluar dari target. Pada *script* `SpawnPlane_V3.cs`, untuk dapat memunculkan *slicing plane*, pertama, pengguna seperti menaruh *ghost plane* yang ada di jari mereka ke dalam mandibula. Kemudian, ketika *collider* dari *ghost plane* menetap di dalam mandibula, maka munculkan *slicing plane* tepat di mana *ghost plane* berada. Berikut adalah Kode Sumber 4.8 untuk mengecek apakah *ghost plane* dapat memunculkan *slicing plane.*

SpawnPlane_V3.cs

```
01. Method OnCollisionStay(collision):
02.     If collision.gameObject.CompareTag("Mandible"):
03.         Log "Collide Stay" to the console
04.         hasCollided = True
```

```
05.     End If
06. End Method

07. Method OnCollisionExit(collision):
08.     If collision.gameObject.CompareTag("Mandible"):
09.         hasCollided = False
10.         Log "Exit" to the console
11.     End If
12. End Method
```

Kode Sumber 4.8 *Pseudocode method* untuk mengecek *collider* pada *ghost plane*

Ketika *collider* menetap di mandibula, *flag boolean* `hasCollided` diubah menjadi *true*, namun ketika *collider* keluar dari mandibula, *flag boolean* `hasCollided` kembali diubah menjadi *false*. Gunanya adalah ketika *ghost plane* berada di luar mandibula, pengguna tidak dapat memunculkan *slicing plane* baik disengaja maupun tidak. Kemudian pada *method* `OnCollisionStay()`, ketika pengguna melakukan pose, maka *slicing plane* akan dimunculkan karena tiap *frame*, nilai *transform* dari *ghost plane* akan dicek. Kode Sumber 4.9 adalah *method* untuk memunculkan *slicing plane*.

SpawnPlane_V3.cs

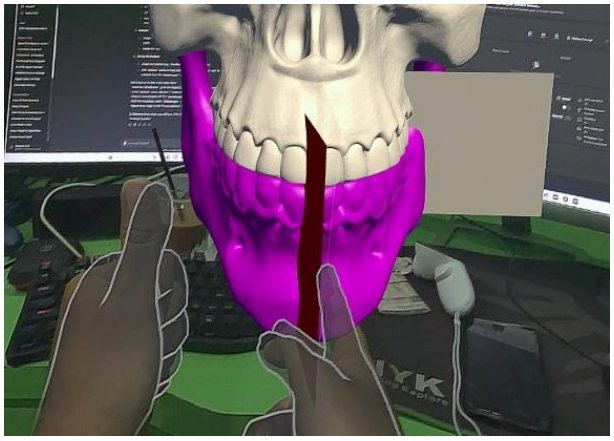
```
01. Method SpawnPlane():
02.     If hasCollided is True:
03.        planeObject = Instantiate planePrefab at this.transform.position
with this.transform.rotation
04.       slicingPlane = Get component PlaneSlice_EzySlice from planeObject
05.
06.         Set planeObject.transform as child of target.transform
07.         Set slicingPlane.target to target
08.
09.         If spawnedPlanes.Count is 0:
10.             // First plane
11.             Set slicingPlane.firstPlane to planeObject.transform
12.         Else If spawnedPlanes.Count is 1:
13.             // Second plane
14.              Rotate planeObject.transform by 180 degrees around the y-
axis
15.             Set slicingPlane.firstPlane to slicingPlanes[0].firstPlane
16.             Set slicingPlane.secondPlane to planeObject.transform
17.             Set slicingPlanes[0].secondPlane to planeObject.transform
18.         End If
19.
20.         Add planeObject to spawnedPlanes
21.         Add slicingPlane to slicingPlanes
22.     End If
23. End Method
```

Kode Sumber 4.9 *Pseudocode method* untuk memunculkan *slicing plane* pada interaksi ketiga

Pada interaksi ketiga, jika *flag boolean* `hasCollided` adalah *true*, maka munculkan *slicing plane* dimana *ghost plane* berada. Sama seperti pada interaksi sebelumnya, semua *slicing plane* yang muncul akan mendapatkan komponen dari *script* `PlaneSlice_EzySlice.cs`. Kemudian semua *slicing plane* yang muncul dijadikan *child* dari target dan dimasukkan ke dalam sebuah *list*. Jika *slicing plane* pertama muncul, *assign* nilai *transform* dari objek tersebut ke dalam *field* First Plane di *script* `PlaneSlice_EzySlice.cs`, sementara nilai *transform* dari *slicing plane* kedua akan di-*assign* ke dalam *field* Second Plane.

Untuk memunculkan *slicing plane* setelah pengguna menyisipkan *ghost plane* ke dalam

mandibula, pengguna dapat melakukan pose jempol ke atas seperti pada Gambar 4.28.



Gambar 4.28 Pose untuk memunculkan *slicing plane* melalui *ghost plane*

4.7 Implementasi Slicing

Slicing adalah interaksi yang ingin dicapai dari aplikasi XR bedah rahang. Untuk dapat melakukan proses *slicing*, dibutuhkan dua buah *slicing plane* yang menghadap berlawanan. Bagian yang di *slice* adalah bagian yang terdapat di dala, atau di belakang dua *plane*. Untuk dapat melakukan *slicing*, semua *slicing plane* menggunakan komponen *script* `PlaneSlice_EzySlice.cs`. Komponen *script* `PlaneSlice_EzySlice.cs` pada *slicing plane* merupakan *script* yang menjadikan *plane* dapat memotong target menggunakan nilai *transform* dari GameObject *plane*, yang didapat setelah *slicing plane* dimunculkan pada *script* `SpawnPlane.cs` dan versi lainnya. *Script* ini menggunakan *state* untuk mengatur proses *slice* dan *revert slice* untuk dapat memotong serta mengurungkan hasil potongan. Berikut adalah kode dari *script* `PlaneSlice_EzySlice.cs` yang terdapat pada Kode Sumber 4.10.

PlaneSlice_EzySlice.cs

```
01. Method Start():
02.     skullParent = Find game object with tag "Skull"
03. End Method
04.
05. Method Slice(target: GameObject):
07.     Detach target from parent
08.
09.    firstSlice = Slice target at firstPlane.position along firstPlane.up
10.
11.     If firstSlice is not null:
12.         upperHull = Create upper hull from firstSlice using target and
crossSectionMaterial
13.         lowerHull = Create lower hull from firstSlice using target and
crossSectionMaterial
14.         Set upperHull's parent to skullParent.transform
15.
16.           secondSlice = Slice lowerHull at secondPlane.position along
secondPlane.up
17.
18.         If secondSlice is not null:
19.             secondUpperHull = Create upper hull from secondSlice using
target and crossSectionMaterial
20.           middleHull = Create lower hull from secondSlice using target
```

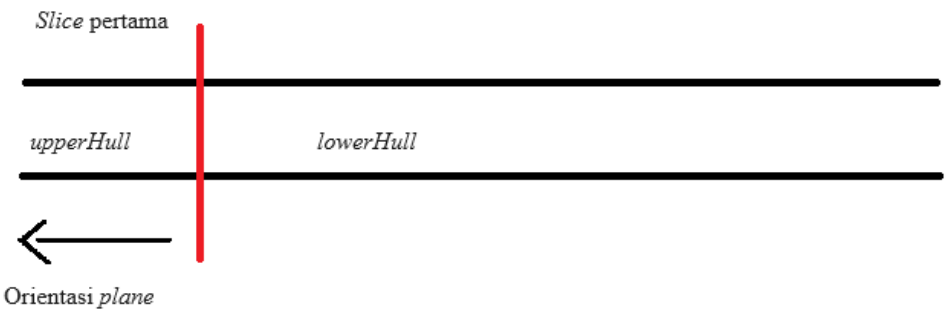
```
and crossSectionMaterial
21.             Set secondUpperHull's parent to skullParent.transform
22.
23.             Add upperHull to slicedParts
24.             Add secondUpperHull to slicedParts
25.
26.             Destroy lowerHull
27.             Destroy middleHull
28.         End If
29.
30.         Set target to inactive
31.         Set currentState to SliceState.Sliced
32.     End If
33. End Method
```

Kode Sumber 4.10 *Psuedocode method* untuk melakukan *slice*

Setelah *slicing plane* dimunculkan, kondisi *state* masih berada pada *Original*. Kemudian pada method `Start()`, *script* akan mencari apakah ada GameObject dengan *tag* "Skull" di dalam *scene*. Pada method `Slice()`, yang memiliki argumen target bertipe GameObject, target yaitu Skull_Mandible akan dilepas dari parent-nya.

Cara kerja dari *package* EzySlice yang digunakan pada *script* ini adalah melakukan proses *slice*, kemudian dari hasil *slice* tersebut terbentuk *hull* atas dan *hull* bawah. *Hull* atas terbentuk di depan orientasi *plane* dan *hull* bawah terbentuk di belakang *plane*. Dalam *script* `PlaneSlice_EzySlice.cs`, jika kedua *plane* telah muncul dan nilai transform dari kedua *plane* telah diisi pada *field* First Plane dan Second Plane, maka lakukan *slice* pertama. Hasil *slice* tersebut adalah dua GameObject baru yaitu `upperHull` dan `lowerHull`, di mana `upperHull` dijadikan *child* dari *target parent*. Setelah *slice* pertama, lanjutkan ke proses *slice* kedua. Target *slice* kedua adalah `lowerHull` yang dihasilkan dari proses *slice* pertama. Hasil dari *slice* kedua adalah `upperHull` dan `lowerHull` lagi yang diberi nama `secondUpperHull` dan `middleHull`. Sama seperti sebelumnya, *secondUpperHull* dijadikan *child* dari target parent.

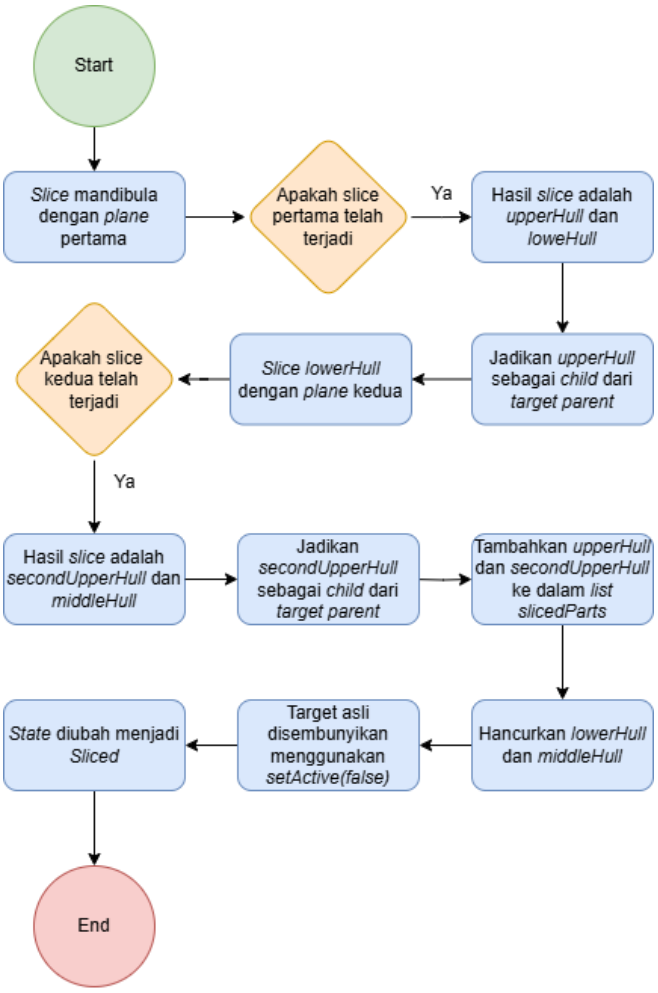
Setelah proses *slice* selesai secara keseluruhan, semua `upperHull` akan dimasukkan ke dalam sebuah list sedangkan semua `lowerHull` akan dihancurkan. Kemudian target asli akan disembunyikan dengan fungsi `setActive(false)` dan *state* akan diganti ke *Sliced*. Untuk memudahkan pemahaman alur *slice*, Gambar 4.29, Gambar 4.30, dan Gambar 4.31 adalah contoh visualisasi proses slice, dan Gambar 4.32 adalah alur bagaimana proses slice terjadi.



Gambar 4.29 Proses *slice* pertama

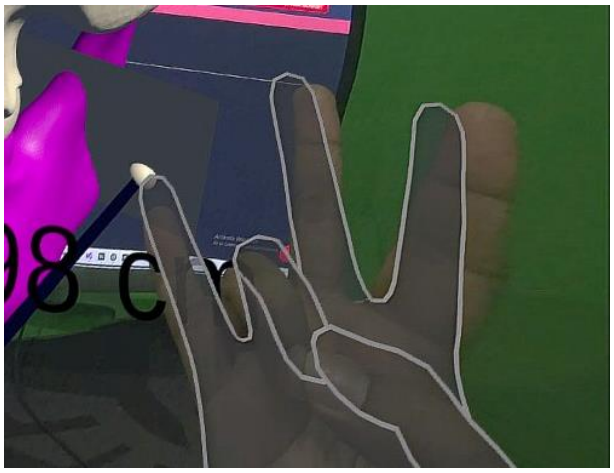
Gambar 4.30 Proses *slice* kedua

Gambar 4.31 Hasil dari proses *slice*

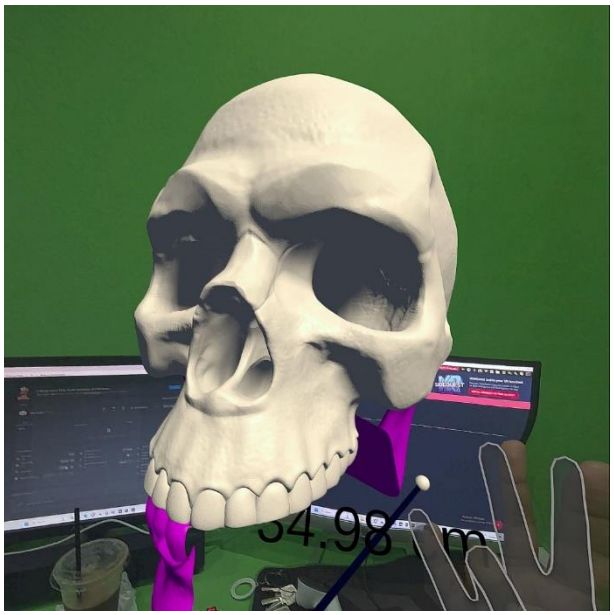


Gambar 4.32 Diagram alur proses *slice*

Untuk dapat melakukan slicing, pengguna melakukan *hand pose* seperti pada Gambar 4.33, dan hasil dari mandibula yang telah dilakukan *slicing* dapat dilihat pada Gambar 4.34**.**



Gambar 4.33 Pose untuk melakukan *slice*



Gambar 4.34 Hasil dari proses *slice* pada mandibula

4.8 Implementasi Undo

Interaksi lain yang ada di dalam aplikasi XR bedah rahang adalah interaksi untuk melakukan *undo* atau mengurungkan. Interaksi ini terbagi menjadi dua yaitu mengurungkan *slicing plane* yang telah muncul dan mengurungkan hasil dari *slice*. Ketika pengguna merasa melakukan kesalahan atau tidak puas baik dari penempatan *plane* atau hasi *slice*, pengguna dapat mengurungkan hasil tersebut. Proses *undo* dibagi menjadi dua yaitu *undo* untuk *slicing plane* dan hasil *slice*.

4.8.1 Implementasi Undo Plane

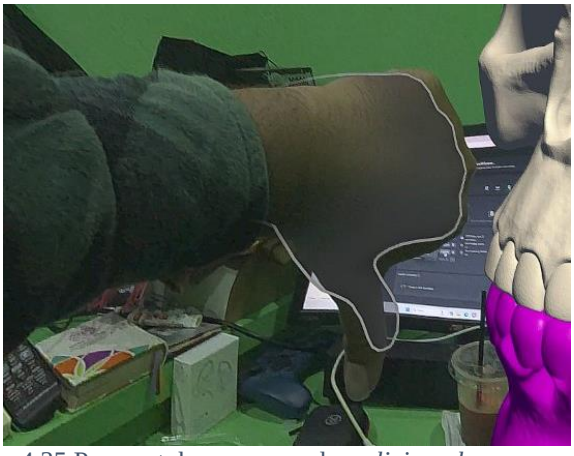
Undo dalam *slicing plane* digunakan ketika pengguna merasa salah atau kurang puas dalam memunculkan *slicing plane* di lokasi tertentu. Untuk dapat mengimplementasi proses mengurungkan *plane* tersebut, maka pada semua *script* `SpawnPlane.cs`, `SpawnPlane_V2.cs`, dan `SpawnPlane_V3.cs` memiliki sebuah *method* bernama `UndoSpawnPlane()`. Kode Sumber 4.11 merupakan *method* tersebut.

SpawnPlane.cs, SpawnPlane_V2.cs, SpawnPlane_V3.cs

```
01. Method UndoSpawnPlane():
02.     If spawnedPlanes.Count > 0:
03.         lastPlane = spawnedPlanes[spawnedPlanes.Count - 1]
04.         lastSlicingPlane = slicingPlanes[slicingPlanes.Count - 1]

05.         Destroy lastPlane

06.         Remove last element from spawnedPlanes
07.         Remove last element from slicingPlanes

08.         If slicingPlanes.Count > 0:
09.                 // Update the secondPlane reference of the remaining
PlaneSlice_EzySlice if needed
10.             slicingPlanes[0].secondPlane = (slicingPlanes.Count > 1) ?
slicingPlanes[1].transform : null
11.         Else:
12.             // Reset PlaneSlice_EzySlice references when no planes are
left
13.             For each plane in slicingPlanes:
14.                 plane.firstPlane = null
15.                 plane.secondPlane = null
16.             End For
17.         End If
18.     End If
19. End Method
```

Kode Sumber 4.11 *Pseudocode method* untuk melakukan *undo* pada *slicing plane*

Method tersebut mengecek apakah ada *slicing plane* yang dimunculkan. Jika lebih dari nol, maka ketika pengguna melakukan *undo*, *plane* terakhir di dalam *list* objek *plane* dan *list* nilai *transform* pada *script* `PlaneSlice_EzySlice.cs` akan dikurangi satu. Kemudian *slicing plane* yang ada di *scene* akan dihancurkan dan elemen-elemen yang dipasang di *field script* terkait akan dihapus. Kemudian *script* akan mengecek lagi apakah *list slicing plane* lebih dari nol. Jika iya, maka referensi di *script* `PlaneSlice_EzySlice.cs` akan dihapus satu dari yang terakhir terpasang. Jika tidak ada *plane* yang ada di *scene*, maka *reset* semua nilai dan referensi pada *script* terkait sehingga semua *field* dan *script* yang bersangkutan akan bernilai nol. Untuk dapat mengurungkan *slicing plane* yang telah muncul, pengguna dapat melakuka *hand pose* jari telunjuk ke bawah seperti pada Gambar 4.35.



Gambar 4.35 Pose untuk mengurungkan *slicing plane* yang muncul

4.8.2 Implementasi Undo Slice

Interaksi *undo* selanjutnya adalah *undo slice* yang digunakan untuk mengembalikan mandibula

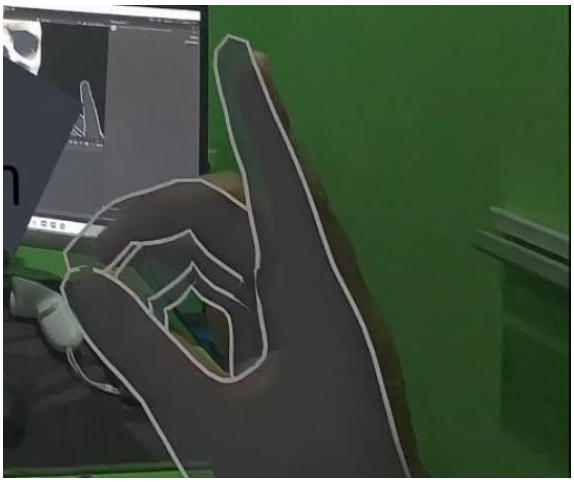
seperti semula setelah proses *slice* ketika pengguna merasa kurang puas dengan hasil *slice*. Ketika mandibula telah dalam kondisi *Sliced*, maka *script* `PlaneSlice_EzySlice.cs` dapat memanggil *method* `RevertSlice()` yang dapat dilihat pada Kode Sumber 4.12**.**

PlaneSlice_EzySlice.cs

```
01. Method RevertSlice():
02.     If currentState is SliceState.Sliced:
03.         For each part in slicedParts:
04.             Destroy part
05.         End For
06.         Clear slicedParts list
07.
08.         Set target to active
09.
10.       // Synchronize transform properties with the parent (skullParent)
11.        Set target.transform.position to skullParent.transform.position
12.        Set target.transform.rotation to skullParent.transform.rotation
13.                                  Set   target.transform.localScale   to
skullParent.transform.localScale
14.         Set target.transform as child of skullParent.transform
15.
16.         Set currentState to SliceState.Original
17.     End If
18. End Method
```

Kode Sumber 4.12 *Pseudocode method* untuk mengurungkan hasil *slice*

Ketika *state* mandibula saat ini adalah *Sliced*, maka setiap *part* yang ada di *list*, yaitu semua `upperHull` yang telah dimasukkan ke dalam *list* akan dihancurkan. Setelah dihancurkan, maka *list parts* akan dihapus. Kemudian, target asli yang disembunyikan setelah *slicing* akan diaktifkan kembali menggunakan `setActive(true)`. Kemudian, *slicing plane* yang dihancurkan akan dikembalikan lagi dan diisi referensi-referensi pada *script* terkait. Terakhir adalah mengembalikan *state* kembali ke *Original* agar mandibula dapat dilakukan *slice* kembali. Untuk dapat melakukan *undo slice*, pengguna melakukan pose seperti pada Gambar 4.36.

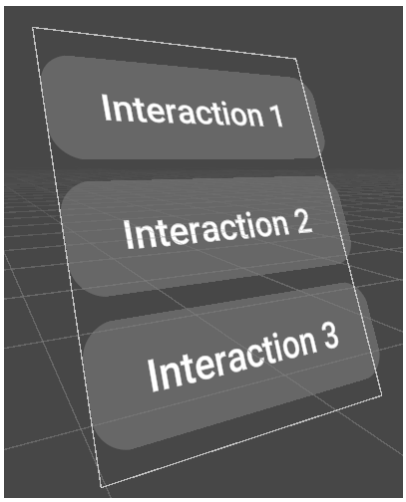


Gambar 4.36 Pose untuk mengurungkan hasil *slice*

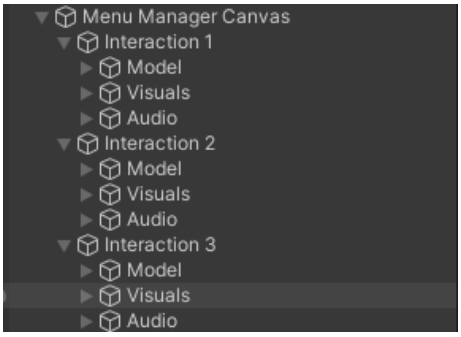
4.9 Implementasi Antarmuka

UI atau antarmuka pada aplikasi aplikasi XR bedah rahang digunakan untuk memindah *scene* dan memainkan video *guide* tahap-tahap melakukan interaksi sampai ke proses *slicing* dan *undo* slicing. Tombol-tombol yang ada di antarmuka pada aplikasi ini menggunakan *interactable* dari

Oculus Interaction SDK yaitu *poke interactable*. Salah satu contoh penggunaan *poke* adalah pada tombol pada UI *scene menu*. Gambar 4.37 adalah tampilan dari UI *scene menu* sedangkan Gambar 4.38 adalah *hierarchy*-nya.

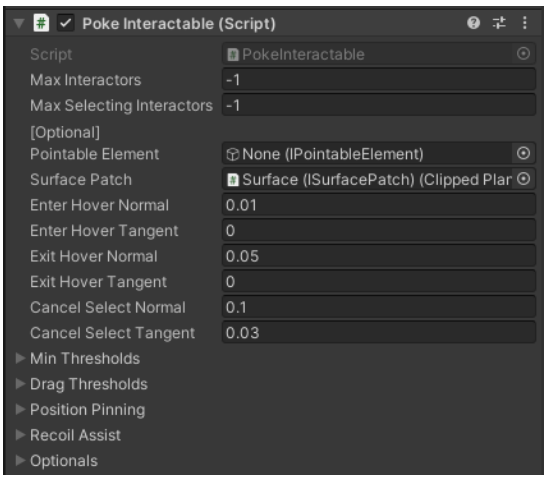


Gambar 4.37 Tampilan UI *scene menu*



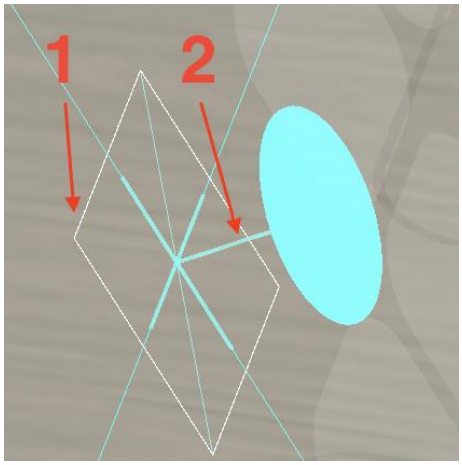
Gambar 4.38 *Hierarchy* dari UI *scene menu*

Tiap tombol *interaction*, terdapat tiga *child* yaitu *model, visuals*, dan *audio*. *Visual* digunakan hanya untuk menampilkan bentuk tombol dan juga teks, sedangkan *audio* digunakan untuk memainkan suara ketika tombol ditekan dan dilepas. Namun, *model* adalah komponen utama dimana ada sebuah *surface* yang membuat tombol dapat oleh tangan pengguna. Di tiap tombol *interaction*, terdapat komponen *script* PokeInteractable yang dipasang. Gambar 4.39 adalah tampilan dari komponen tersebut.



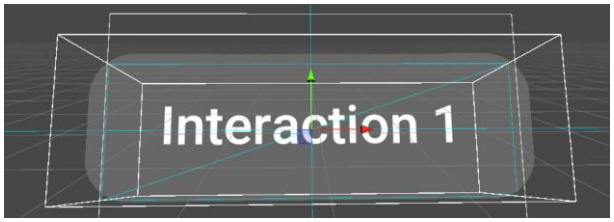
Gambar 4.39 Komponen *script poke interactable* pada tombol *antarmuka*

Pada komponen tersebut, dapat diatur bagaimana kondisi pengguna dapat menyentuh tombol seperti sejauh apa *surface* pada tombol hingga beberapa pengaturan lain. Untuk menentukan kapan tangan pengguna memasukkan *hover*, maka menggunakan MinThresholdsConfig. Menggunakan RecoilAssistConfig dapat mengatur aturan pembatalan dan pemilihan ulang. Menggunakan DragThresholdsConfig untuk membedakan antara menyeret dan menekan, dan untuk menekan *event* penunjuk bergerak ketika *poke interactor* mengikuti gerakan menekan, pengguna dapat *poke* permukaan yang dapat disodok. Dengan menggunakan PositionPinningConfig, pengguna dapat menciptakan rasa gesekan selama menyeret. Sebagai contoh, Gambar 4.40 adalah contoh *surface* dan *arah* dimana jari dapat melakukan *poke*. Nomor 1 adalah permukaan sebuah tombol sedangkan nomor dua adalah nilai normal-nya.

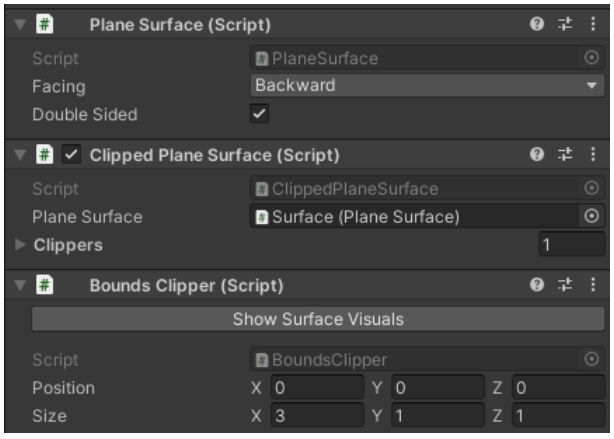


Gambar 4.40 Tampilan permukaan dan arah normal untuk *poke*

Sebagai contoh, pada tombol interaksi pertama, Gambar 4.41 adalah tampilan *surface* dari tombol sedangkan Gambar 4.42 adalah komponen *script*-nya.

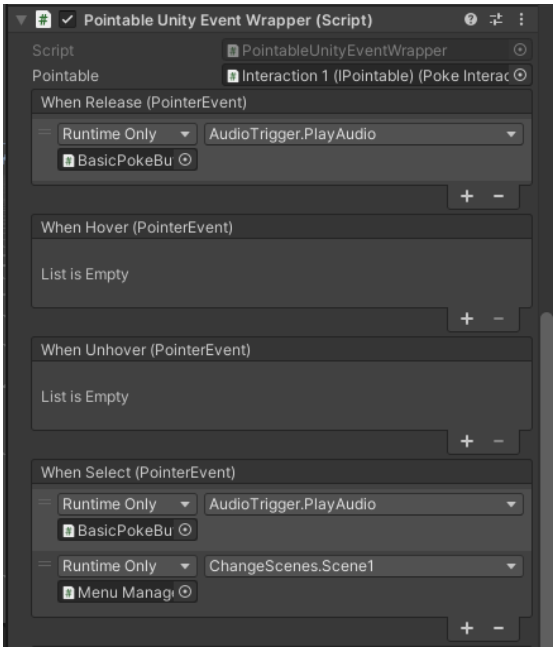


Gambar 4.41 Tampilan *surface* pada tombol *interaction* 1



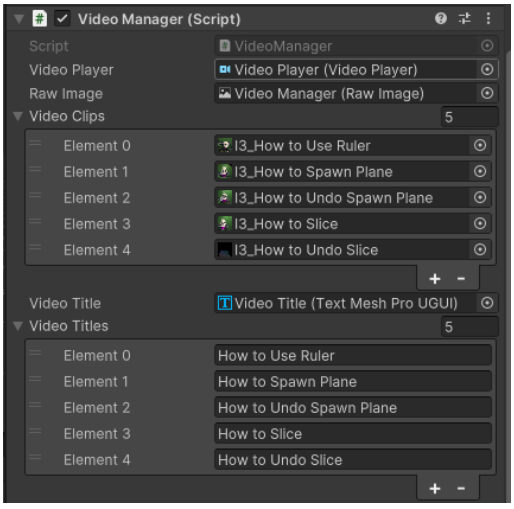
Gambar 4.42 Tampilan komponen *surface*

Lalu kemudian ada komponen *event wrapper* seperti pada Gambar 4.43**.** Komponen tersebut digunakan untuk menangani *events* yang akan dieksekusi ketika pengguna menekan tombol. Pada komponen tersebut, sebagai contoh, ketika tombol ditekan, maka dalam *field* `When Select()`, *event* akan memainkan *audio* dan mengganti *scene*.



Gambar 4.43 Tampilan *event wrapper*

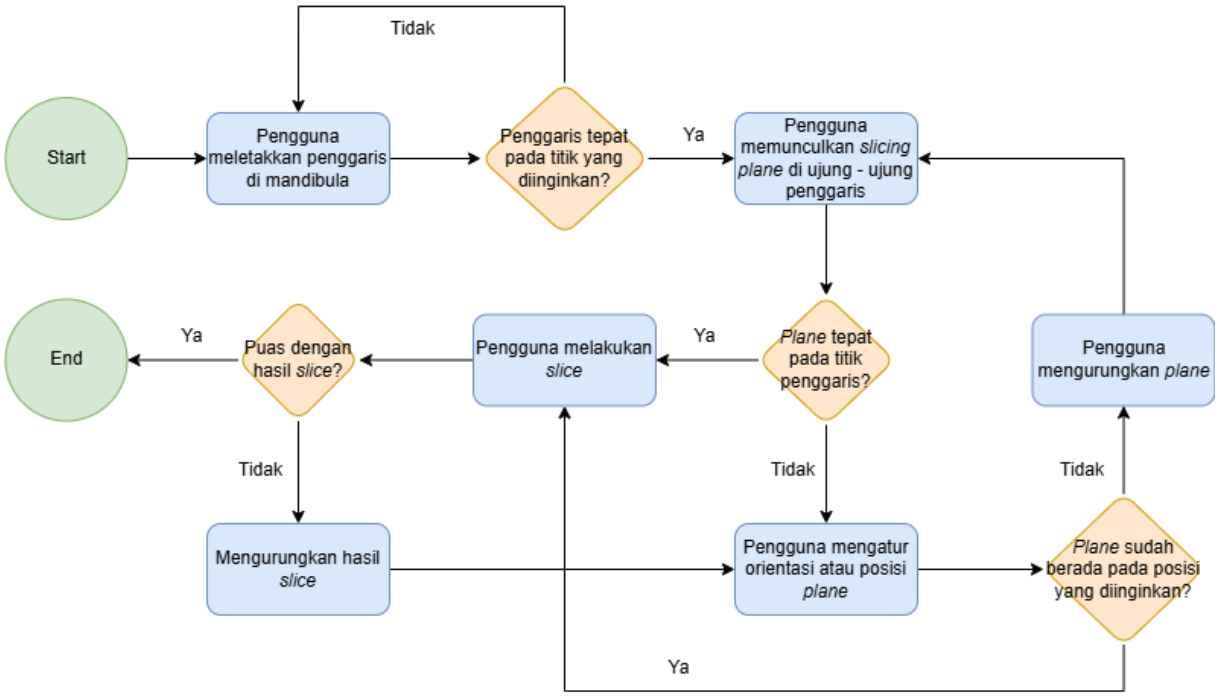
Untuk antarmuka *video guide*, masih ada komponen tombol di dalamnya. Namun, di dalam *canvas* terdapat GameObject VideoPlayer dan RawImage. RawImage digunakan sebagai *frame* dan manajer untuk video, sedangkan VideoPlayer bertugas untuk memainkan video. Panel atau *canvas* video juga dapat dipindah karena memiliki komponen Grabbable di dalamnya. Di dalam manajer video, terdapat *script* yang digunakan untuk memindah video dari video satu ke video selanjutnya atau sebelumnya. Hal tersebut dapat tercapai karena *script* membuat semacam *list* dari semua video. Gambar 4.44 adalah tampilan komponen *script* VideoManager. Ketika pengguna menekan tombol *next* atau *previous*, maka tombol tersebut memanggil `WhenSelect()` pada *event wrapper* dan mengganti video.



Gambar 4.44 Tampilan komponen *script* VideoManager

4.10Implementasi Interaksi

Jika di dalam sebuah aplikasi permainan terdapat level, maka di aplikasi XR bedah rahang terdapat interaksi. Interaksi di dalam aplikasi adalah level-level yang akan dimainkan atau dipakai pengguna. Ada tiga variasi interaksi dimana interaksi pertama dan kedua menggunakan *pointer* dan *point* untuk memunculkan *slicing plane* sementara interaksi ketiga menggunakan *ghost plane*. Misi atau objektif dari semua interaksi adalah sama, yaitu untuk memotong mandibula. Untuk dapat lebih memahami alur interaksi, dapat dilihat pada Gambar 4.45, namun dengan perbedaan pada bagaimana cara memunculkan *slicing plane* pada masing-masing interaksi.



Gambar 4.45 Diagram alur interaksi yang dilakukan

4.10.1 Interaksi Pertama

Interaksi pertama menggunakan dua buah *point* untuk memunculkan *slicing plane* di antara kedua *point*. *Slicing plane* yang digunakan adalah *normal slicing plane* yang dapat dipindah dan dirotasi bebas kemana saja sesuai keinginan pengguna. Seperti yang digambarkan pada diagram alur yang pada Gambar 4.45, pengguna meletakkan penggaris dengan memegang titik-titik ujung penggaris ke mandibula. Kemudian, setelah penggaris telah berada di tempat yang sesuai, maka kemudian pengguna mengaktifkan *pointer* terlebih dahulu di jari telunjuk mereka. Pose untuk mengaktifkan *pointer* dapat dilihat pada Gambar 4.24. Kemudian ketika *pointer* muncul, maka pengguna dapat menyentuh mandibula dengan jari telunjuk mereka untuk memunculkan *points* di titik yang sudah ditentukan penggaris. Tampilan ketika *point* telah muncul di mandibula dapat dilihat pada Gambar 4.25. Ketika dua buah *point* telah muncul, maka *slicing plane* akan muncul di antara kedua *point* dan *point* akan dihapus. Kemudian ulangi lagi tahapan yang sama untuk *slicing plane* kedua. Jika pengguna merasa puas atau merasa *slicing plane* sudah pada posisi yang tepat, maka pengguna bisa melanjutkan ke pose *slice* untuk memotong mandibula. Pose *slice* dapat dilihat pada Gambar 4.33. Namun jika pengguna merasa posisi *slicing plane* kurang tepat, pengguna dapat memegang objek tersebut dan memindahnya, atau pengguna juga dapat mengurungkan *slicing plane* yang dimunculkan ketika pengguna kelebihan mengeluarkan *slicing plane* atau melakukan kesalahan dan tidak ingin memindahnya. Pose untuk mengurungkan *slicing plane* terdapat pada Gambar 4.35. Terakhir, ketika pengguna

telah melakukan *slice* dan merasa salah memotong atau kurang puas dengan hasil potongan, maka pengguna dapat juga mengurungkan hasil potongan dengan melakukan pose seperti pada Gambar 4.36, dan mandibula akan kembali seperti semula dengan *slicing planes* yang muncul lagi sesuai dengan keadaan sebelumnya.

4.10.2 Interaksi Kedua

Interaksi kedua kurang lebih mirip dengan interaksi pertama, yaitu pengguna memunculkan *point* pada mandibula dengan *pointer* di jari telunjuk mereka. Namun yang membedakan adalah jumlah *point* yang dimunculkan dan jenis *slicing plane* yang digunakan. Pada interaksi kedua, alur yang dikerjakan sama seperti sebelumnya, namun pengguna hanya memerlukan satu *point* untuk memunculkan *slicing plane*. *Slicing plane* tersebut akan muncul tepat dimana *point* berada. Kemudian, *slicing plane* yang digunakan pada interaksi kedua adalah *fixed slicing plane*, dimana objek tersebut tidak bisa dipindahkan namun hanya bisa dirotasi. Jadi ketika pengguna salah memunculkan *slicing plane,* maka pengguna harus mengurungkan *slicing plane* menggunakan pose pada Gambar 4.35. Untuk proses *slice* dan *undo slice* juga sama seperti sebelumnya.

4.10.3 Interaksi Ketiga

Interaksi ketiga adalah interaksi yang paling berbeda dari kedua interaksi sebelumnya. Pada interaksi ini, pengguna tidak menggunakan *pointer* untuk memunculkan *slicing plane* melainkan menggunakan *ghost plane* di jari telunjuk mereka. Pengguna meletakkan penggaris pada mandibula. Kemudian, pengguna melakukan pose seperti pada Gambar 4.27. Ketika *ghost plane* telah muncul di jari telunjuk pengguna, maka pengguna akan seperti menyisipkan *ghost plane* tersebut ke dalam mandibula. Ketika *ghost plane* telah disisipkan, maka pengguna melakukan pose seperti pada Gambar 4.28 untuk memunculkan *slicing plane* di posisi dan rotasi yang sama dengan *ghost plane* yang disisipkan. *Slicing plane* yang digunakan pada interaksi ketiga adalah *normal slicing plane* karena pengguna juga dapat memindah *plane* ke posisi yang mereka inginkan. Selanjutnya, pengguna melakukan *slice* jika dirasa sudah tepat dalam memunculkan *slicing plane* atau dapat memindah atau mengurungkannya jika merasa kurang tepat. Terakhir, pengguna juga bisa melakukan *undo* slice seperti sebelumnya.

BAB 5
HASIL DAN PEMBAHASAN

5.1 Hasil Pengujian

Pengujian menggunakan metode Shapiro-Wilk untuk menentukan apakah data adalah distribusi normal atau bukan. Kemudian, data diujikan menggunakan metode uji Kruskal Wallis dan Wilcoxon Signed-Rank untuk mengevaluasi performa partisipan dalam menyelesaikan interaksi dan untuk membandingkan interaksi mana yang paling mudah diselesaikan oleh partisipan agar dapat menemukan mana interaksi yang paling mudah dikerjakan. Pengujian juga menggunakan metode uji System Usability Scale (SUS) untuk mendapatkan skor *usability* dari pengguna untuk mendapatkan hasil apakah aplikasi dapat diterima atau tidak dapat diterima oleh pengguna. Pengujian Shapiro-Wilk digunakan untuk menentukan set data adalah distribusi normal atau bukan. Pengujian Kruskal Wallis digunakan untuk membandingkan performa partisipan dalam menyelesaikan *tasks* yang diberikan di seluruh tiga interaksi dan juga membandingkan total performa antara grup dan untuk menemukan apakah ada perbedaan yang signifikan di antara ketiga interaksi. Pengujian Wilcoxon Signed-Rank digunakan untuk membandingkan pasangan-pasangan interaksi apakah ada perbedaan signifikan antar pasangan interaksi tersebut. Sedangkan pengujian SUS digunakan untuk mendapatkan fakta untuk *usability* apakah aplikasi dapat diterima oleh pengguna atau tidak.

Pengujian aplikasi XR bedah rahang diberikan kepada enam mahasiswa yang belum pernah atau belum familiar dalam menggunakan perangkat atau aplikasi VR atau XR. Pengujian diberikan menggunakan metode yang dijelaskan pada subbab 3.5 Pengujian. Jumlah ini memenuhi perkiraan awal pengujian. Tabel 5.1 menampilkan detail partisipasi serta kode partisipasi.

Tabel 5.1 Kode dan Nama Partisipan

Kode Partisipan	Nama Partisipan
P1	Anak Agung Yatesha Parwata
P2	Heru Dwi Kurniawan
P3	Pierra Muhammad Shobr
P4	Cholid Junoto
P5	Hilmi Zharfan Rachmadi
P6	Andhika Ditya Bagaskara D

Partisipan akan dibagi menjadi dua grup yaitu Grup A dan Grup B dengan masing-masing grup berisikan tiga anggota. Grup A akan mengerjakan *task* pada urutan interaksi pertama, interaksi kedua, dan interaksi ketiga sedangkan Grup B akan mengerjakan urutan sebaliknya. Pembagian peserta pada tiap grup dapat dilihat pada Tabel 5.2**.**

Tabel 5.2 Grup dan Anggotanya

Grup A	Nama Partisipan	Grup B	Nama Partisipan
	Anak Agung Yatesha Parwata		Cholid Junoto
	Heru Dwi Kurniawan		Hilmi Zharfan Rachmadi
	Pierra Muhammad Shobr		Andhika Ditya Bagaskara D

Setelah itu, penguji memberikan sejumlah *tasks* kepada para peserta untuk mengevaluasi performa kinerja mereka. Rincian *tasks* yang diberikan ditunjukkan dalam Tabel 5.3**.**

Tabel 5.3 Task dalam Uji Performa Pengguna

Kode Pengujian Performa	*Task* Pengujian	Deskripsi *Task* Pengujian
PT-1	Menggunakan Penggaris	Partisipan diminta untuk menggunakan penggaris dan menaruh dua ujung penggaris sesuai keinginan mereka.
PT-2	Memunculkan Slicing Plane	Partisipan diminta untuk memunculkan *slicing plane* pada titik-titik yang telah ditentukan oleh penggaris.
PT-3	Mengurungkan Slicing Plane	Partisipan diminta untuk mengurungkan minimal satu *slicing plane* jika dua buah *slicing plane* telah muncul atau mengurungkan *slicing plane* yang muncul tanpa disengaja.
PT-4	Melakukan Slice	Partisipan diminta untuk melakukan *slice* setelah dua *slicing plane* muncul dan lokasi atau titik *plane* sudah sesuai keinginan.
PT-5	Mengurungkan Slice	Partisipan diminta untuk mengurungkan hasil *slice* setelah melakukan *slicing*.

Dari semua *tasks* yang diberikan, pengguna akan dihitung jumlah durasi melakukan semua *tasks* secara keseluruhan pada tiap interaksi. Kemudian jumlah akhir keseluruhan waktu dari ketiga interaksi juga akan dihitung. Dari tabel sebelumnya, dapat dibuat kode pengujian durasi untuk tiap interaksi dan jumlah durasi keseluruhan dari tiga interaksi. Kode tersebut dapat dilihat pada Tabel 5.4.

Tabel 5.4 Kode Hasil Pengujian dan Deskripsi

Kode Hasil Pengujian	Deskripsi Hasil Pengujian
I-1	Durasi untuk menyelesaikan interaksi pertama
I-2	Durasi untuk menyelesaikan interaksi kedua
I-3	Durasi untuk menyelesaikan interaksi ketiga
SUM	Jumlah keseluruhan durasi dari ketiga interaksi

Setelah partisipan dimasukkan ke grup masing-masing, selanjutnya partisipan mulai melakukan uji coba dengan melakukan *tasks* yang sudah diberikan pada semua interaksi. Setelah partisipan menyelesaikan uji coba, maka partisipan akan diberikan kuisioner mengenai *usability* dari aplikasi yang berisi 10 pertanyaan dengan skala 1 sampai 5 dan juga tiga pertanyaan mengenai karakteristik pengguna. Detail lebih lanjut mengenai pertanyaan kuisioner dapat dilihat pada Tabel 5.5 dan Tabel 5.6.

Tabel 5.5 Pertanyaan Karakteristik Pengguna

Kode Kuisioner	Pertanyaan Karakteristik Pengguna
PK-1	Nama
PK-2	Umur

PK-3	Jenis Kelamin

Tabel 5.6 Pertanyaan System System Usabilty Scale

Kode Kuisioner	Pertanyaan System Usability Scale
PSUS-1	Saya pikir saya akan sering menggunakan aplikasi ini
PSUS-1	Saya menemukan aplikasi ini terlalu kompleks
PSUS-1	Saya merasa aplikasi ini mudah digunakan
PSUS-1	Saya pikir saya memerlukan bantuan tenaga teknis untuk dapat menggunakan aplikasi ini
PSUS-1	Saya merasa fungsi-fungsi dalam aplikasi ini terintegrasi dengan baik
PSUS-1	Saya merasa terdapat terlalu banyak inkonsistensi dalam aplikasi ini
PSUS-1	Saya membayangkan kebanyakan orang akan belajar menggunakan aplikasi ini dengan sangat cepat
PSUS-1	Saya menemukan aplikasi ini sangat rumit untuk digunakan
PSUS-1	Saya merasa sangat percaya diri saat menggunakan aplikasi ini
PSUS-1	Saya perlu mempelajari banyak hal sebelum bisa menggunakan sistem ini

Data-data hasil pengujian akan diproses setelah seluruh data didapatkan dari semua partisipan. Rincian dari durasi penyelesaian setiap interaksi untuk tiap partisipan dapat dilihat pada Tabel 5.7.

Tabel 5.7 Hasil Pengambilan Durasi Pengguna dalam Menyelesaikan Interaksi

Grup	Kode Partisipan	I-1	I-2	I-3	SUM
A	P1	1.53	1.41	1.16	4.50
	P2	4.40	4.27	3.18	12.25
	P3	3.45	2.09	1.58	7.52
B	P4	2.46	1.23	1.06	5.15
	P5	2.12	2.01	2.58	7.11
	P6	2.54	1.38	3.57	8.29

Dari Tabel 5.7, maka dapat diolah menggunakan uji Shapiro Wilk untuk menentukan apakah data termasuk distribusi normal, kemudian data diolah menggunakan uji Kruskal Willis untuk setiap hasil durasi penyelesaian pada tiap interaksi dan jumlah dari keseluruhan ketiga interaksi. Dari Tabel 5.7 juga dapat diolah menggunakan uji Wilcoxon Signed-Rank untuk pasangan interaksi pertama dan interaksi kedua, pasangan interaksi kedua dan interaksi ketiga, pasangan interaksi pertama dan interaksi ketiga, dan pasangan jumlah waktu pada Grup A dan Grup B. Interpretasi dari hasil tabel adalah sebagai berikut.

- Jika (*p-value* < 0,05), ada perbedaan yang signifikan di seluruh level.
- Jika (*p-value* ≥ 0,05), tidak ada perbedaan yang signifikan di seluruh level.

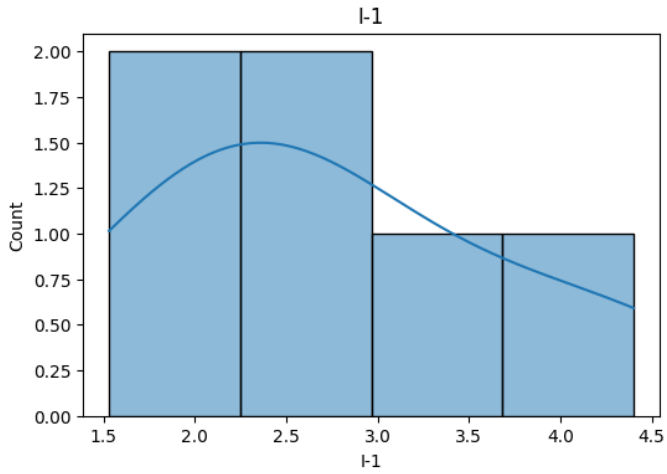
Hasil olah data menggunakan uji Shapiro-Wilk untuk setiap interaksi dan total waktu dapat dilihat pada Tabel 5.8.

Tabel 5.8 Hasil Uji Shapiro-Wilk untuk Seluruh Data

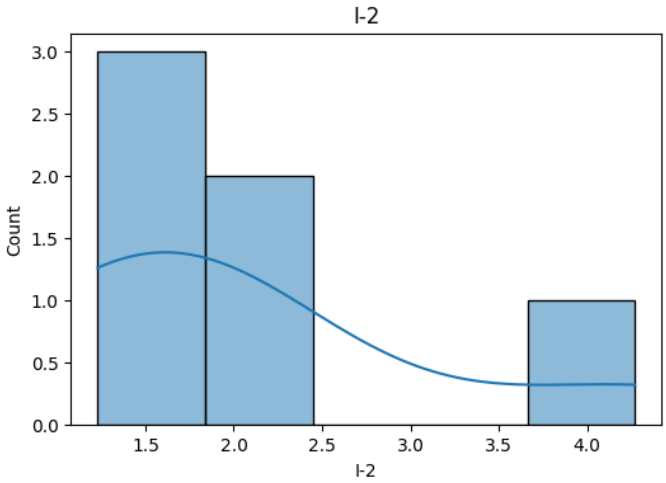
	I-1	I-2	I-3	SUM
P1	1.53	1.41	1.16	4.50
P2	4.40	4.27	3.18	12.25

P3	3.45	2.09	1.58	7.52
P4	2.46	1.23	1.06	5.15
P5	2.12	2.01	2.58	7.11
P6	2.54	1.38	3.57	8.29
Hasil				
Mean	2.75	2.06	2.18	7.47
Std	1.02	1.13	1.07	2.75
Min	1.53	1.23	1.06	4.50
25%	2.20	1.38	1.27	5.64
50%	2.50	1.71	2.08	7.31
75%	3.22	2.07	3.03	8.09
Max	4.40	4.27	3.57	12.25
Uji Statistik	0.95	0.75	0.89	0.92
p-value	0.72	0.02	0.35	0.50
Interpretasi			**I-1**	Tidak ada perbedaan yang signifikan di seluruh interaksi
			I-2	Ada perbedaan yang signifikan di seluruh interaksi
			I-3	Tidak ada perbedaan yang signifikan di seluruh interaksi
			SUM	Tidak ada perbedaan yang signifikan di seluruh interaksi

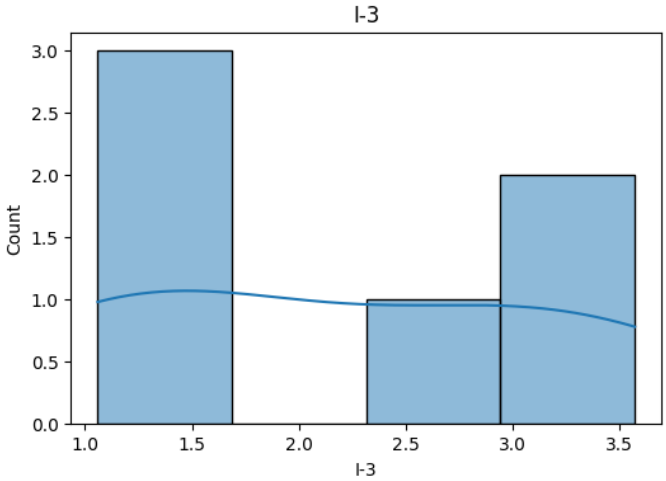
Dan untuk grafik hasil dari uji Shapiro-Wilk untuk interaksi pertama dapat dilihat pada Gambar 5.1, interaksi kedua pada Gambar 5.2, interaksi ketiga pada Gambar 5.3, dan total waktu pada Gambar 5.4.



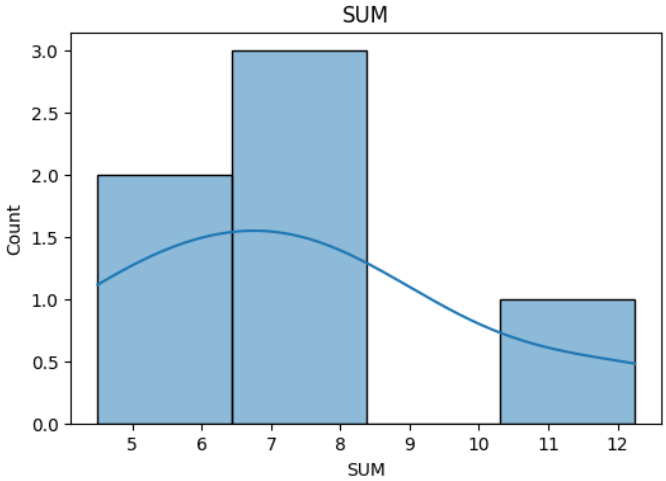
Gambar 5.1 Grafik uji Shapiro-Wilk untuk waktu interaksi pertama



Gambar 5.2 Grafik uji Shapiro-Wilk untuk waktu interaksi kedua



Gambar 5.3 Grafik uji Shapiro-Wilk untuk waktu interaksi ketiga



Gambar 5.4 Grafik uji Shapiro-Wilk untuk total waktu tiap partispan

Hasil olah data dengan uji Kruskal Wallis untuk hasil waktu penyelesaian tiap interaksi dapat dilihat pada Tabel 5.9. Sedangkan hasil waktu penyelesaian untuk jumlah durasi dari ketiga interaksi pada masing-masing grup dapat dilihat pada Tabel 5.10.

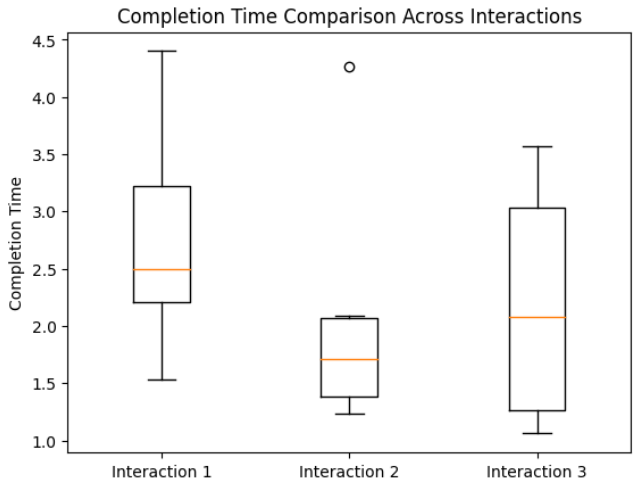
Tabel 5.9 Hasil Uji Kruskal Wallis untuk Waktu Penyelesaian pada Tiap Interaksi

	I-1	I-2	I-3
P1	1.53	1.41	1.16
P2	4.40	4.27	3.18
P3	3.45	2.09	1.58
P4	2.46	1.23	1.06
P5	2.12	2.01	2.58
P6	2.54	1.38	3.57
		Jumlah Sampel (Interaksi)	3
		Uji Statistik	2.12
		p-value	0.35
Interpretasi	Tidak ada perbedaan yang signifikan di seluruh interaksi		

Tabel 5.10 Hasil Uji Kruskal Wallis untuk Jumlah Durasi Penyelesaian dari Ketiga Interaksi pada Masing – masing Grup

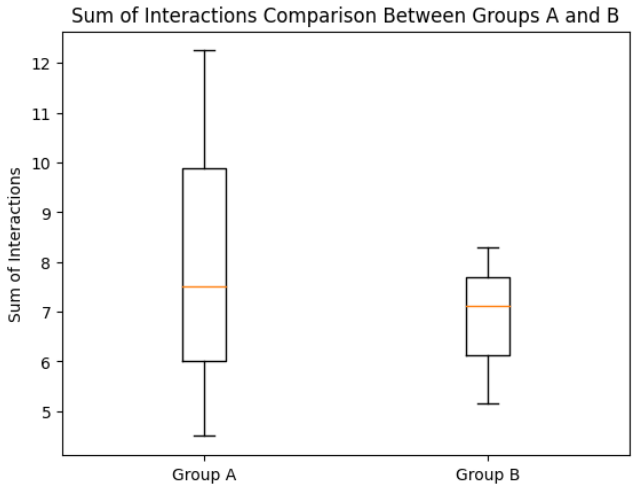
Group	Participants	SUM
Group A	P1	4.50
	P2	12.25
	P3	7.52
Group B	P4	5.15
	P5	7.11
	P6	8.29
	Jumlah Sampel (Group)	2
	Uji Statistik	0.05
	p-value	0.83
Interpretasi	Tidak ada perbedaan yang signifikan di seluruh interaksi	

Lalu untuk boxplot dari uji Kruskal Wallis untuk hasil waktu penyelesaian tiap interaksi dapat dilihat pada Gambar 5.5.



Gambar 5.5 Boxplot waktu penyelesaian pada tiap interaksi

Sedangkan boxplot dari hasil waktu penyelesaian untuk jumlah durasi dari ketiga interaksi pada masing-masing grup dapat dilihat pada Gambar 5.6.



Gambar 5.6 Boxplot waktu penyelesaian ketiga interaksi pada masing - masing grup

Sementara untuk hasil uji Wilcoxon Signed-Rank untuk hasil waktu penyelesaian pasangan interaksi pertama dan interaksi kedua dapat dilihat pada Tabel 5.11**,** untuk pasangan interaksi kedua dan interaksi ketiga dapat dilihat pada Tabel 5.12, untuk pasangan interaksi pertama dan interaksi ketiga dapat dilihat pada Tabel 5.13, dan untuk pasangan Grup A dan Grup B dapat dilihat pada Tabel 5.14.

Tabel 5.11 Hasil Uji Wilcoxon Signed-Rank pada Pasangan Interaksi Pertama dan Interaksi Kedua

	I-1	I-2	Selisih	Selisih Mutlak	Perinkat Positif	Peringkat Negatif
P1	1.53	1.41	0.12	0.12	2	0
P2	4.40	4.27	0.13	0.13	3	0
P3	3.45	2.09	1.36	1.36	6	0
P4	2.46	1.23	1.23	1.23	5	0
P5	2.12	2.01	0.11	0.11	1	0
P6	2.54	1.38	1.16	1.16	4	0
				Total	21	0
				Jumlah Sample (P-n)		6
				Uji Statistik		0.0
				p-value		0.03
	Interpretasi	Ada perbedaan yang signifikan di kedua interaksi				

Tabel 5.12 Hasil Uji Wilcoxon Signed-Rank pada Pasangan Interaksi Kedua dan Interaksi Ketiga

	I-2	I-3	Selisih	Selisih Mutlak	Perinkat Positif	Peringkat Negatif
P1	1.41	1.16	0.25	0.25	2	0
P2	4.27	3.18	1.09	1.09	5	0
P3	2.09	1.58	0.51	0.51	3	0
P4	1.23	1.06	0.17	0.17	1	0
P5	2.01	2.58	-0.57	0.57	0	4
P6	1.38	3.57	-2.19	2.19	0	6
				Total	11	10
				Jumlah Sample (P-n)		6
				Uji Statistik		10.0
				p-value		1.0

Interpretasi	Tidak ada perbedaan yang signifikan di kedua interaksi

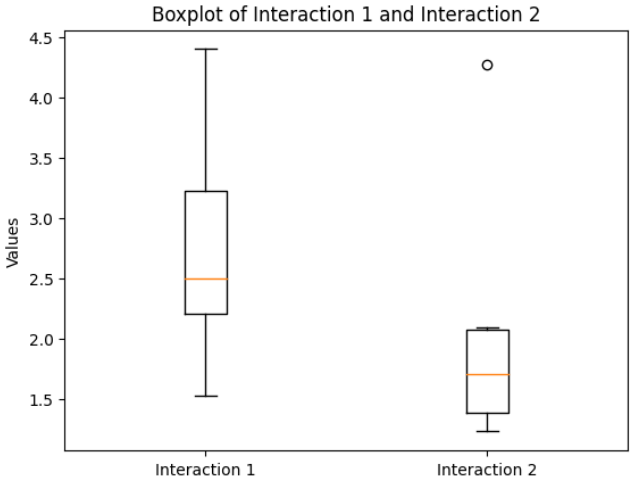
Tabel 5.13 Hasil Uji Wilcoxon Signed-Rank pada Pasangan Interaksi Pertama dan Interaksi Ketiga

	I-1	I-3	Selisih	Selisih Mutlak	Perinkat Positif	Peringkat Negatif
P1	1.53	1.16	0.37	0.37	1	0
P2	4.40	3.18	1.22	1.22	4	0
P3	3.45	1.58	1.87	1.87	6	0
P4	2.46	1.06	1.4	1.4	5	0
P5	2.12	2.58	-0.46	0.46	0	2
P6	2.54	3.57	-1.03	1.03	0	3
				Total	16	5
				Jumlah Sample (P-n)		6
				Uji Statistik		5.0
				p-value		0.31
	Interpretasi	Tidak ada perbedaan yang signifikan di kedua interaksi				

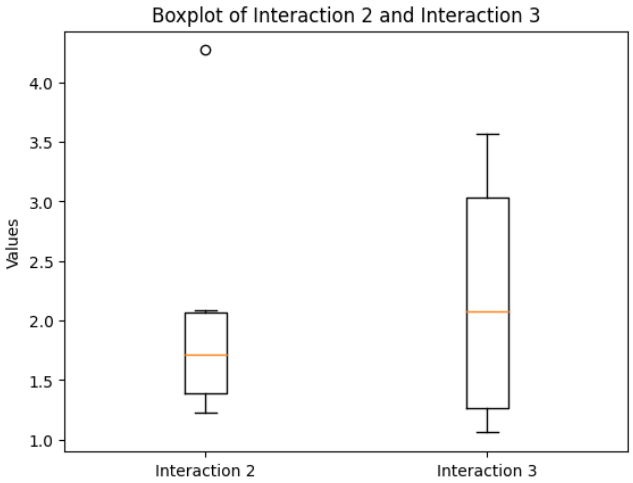
Tabel 5.14 Hasil Uji Wilcoxon Signed-Rank pada Pasangan Grup A dan Grup B

	Grup A	Grup B	Selisih	Selisih Mutlak	Perinkat Positif	Peringkat Negatif
I-1	4.50	1.06	3.44	3.44	1	0
I-2	12.25	2.58	9.67	9.67	3	0
I-3	7.52	3.57	3.95	3.95	2	0
				Total	6	0
				Jumlah Sample		3
				Uji Statistik		0.0
				p-value		0.25
	Interpretasi	Tidak ada perbedaan yang signifikan di kedua interaksi				

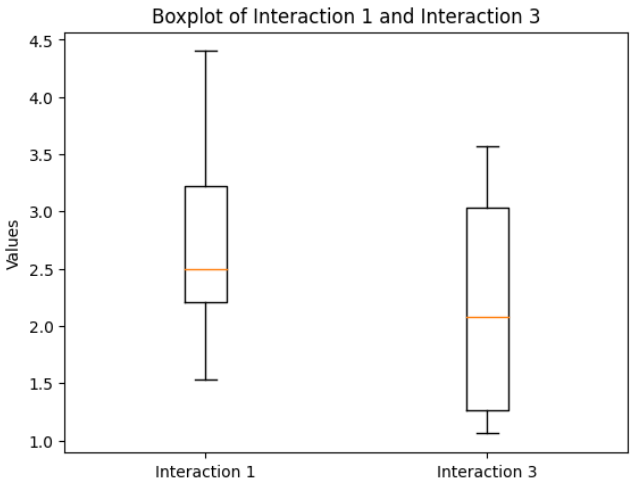
Dan untuk gambar boxplot untuk dari urutan pasangan dapat dilihat pada Gambar 5.7, Gambar 5.8, Gambar 5.9, dan Gambar 5.10.



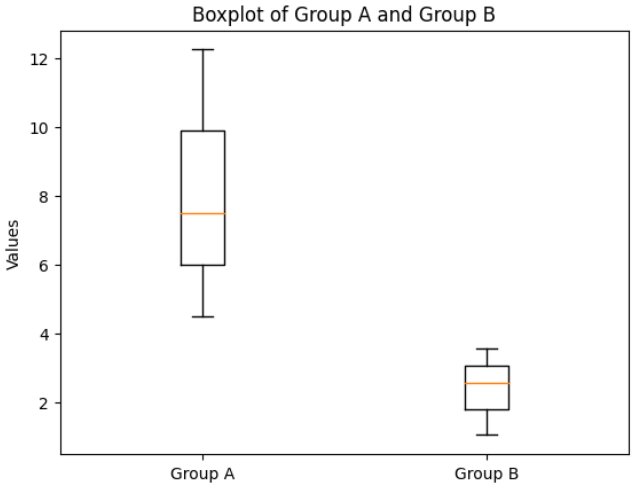
Gambar 5.7 Boxplot pasangan interaksi 1 dan 2



Gambar 5.8 Boxplot pasangan interaksi 2 dan 3



Gambar 5.9 Boxplot pasangan interaksi 1 dan 3



Gambar 5.10 Boxplot pasangan interaksi grup A dan grup B

Setelah partisipan melakukan uji coba performa pengguna dan telah mendapatkan data durasi serta telah mengolahnya, maka selanjutnya adalah partisipan mengisi kuesioner tentang System Usability Scale untuk menguji *usability* dari aplikasi XR bedah rahang. Di dalam kuesioner, pertama-tama partisipan mengisi pertanyaan karakteristik. Hasil kuesioner dari pertanyaan

karakteristik pengguna dapat dilihat pada lampiran pada bagian pertanyaan karakteristik. Setelah partisipan mengisi pertanyaan karakteristik, selanjutnya partisipan mengisi pertanyaan System Usability Scale. Hasil dari pertanyaan tersebut dapat dilihat pada lampiran bagian System Usability Scale.

Dari hasil pertanyaan karakteristik partisipan, berikut merupakan tabel hasil pertanyaan yang dapat dilihat pada Tabel 5.15**.**

Tabel 5.15 Jawaban Partisipan terhadap Pertanyaan Karakteristik

Partisipan	PK-1	PK-2	PK-3
P1	Estha	21	Laki-laki
P2	Heru Dwi Kurniawan	21	Laki-laki
P3	Pierra Muhammad Shobr	22	Laki-laki
P4	Cholid Junoto	23	Laki-laki
P5	Hilmi Zharfan Rachmadi	21	Laki-laki
P6	Bagas	22	Laki-laki

Dan pertanyaan System Usability Scale yang telah diisi oleh partisipan, dapat didata nilai SUS dari aplikasi ini pada Tabel 5.16**.**

Tabel 5.16 Jawaban Partisipan terhadap Pertanyaan System Usability Scale

Partisipan	PSUS-1	PSUS-2	PSUS-3	PSUS-4	PSUS-5	PSUS-6	PSUS-7	PSUS-8	PSUS-9	PSUS-10
P1	2	3	3	5	5	2	4	1	5	2
P2	3	2	4	5	5	4	4	4	5	5
P3	2	4	3	4	5	1	4	4	3	5
P4	4	2	4	1	5	2	5	1	5	4
P5	4	2	4	4	5	2	4	2	5	3
P6	3	3	4	3	4	2	4	2	4	3

Yang kemudian akan dihitung dengan hasil yang dapat dilihat pada Tabel 5.17**.**

Tabel 5.17 Hasil Perhitungan System Usability Scale

	PSUS-1	PSUS-2	PSUS-3	PSUS-4	PSUS-5	PSUS-6	PSUS-7	PSUS-8	PSUS-9	PSUS-10	Poin SUS
P1	2	3	3	5	5	2	4	1	5	2	65
P2	3	2	4	5	5	4	4	4	5	5	52.5
P3	2	4	3	4	5	1	4	4	3	5	47.5
P4	4	2	4	1	5	2	5	1	5	4	82.5
P5	4	2	4	4	5	2	4	2	5	3	72.5
P6	3	3	4	3	4	2	4	2	4	3	65

Menurut interpretasi hasil pengujian SUS seperti berikut:

- <50: Tidak dapat diterima
- 50 – 70: Marjinal
- >70: Dapat diterima.

Berdasarkan skor SUS dari partisipan 4 dan 5, *usability* aplikasi berada pada kategori dapat diterima. Sedangkan berdasarkan partisipan 1, 2, dan 6, *usability* dari aplikasi berada dalam

kategori marjinal. Namun berdasarkan partisipan 3, aplikasi berada pada kategori *usability* tidak dapat diterima.

5.2 Pembahasan

Pada percobaan, waktu yang diperlukan partisipan untuk menyelesaikan tugas di setiap interaksi adalah sebagai berikut:

- **I-1**: [1.53, 4.40, 3.45, 2.46, 2.12, 2.54]
- **I-2**: [1.41, 4.27, 2.09, 1.23, 2.01, 1.38]
- **I-3**: [1.16, 3.18, 1.58, 1.06, 2.58, 3.57]

Untuk mengetahui perbedaan performa pada masing-masing grup, untuk mengetahui apakah ada perbedaan yang signifikan secara statistik dalam jumlah interaksi antara dua grup (Grup A dan Grup B), jumlah interaksi untuk setiap partisipan dalam masing-masing grup adalah sebagai berikut:

- **Group A**:
  - P1: 4.50
  - P2: 12.25
  - P3: 7.52
- **Group B**:
  - P4: 5.15
  - P5: 7.11
  - P6: 8.29

Pada uji Shapiro-Wilk, interaksi pertama memiliki rata-rata 2,75, kemudian standar deviasi 1,02, median bernilai 2,50, sedangkan hasil *p-value* memiliki nilai 0,72 yang berarti interaksi pertama tidak memiliki perbedaan signifikan di seluruh interaksi. Interaksi kedua memiliki rata-rata 2,06, standar deviasi 1,13, median bernilai 1,71, dan hasil *p-value* bernilai 0,022 yang berarti interaksi kedua memiliki perbedaan yang signifikan. Interaksi ketiga memiliki rata-rata 2,18, standar deviasi 1,07, median bernilai 2,08, dan hasil *p-value* bernilai 0,35 yang berarti interaksi ketiga tidak memiliki perbedaan yang signifikan. Terakhir, total waktu semua interaksi memiliki rata-rata 7,47, standar deviasi 2,75, median 7,31, dan hasil *p-value* bernilai 0,50 yang artinya total waktu tidak memiliki perbedaan signifikan. Kemudian dalam visualisasi dengan histogram dan kurva, dapat dilihat bahwa data tampak tidak merata dan tidak mengikuti distribusi normal. Hal tersebut dapat dilihat pada kurva kepadatan yang tidak simetris dan tidak berbentuk lonceng.

Pada percobaan pengujian performa pengguna, untuk mengetahui apakah enam partisipan (P1 - P6) menunjukkan perbedaan yang signifikan secara statistik dalam waktu penyelesaian di tiga interaksi (I-1, I-2, I-3), menggunakan uji Kruskal Wallis. Dapat dilihat pada Tabel 5.9, pengujian ini menghasilkan statistik Kruskal Wallis sebesar 2,12 dan nilai *p-value* sebesar 0,35. Karena nilai *p-value* lebih besar dari tingkat signifikansi 0,05, artinya percobaan gagal menolak hipotesis nol. Hal ini mengindikasikan bahwa tidak ada perbedaan yang signifikan secara statistik dalam waktu penyelesaian di ketiga interaksi.

Hasil menunjukkan bahwa level I-1, I-2, dan I-3 tidak memiliki perbedaan yang signifikan dalam jumlah waktu yang dibutuhkan partisipan untuk menyelesaikan *task*s tersebut. Ini berarti berdasarkan waktu penyelesaian yang dicatat, tidak ada interaksi yang dapat dianggap lebih mudah atau lebih sulit secara signifikan daripada interaksi lainnya. Tidak adanya perbedaan

yang signifikan dapat menunjukkan bahwa *task*s di setiap interaksi memiliki kemiripan, atau bahwa variasi dalam kinerja setiap peserta tidak cukup signifikan untuk membedakan antara interaksi-interaksi tersebut. Akibatnya, hasil penelitian menunjukkan bahwa partisipan memiliki kinerja yang sama di ketiga interaksi.

Kemudian untuk hasil pengujian performa grup, dapat dilihat pada Tabel 5.10, pengujian ini menghasilkan statistik Kruskal Wallis sebesar 0,048 dan nilai *p-value* sebesar 0,827. Karena nilai *p-value* lebih besar dari tingkat signifikansi 0,05, artinya percobaan gagal menolak hipotesis nol. Hal ini menunjukkan bahwa tidak ada perbedaan yang signifikan secara statistik dalam jumlah interaksi antara Grup A dan Grup B.

Hasil penelitian menunjukkan bahwa tidak ada perbedaan yang signifikan antara partisipan di Grup A dan Grup B dalam jumlah durasi atau performa interaksi yang mereka lakukan. Tidak adanya perbedaan yang signifikan dapat menunjukkan bahwa distribusi tingkat kesulitan tugas dan persyaratan interaksi di kedua kelompok sebanding. Oleh karena itu, hasil analisis ini menunjukkan bahwa interaksi total peserta di kedua kelompok sebanding, menunjukkan bahwa peserta terlibat dalam tugas dan berusaha menyelesaikannya dengan cara yang sama.

Hasil pengujian juga di-*plotting* pada boxplot yang dihasilkan setelah menghitung hasil pada Python. Hasil boxplot untuk pengujian tiap interaksi dapat dilihat pada Gambar 5.5. Dalam boxplot tersebut, median waktu penyelesaian untuk interaksi pertama di 2,5 ditunjukkan, dengan Interquartile Range (IQR) berkisar antara lebih dari 2,0 dan 3,0. *Whiskers* meningkat sekitar 1,5 dan naik ke lebih dari 4,0 yang menunjukkan bahwa kisaran data tidak termasuk *outliers*. Sedangkan pada interaksi kedua, menunjukkan median waktu penyelesaian yang jauh lebih rendah, sedikit di atas 1,5. IQR cukup kecil, berada di antara sedikit di bawah 1,5 dan di nilai 2,0. Plot ini tidak memiliki *whiskers* yang sangat kecil, yang menunjukkan bahwa semua titik data mungkin berada di dalam IQR. Namun ada titik *outliers* yang sangat jauh di atas ujungnya. Terakhir, boxplot interaksi ketiga menunjukkan median waktu penyelesaian yang lebih lama dibandingkan dengan kedua interaksi sebelumnya; sumbu waktu penyelesaian berada sedikit di atas nilai 2,0. IQR berkisar dari sedikit di bawah 1,5 hingga 3,0. Di atas *whiskers* atas, tidak ada titik terpisah yang menunjukkan *outliers*. Singkatnya, *tasks* di interaksi kedua diselesaikan paling cepat, *tasks* di interaksi pertama membutuhkan waktu yang lebih lama, dan *tasks* di interaksi ketiga membutuhkan waktu yang paling lama. Ini menunjukkan variasi dalam kinerja atau efisiensi di setiap interaksi.

Hasil pengujian boxplot untuk jumlah interaksi antara dua grup, Grup A dan Grup B, dapat dilihat pada Gambar 5.6. Boxplot tersebut menunjukkan perbandingan jumlah interaksi antara dua kelompok. Boxplot untuk kelompok A menunjukkan median interaksi di sekitar 7,5, dan IQR berkisar dari sedikit di atas 6 hingga sedikit di bawah 10. Ini menunjukkan bahwa setengah dari jumlah interaksi kelompok A berada di rentang ini. *Whiskers* berkisar dari 1 hingga 12, menunjukkan variabilitas di luar IQR, tetapi tidak ada *outliers*. Di sisi lain, boxplot Grup B menunjukkan jumlah median interaksi sedikit di nilai 7, dengan IQR yang lebih rendah dari sedikit di atas 6 hingga di sekitar 7,5, menunjukkan jumlah interaksi yang lebih konsisten dibandingkan Grup A. *Whiskers* Grup B berkisar dari sekitar 5 hingga sedikit di atas 8, sekali lagi menunjukkan beberapa variabilitas tetapi tidak ada *outliers*. Secara keseluruhan, Grup A memiliki lebih banyak waktu untuk menyelesaikan interaksi, sementara Grup B memiliki konsistensi yang lebih tinggi dengan median yang sedikit lebih rendah. Perbandingan ini menunjukkan bahwa pola interaksi antara kedua kelompok berbeda.

Selanjutnya adalah dari pengujian Wilcoxon Signed-Rank. Untuk menunjukkan apakah ada

perbedaan pada pasangan I-1 dan I-2, maka dapat dilihat pada Tabel 5.11, pengujian ini menghasilkan statistik *W* sebesar 0 dan nilai *p-value* sebesar 0,03125. Karena nilai *p-value* lebih kecil dari tingkat signifikansi 0,05, artinya percobaan berhasil menolak hipotesis nol. Hal ini mengindikasikan bahwa ada perbedaan yang signifikan secara statistik antara interaksi pertama dan kedua. Kemudian, untuk pasangan I-2 dan I-3 yang dapat dilihat pada Tabel 5.12, menghasilkan statistik *W* sebesar 10,0 dan nilai *p-value* sebesar 1.0. Karena *p-value* bernilai lebih besar dari tingkat signifikansi, maka percobaan mengindikasikan bahwa tidak ada bukti statistik yang menunjukkan adanya perbedaan yang signifikan antara interaksi kedua dan ketiga. Sama seperti pasangan sebelumnya, pasangan I-1 dan I-3 juga tidak menunjukkan bukti perbedaan yang signifikan karena pada Tabel 5.13, dapat dilihat hasil statistik *W* bernilai 5,0 dan *p-value* bernilai 0,3125, masih lebih besar dari tingkat signifikansi. Terakhir, untuk pasangan Grup A dan Grup B, juga tidak menunjukkan perbedaan yang signifikan. Dapat dilihat pada Tabel 5.14, hasil uji statistik *W* bernilai 0,0 dan *p-value* bernilai 0.25. Tingkat signifikansi masih lebih kecil dari hasil *p-value* dari kedua grup.

Tidak lupa juga, hasil pengujian Wilcoxon Signed-Rank juga di-*plotting* ke dalam boxplot untuk semua pasangan. Hasil boxplot untuk pengujian untuk pasangan interaksi pertama dan interaksi kedua dapat dilihat pada Gambar 5.7, pasangan interaksi kedua dan interaksi ketiga dapat dilihat pada Gambar 5.8, dan pasangan interaksi pertama dan interaksi ketiga dapat dilihat pada Gambar 5.9. Dalam ketiga boxplot tersebut, hasil boxplot sama dengan hasil boxplot dari pengujian Kruskal Wallis. Gambar boxplot dari pengujian Wilcoxon Signed-Rank pada ketiga pasangan interaksi dengan hasil boxplot pengujian Kruskal Wallis untuk ketiga interaksi sama, dengan nilai median, IQR, dan *whiskers* yang sama.

Perbedaan boxplot ada pada pengujian pasangan grup. Hasil boxplot pengujian Wilcoxon Signed-Rank dengan Kruskal Wallis mendapati hasil yang berbeda. Untuk boxplot pengujian Wilcoxon Signed-Rank, dapat dilihat pada Gambar 5.10. Grup A memiliki nilai median di bawah 8 atau sekitar di atas 7,5, kemudian IQR di nilai 6 hingga 10, *whiskers* berkisar di atas 4 hingga di atas 12, dan grup A tidak memiliki *outliers*. Grup B menunjukkan jumlah median interaksi di atas nilai 2, dengan IQR di nilai 2 hingga di bawah 4, *whiskers* berkisar dari sekitar 1 hingga sedikit di bawah 4, dan sekali lagi menunjukkan beberapa variabilitas tetapi tidak ada *outliers*.

Kemudian dari sisi percobaan System Usability Scale, berdasarkan skor SUS pada Tabel 5.17, menurut partisipan 4 dan 5, *usability* aplikasi berada pada kategori dapat diterima. Sedangkan berdasarkan partisipan 1, 2, dan 6, *usability* dari aplikasi berada dalam kategori marjinal. Namun berdasarkan partisipan 3, aplikasi berada pada kategori *usability* tidak dapat diterima. Namun secara keseluruhan, jika dirata-rata, maka nilai *usability* aplikasi berada pada nilai 64,2 atau masih berada dalam kategori marjinal.

(halaman ini sengaja dikosongkan)

BAB 6
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Hasil dari perancangan aplikasi XR bedah rahang, pengembangan aplikasi, dan pengujian aplikasi yang telah dilakukan menghasilkan aplikasi XR yang dapat digunakan. Kesimpulan dari perancangan dan pengembangan aplikasi XR bedah rahang adalah sebagai berikut:

1. Perancangan aplikasi XR dikembangkan untuk mengatasi keterbatasan aksesibilitas, biaya tinggi, dan waktu yang lama dalam proses persiapan bedah rahang serta untuk mengetahui penggunaan dan pemahaman proses bedah rahang sebelum dan sesudah menggunakan aplikasi persiapan bedah rahang berbasis XR.
2. Aplikasi yang dikembangkan mampu membantu ahli bedah untuk mengatasi keterbatasan aksesibilitas, biaya tinggi, dan waktu yang lama dalam proses persiapan bedah rahang dengan mengimplementasikan teknologi XR.
3. Dari hasil uji Shapiro-Wilk, data tidak termasuk dalam distribusi normal, sedangkan untuk perbedaan interaksi hanya interaksi kedua saja yang memiliki perbedaan signifikan diantara yang lain.
4. Dari hasil uji Kruskal Wallis, secara keseluruhan, interaksi kedua paling cepat diselesaikan. Hasil pengujian performa menunjukkan bahwa tidak ada perbedaan yang signifikan antara interaksi, atau bahwa variasi kinerja setiap peserta tidak cukup besar untuk membedakan antara interaksi.
5. Dari pengujian Wilcoxon Signed-Rank antara pasangan interaksi, hasil menunjukkan bahwa interaksi pertama dan kedua menunjukkan perbedaan yang siginifikan namun pasangan interaksi lain menunjukkan tidak adanya perbedaan yang signifikan.
6. Hasil pengujian kedua grup secara keseluruhan menunjukkan bahwa Grup A memiliki waktu yang lebih lama untuk menyelesaikan interaksi, sementara Grup B memiliki konsistensi yang lebih tinggi dengan median yang sedikit lebih rendah.
7. Dari pengujian System Usability Scale, secara keseluruhan aplikasi masih berada dalam kategori marjinal.
8. Aplikasi dapat membantu pengguna memahami proses bedah rahang sesudah menggunakan aplikasi XR.

6.2 Saran

Dalam pengembangan aplikasi XR bedah rahang, terdapat beberapa hal yang bisa disarankan untuk melanjutkan aplikasi menjadi aplikasi yang lebih baik. Berikut merupakan saran-saran yang didapatkan untuk aplikasi XR bedah rahang.

1. Aplikasi XR perencanaan bedah rahang dapat digunakan untuk membantu dokter bedah dalam memvisualisasikan proses bedah rahang tanpa harus menggunakan alat fisik.
2. Aplikasi XR perencanaan bedah rahang dapat digunakan untuk mencoba beberapa skenario proses bedah rahang sehingga dokter bedah dapat mencoba-coba untuk memotong tengkorak di berbagai titik.
3. Aplikasi XR perencanaan bedah rahang dapat dibuat lebih intuitif, interaktif, responsif, dan mudah digunakan untuk pengguna, khususnya pengguna yang masih awam dengan aplikasi VR atau XR.

(halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

Albert, W., & Tullis, T. (2013). *Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics* (2nd ed.). Morgan Kaufmann

Bobbitt, Z. (2019, January 18). *Kruskal-Wallis Test: Definition, Formula, and Example*. Statology. Retrieved June 29, 2024, from https://www.statology.org/kruskal-wallis-test/

Dealessandri, Marie. (2020, Januari 16). What is the best game engine: is Unity right for you?. Retrieved from https://www.gamesindustry.biz/what-is-the-best-game-engine-is-unity-the-right-game-engine-for-you

Desselle, M. R., Brown, R. A., James, A. R., Midwinter, M. J., Powell, S. K., & Woodruff, M. A. (2020). Augmented and Virtual Reality in Surgery. *Computing in Science and Engineering*, *22*(3), 18–26. https://doi.org/10.1109/MCSE.2020.2972822

Educational Research Techniques. (2023, Juni 14). Wilcoxon Signed Rank Test in R. Retrieved from Educational Research Techniques: https://educationalresearchtechniques.com/2016/03/30/wilcoxon-signed-rank-test-in-r/

Frost, J. (2024). *Kruskal Wallis Test Explained*. Statistics By Jim. Retrieved June 29, 2024, from https://statisticsbyjim.com/hypothesis-testing/kruskal-wallis-test/

Hayes, A. (2023, Mei 14). Wilcoxon Test: Definition in Statistics, Types, and Calculation. Retrieved from Investopedia: https://www.investopedia.com/terms/w/wilcoxon-test.asp

KBBI. (n.d.). Arti kata kedokteran - Kamus Besar Bahasa Indonesia (KBBI) Online. Retrieved from https://kbbi.kemdikbud.go.id/entri/kedokteran

Li, C., Cai, Y., Wang, W., Sun, Y., Li, G., Dimachkieh, A. L., Tian, W., & Sun, R. (2019). Combined application of virtual surgery and 3D printing technology in postoperative reconstruction of head and neck cancers. *BMC Surgery*, *19*(1), 182. https://doi.org/10.1186/s12893-019-0616-3

Malato, G. (2023, May 3). *An Introduction to the Shapiro-Wilk Test for Normality*. Built In. Retrieved from https://builtin.com/data-science/shapiro-wilk-test

Manzie, T., MacDougall, H., Cheng, K., Venchiarutti, R., Fox, R., Sharman, A., Charters, E., Seyfi, D., Dunn, M., Mukherjee, P., & Clark, J. (2023). Virtual reality digital surgical planning for jaw reconstruction: a usability study. *ANZ Journal of Surgery*, *93*(5), 1341–1347. https://doi.org/10.1111/ans.18307

Microsoft. (2021). *Mixed Reality*. Microsoft Learn. https://learn.microsoft.com/en-us/windows/mixed-reality/discover/mixed-reality

Oh, H., Kwon, D., Ahn, J., & Paeng, J. Y. (2022). Reconstruction of mandibular defects in osteoradionecrosis and medication-related osteonecrosis of the jaw using fibula free flap and management of postoperative wound infections. *Maxillofacial Plastic and Reconstructive Surgery*, *44*(1). https://doi.org/10.1186/s40902-022-00366-2

Pu, J. J., Hakim, S. G., Melville, J. C., & Su, Y. X. (2022). Current Trends in the Reconstruction and Rehabilitation of Jaw following Ablative Surgery. *Cancers*, *14*(14), 3308. https://doi.org/10.3390/cancers14143308

Schäfer, A., Reis, G., & Stricker, D. (2022). A Survey on Synchronous Augmented, Virtual, and Mixed Reality Remote Collaboration Systems. *ACM Computing Surveys*, *55*(6). https://doi.org/10.1145/3533376

Schell, J. (2008). *The Art of Game Design: A Book of Lenses, Third Edition.* CRC Press.

Sheldon, Robert. (2022). What is virtual reality?. Retrieved from https://www.techtarget.com/whatis/definition/virtual-reality

Statisticseasily. (2024, February 27). Kruskal-Wallis test. Diakses 29 Juni 2024, dari https://statisticseasily.com/kruskal-wallis-test/

Statistic How To. (n.d.). Shapiro-Wilk Test: Definition, How to Run it in SPSS. Retrieved from https://www.statisticshowto.com/shapiro-wilk-test/

Statistics Solutions. (2023, June 14). How to Conduct the Wilcoxon Sign Test. Retrieved from Complete Dissertation: https://www.statisticssolutions.com/free-resources/directory-of-statistical-analyses/how-to-conduct-the-wilcox-sign-test/

TechCrunch. (2014, March 27). A Brief History Of Oculus. Retrieved from TechCrunch: https://techcrunch.com/2014/03/26/a-brief-history-of-oculus/

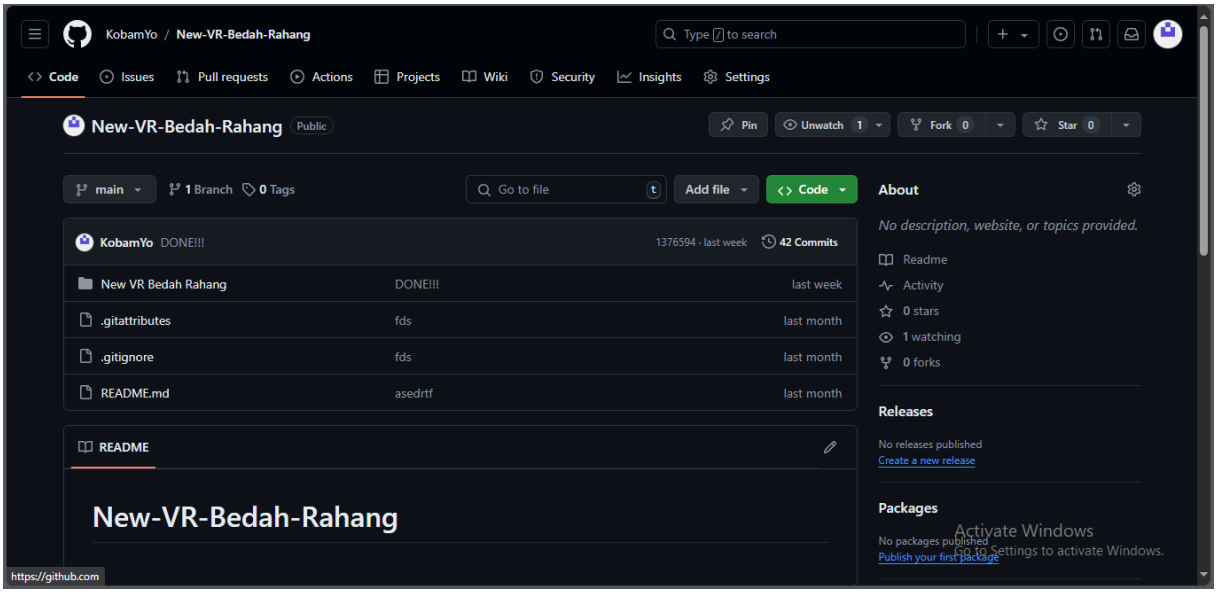
Villagran-Vizcarra, D. C., Luviano-Cruz, D., Pérez-Domínguez, L. A., Méndez-González, L. C., & Garcia-Luna, F. (2023). Applications Analyses, Challenges and Development of Augmented Reality in Education, Industry, Marketing, Medicine, and Entertainment. In *Applied Sciences (Switzerland)* (Vol. 13, Issue 5). MDPI. https://doi.org/10.3390/app13052766

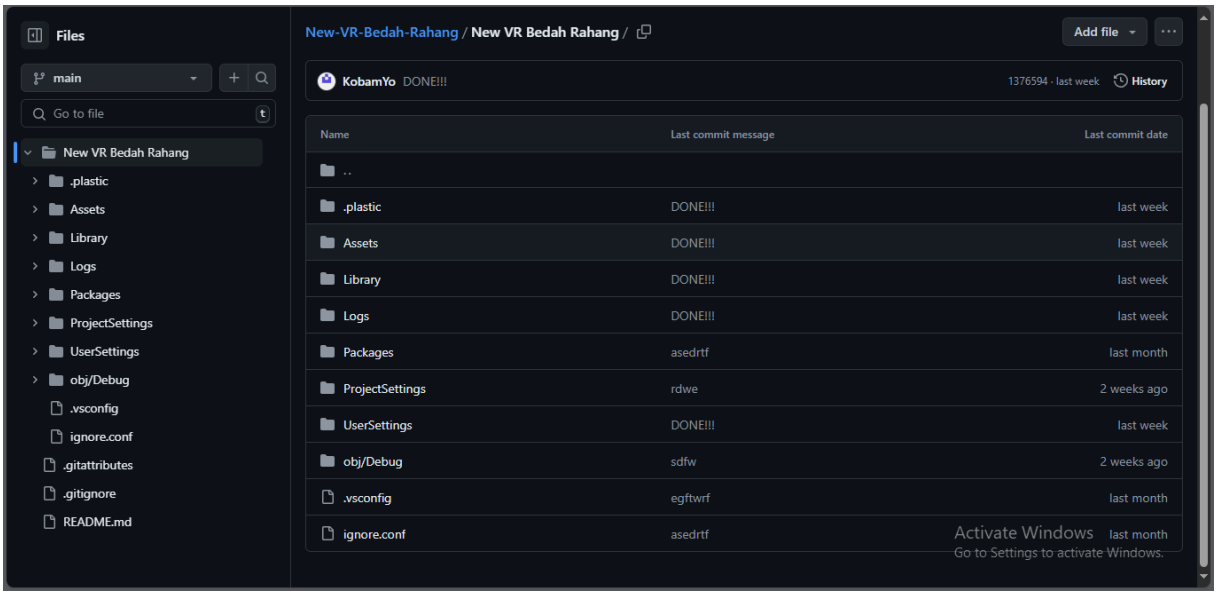
Willing, N. (2024, March 5). *Extended reality (XR)*. Retrieved July 24, 2024, from https://www.techopedia.com/definition/extended-reality-xr

Zhou, L., & Sato, R. (2020). Development and application of a surgical process simulation system using VR technology. In 2020 IEEE 9th Global Conference on Consumer Electronics (GCCE) (pp. 655-657). Kobe, Japan. https://doi.org/10.1109/GCCE50665.2020.9291758

LAMPIRAN

- Repositori proyek[2]





- Uji coba kepada partisipan
  - Hasil Kuisioner
    - Pertanyaan Karakteristik

[2] https://github.com/KobamYo/New-VR-Bedah-Rahang

Nama

6 responses

Estha

Heru Dwi Kurniawan

Bagas

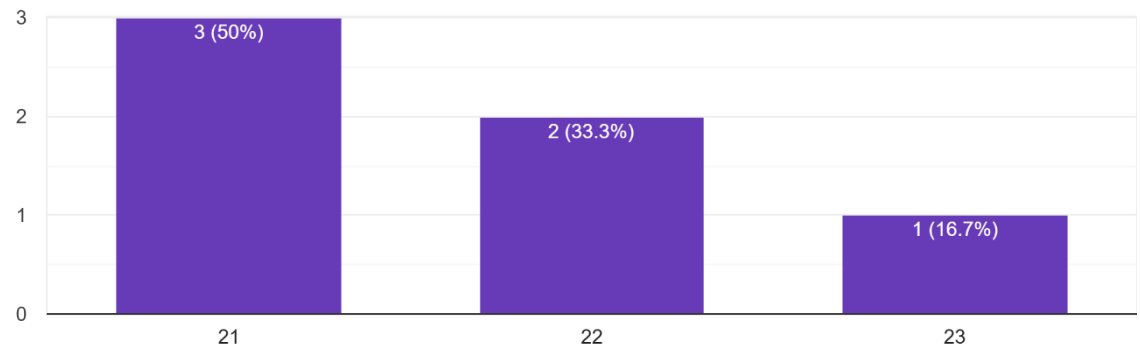
Hilmi Zharfan Rachmadi

Pierra Muhammad Shobr

Cholid Junoto

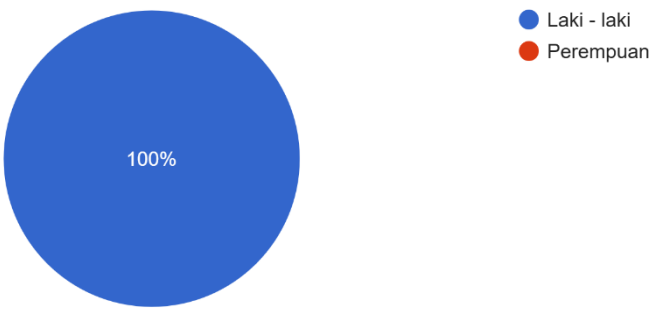
Umur

6 responses



Jenis kelamin

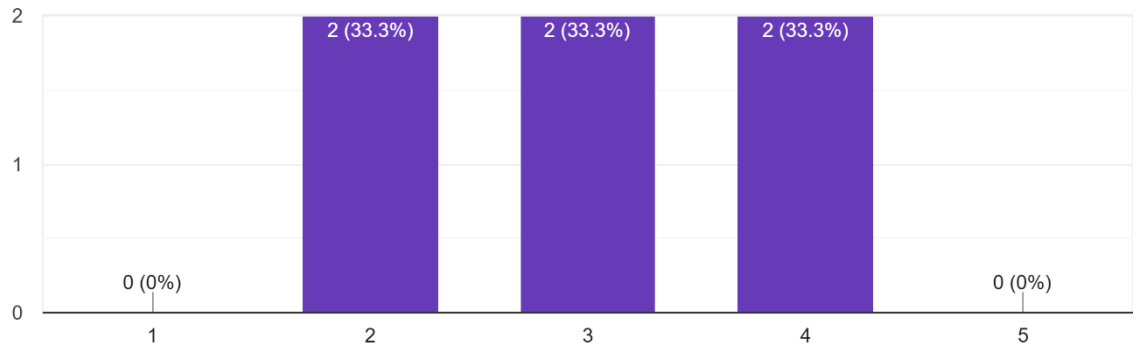
6 responses



- Pertanyaan System Usability Scale

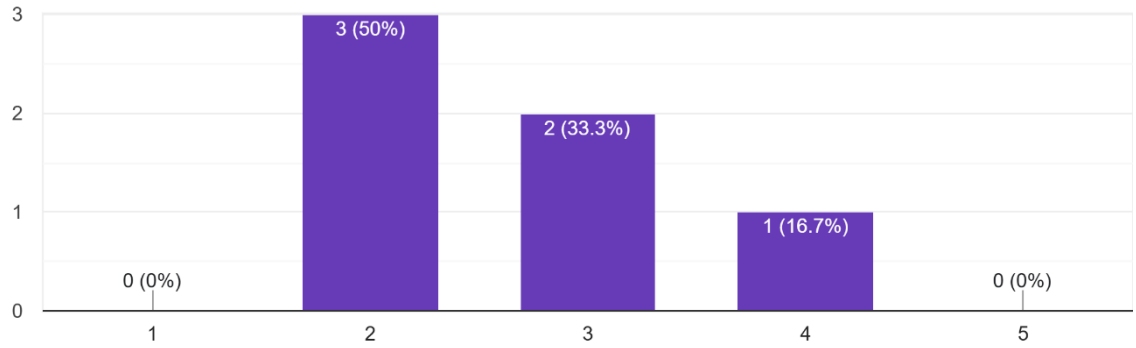
Saya pikir saya akan sering menggunakan aplikasi ini

6 responses



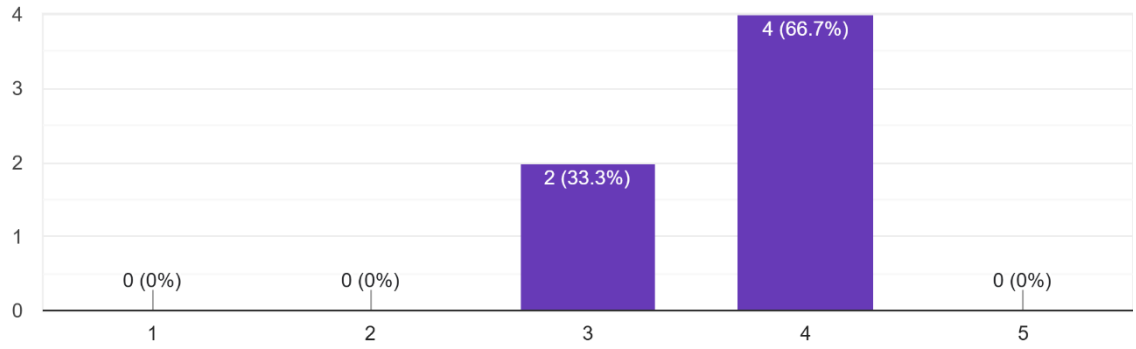
Saya menemukan aplikasi ini terlalu kompleks

6 responses



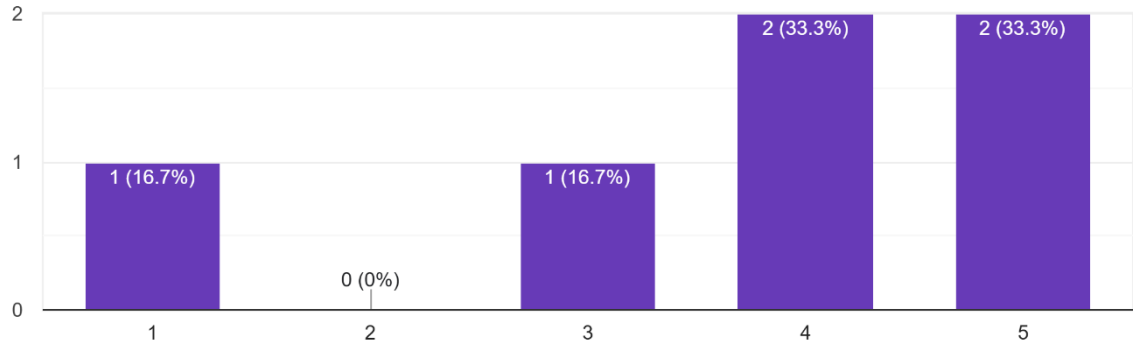
Saya merasa aplikasi ini mudah untuk digunakan

6 responses



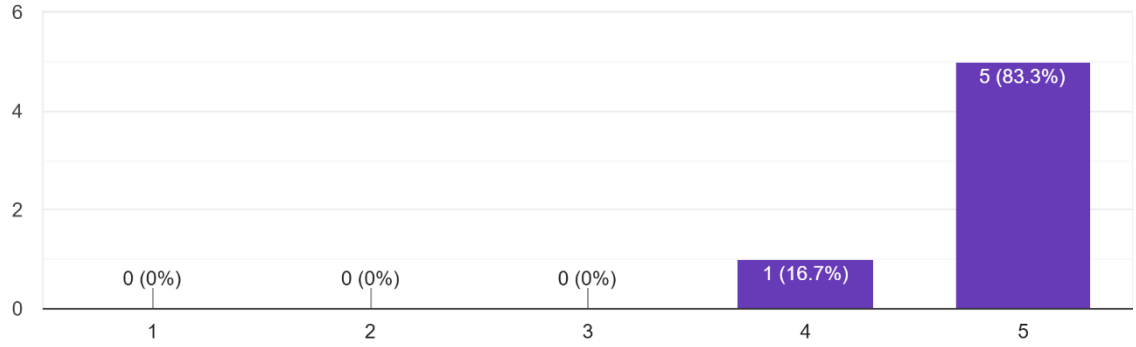
Saya pikir saya memerlukan bantuan tenaga teknis untuk dapat menggunakan aplikasi ini

6 responses



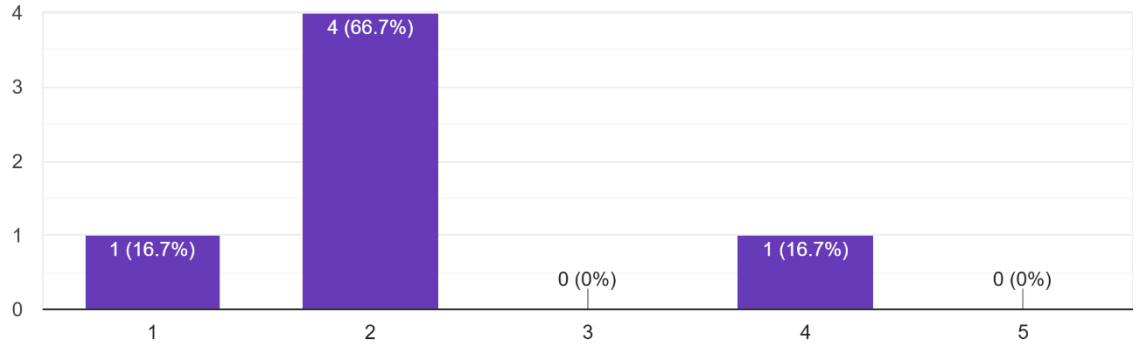
Saya merasa fungsi - fungsi dalam aplikasi ini terintegrasi dengan baik

6 responses



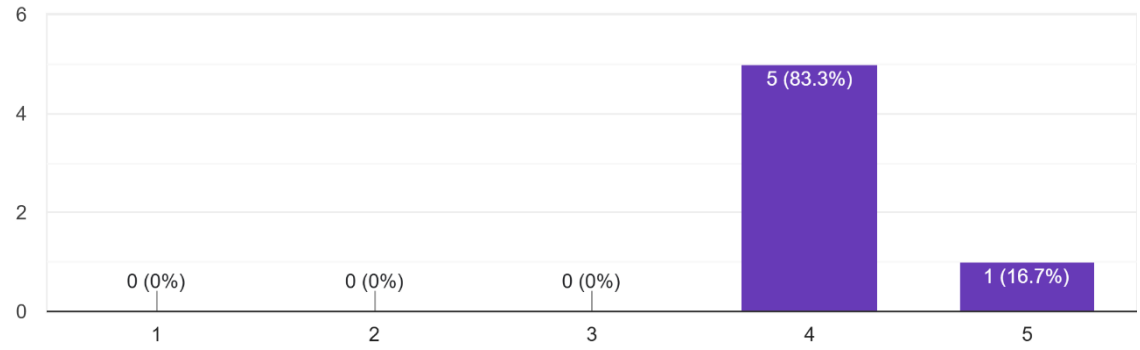
Saya merasa terdapat terlalu banyak inkonsistensi dalam aplikasi ini

6 responses



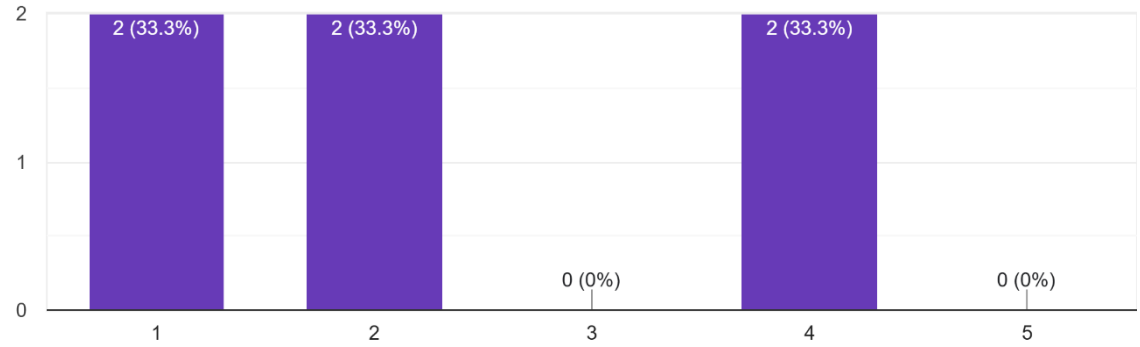
Saya membayangkan kebanyakan orang akan belajar menggunakan aplikasi ini dengan sangat cepat

6 responses



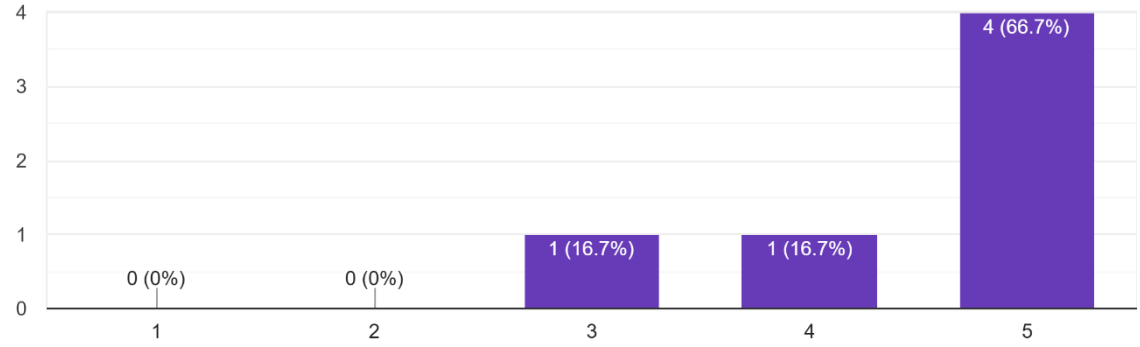
Saya menemukan aplikasi ini sangat rumit untuk digunakan

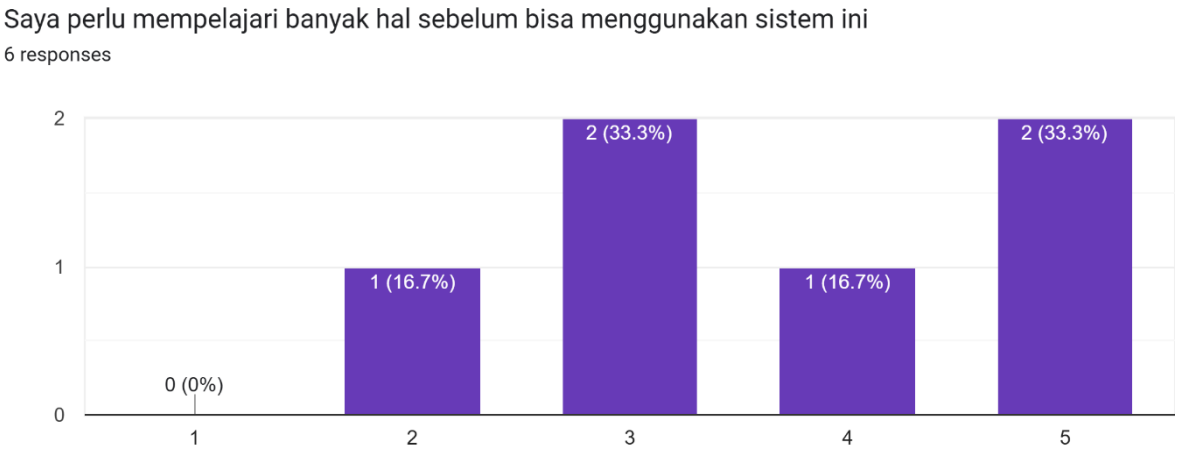
6 responses



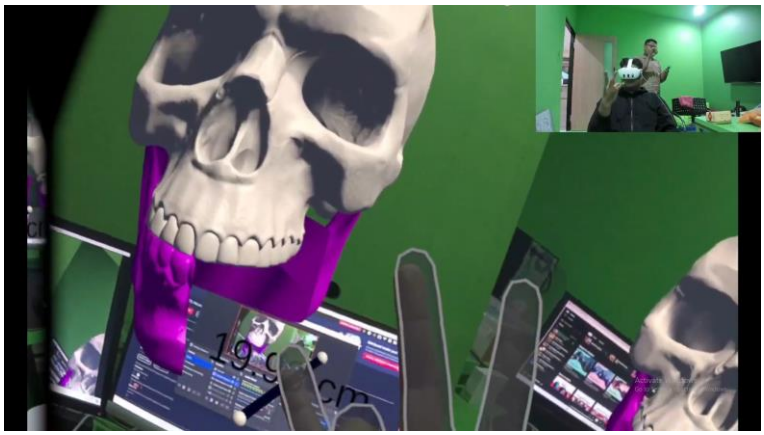
Saya merasa sangat percaya diri saat menggunakan aplikasi ini

6 responses

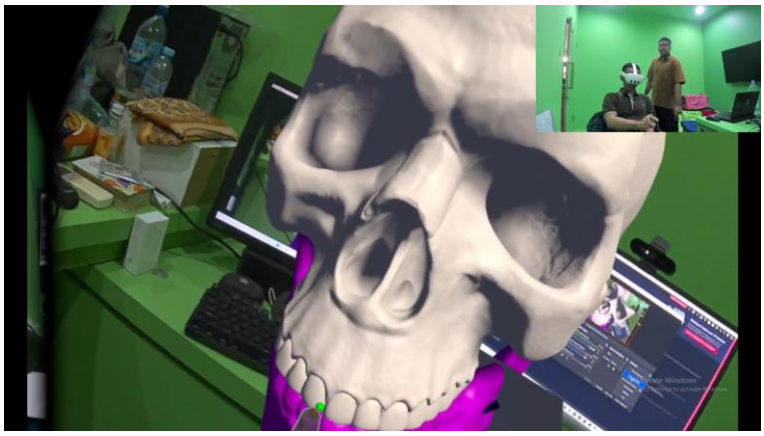



Saya perlu mempelajari banyak hal sebelum bisa menggunakan sistem ini
6 responses
2
1
0
0 (0%)
1 (16.7%)
2 (33.3%)
1 (16.7%)
2 (33.3%)
1
2
3
4
5

- Grup A:
  - **P-1**

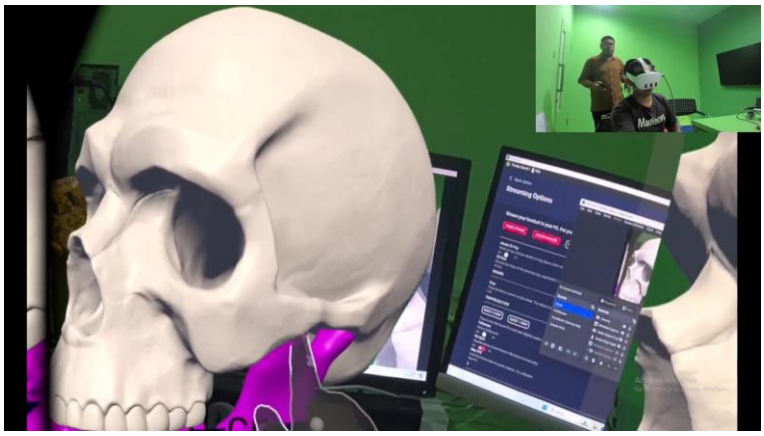


  - **P-2**



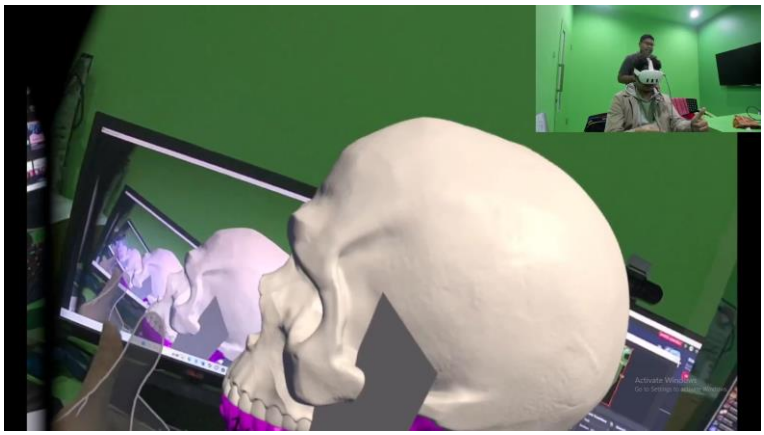
- **P-3**

- Grup B:
  - **P-4**



  - **P-5**

- **P-6**



- Kode Sumber Pengujian Shapiro-Wilk

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats
import seaborn as sns

# Data
data = {
    'I-1': [1.53, 4.40, 3.45, 2.46, 2.12, 2.54],
      'I-2': [1.41, 4.27, 2.09, 1.23, 2.01, 1.38],
      'I-3': [1.16, 3.18, 1.58, 1.06, 2.58, 3.57],
      'SUM': [4.50, 12.25, 7.52, 5.15, 7.11, 8.29]
}

df = pd.DataFrame(data)

# Shapiro-Wilk Test and Descriptive Stats
for column in df:
    W, p = stats.shapiro(df[column])
    print(f"{column}: W = {W:.3f}, p = {p:.3f}")
    print(df[column].describe(), "\n")

for column in df:
    plt.figure(figsize=(6, 4))
    sns.histplot(df[column], kde=True)
    plt.title(column)
    plt.show()
```

- Kode Sumber Pengujian Kruskal Wallis

```
import pandas as pd
import matplotlib.pyplot as plt
from scipy.stats import kruskal

data = {
    'Participant': ['A1', 'A2', 'A3', 'B1', 'B2', 'B3'],
    'Group': ['A', 'A', 'A', 'B', 'B', 'B'],
    'Interaction_1_Time': [1.53, 4.40, 3.45, 2.46, 2.12, 2.54],
    'Interaction_2_Time': [1.41, 4.27, 2.09, 1.23, 2.01, 1.38],
      'Interaction_3_Time': [1.16, 3.18, 1.58, 1.06, 2.58, 3.57],
      'Sum_of_Interactions': [4.50, 12.25, 7.52, 5.15, 7.11, 8.29]
}

# Create DataFrame
df = pd.DataFrame(data)

# Kruskal-Wallis H test for level times
```

```
    stat_interaction, p_val_interaction = kruskal(df['Interaction_1_Time'], df['Interaction_2_Time'], df['Interaction_3_Time'])
    print(f"Kruskal-Wallis H test statistic for Completion Times: {stat_interaction}, P-value: {p_val_interaction}")

    # Group data by Group A and Group B
    group_A_interactions = df[df['Group'] == 'A']['Sum_of_Interactions']
    group_B_interactions = df[df['Group'] == 'B']['Sum_of_Interactions']

    # Kruskal-Wallis H test for sum of interactions between groups A and B
    stat_interaction, p_val_interaction = kruskal(group_A_interactions, group_B_interactions)
    print(f"Kruskal-Wallis H test statistic for Sum of Interactions: {stat_interaction}, P-value: {p_val_interaction}")

    # Visualization for completion times
    data_to_plot = [df['Interaction_1_Time'], df['Interaction_2_Time'], df['Interaction_3_Time']]
    plt.boxplot(data_to_plot, labels=['Interaction 1', 'Interaction 2', 'Interaction 3'])
    plt.ylabel('Completion Time')
    plt.title('Completion Time Comparison Across Interactions')
    plt.show()

    # Visualization for sum of interactions
    plt.boxplot([group_A_interactions, group_B_interactions], labels=['Group A', 'Group B'])
    plt.ylabel('Sum of Interactions')
    plt.title('Sum of Interactions Comparison Between Groups A and B')
    plt.show()

    if p_val_interaction< 0.05:
        print("There is a significant difference in completion times across levels.")
    else:
        print("There is no significant difference in completion times across levels.")

    if p_val_interaction < 0.05:
        print("There is a significant difference in the sum of interactions between Group A and Group B.")
    else:
        print("There is no significant difference in the sum of interactions between Group A and Group B.")
```

- Kode Sumber Pengujian Wilcoxon Signed-Rank

```
import numpy as np
import scipy.stats as stats
import matplotlib.pyplot as plt

# Data Group
I_1 = np.array([1.53, 4.40, 3.45, 2.46, 2.12, 2.54])
I_2 = np.array([1.41, 4.27, 2.09, 1.23, 2.01, 1.38])
I_3 = np.array([1.16, 3.18, 1.58, 1.06, 2.58, 3.57])
Group_A = np.array([4.50, 12.25, 7.52])
    Group_B = np.array([1.06, 2.58, 3.57])

    # Perform Wilcoxon Signed Rank Test
    w_statistic, p_value = stats.wilcoxon(I_1, I_2)

    # Calculate differences
    differences = I_1 - I_2
```

```
17.   absolute_differences = np.abs(differences)
18.
19.   # Rank the absolute differences
20.   ranks = stats.rankdata(absolute_differences)
21.
22.   # Assign ranks to the positive and negative differences
23.   positive_ranks = ranks * (differences > 0)
24.   negative_ranks = ranks * (differences < 0)
25.
26.   # Create the boxplot
27.   plt.boxplot([I_1, I_2], labels=['Interaction 1', 'Interaction 2'])
28.   plt.title('Boxplot of Interaction 1 and Interaction 2')
29.   plt.ylabel('Values')
30.   plt.show()
31.
32.   # Output the results
33.   print("Wilcoxon Signed Rank Test:")
34.   print("W statistic:", w_statistic)
35.   print("p-value:", p_value)
36.
37.   print("\nDifferences:", differences)
38.   print("Absolute Differences:", absolute_differences)
39.   print("Positive Ranks:", positive_ranks)
40.   print("Negative Ranks:", negative_ranks)
41.
42.   # Perform Wilcoxon Signed Rank Test
43.   w_statistic, p_value = stats.wilcoxon(I_2, I_3)
44.
45.   # Calculate differences
46.   differences = I_2 - I_3
47.   absolute_differences = np.abs(differences)
48.
49.   # Rank the absolute differences
50.   ranks = stats.rankdata(absolute_differences)
51.
52.   # Assign ranks to the positive and negative differences
53.   positive_ranks = ranks * (differences > 0)
54.   negative_ranks = ranks * (differences < 0)
55.
56.   # Create the boxplot
57.   plt.boxplot([I_2, I_3], labels=['Interaction 2', 'Interaction 3'])
58.   plt.title('Boxplot of Interaction 2 and Interaction 3')
59.   plt.ylabel('Values')
60.   plt.show()
61.
62.   # Output the results
63.   print("Wilcoxon Signed Rank Test:")
64.   print("W statistic:", w_statistic)
65.   print("p-value:", p_value)
66.
67.   print("\nDifferences:", differences)
68.   print("Absolute Differences:", absolute_differences)
69.   print("Positive Ranks:", positive_ranks)
70.   print("Negative Ranks:", negative_ranks)
71.
72.   # Perform Wilcoxon Signed Rank Test for new pairs
73.   w_statistic, p_value = stats.wilcoxon(I_1, I_3)
74.
75.   # Calculate differences for new pairs
76.   differences = I_1 - I_3
77.   absolute_differences = np.abs(differences)
78.
79.   # Rank the absolute differences for new pairs
80.   ranks = stats.rankdata(absolute_differences)
81.
82.   # Assign ranks to the positive and negative differences for new
    pairs
83.   positive_ranks = ranks * (differences > 0)
```

```
84.    negative_ranks = ranks * (differences < 0)
85.
86.    # Create the boxplot
87.    plt.boxplot([I_1, I_3], labels=['Interaction 1', 'Interaction 3'])
88.    plt.title('Boxplot of Interaction 1 and Interaction 3')
89.    plt.ylabel('Values')
90.    plt.show()
91.
92.    # Output the results for new pairs
93.    print("Wilcoxon Signed Rank Test:")
94.    print("W statistic:", w_statistic)
95.    print("p-value:", p_value)
96.
97.    print("\nDifferences:", differences)
98.    print("Absolute Differences:", absolute_differences)
99.    print("Positive Ranks:", positive_ranks)
100.   print("Negative Ranks:", negative_ranks)
101.
102.   # Perform Wilcoxon Signed Rank Test for new pairs
103.   w_statistic, p_value = stats.wilcoxon(Group_A, Group_B)
104.
105.   # Calculate differences for new pairs
106.   differences = Group_A - Group_B
107.   absolute_differences = np.abs(differences)
108.
109.   # Rank the absolute differences for new pairs
110.   ranks = stats.rankdata(absolute_differences)
111.
112.   # Assign ranks to the positive and negative differences for new
    pairs
113.   positive_ranks = ranks * (differences > 0)
114.   negative_ranks = ranks * (differences < 0)
115.
116.   # Create the boxplot
117.   plt.boxplot([Group_A, Group_B], labels=['Group A', 'Group B'])
118.   plt.title('Boxplot of Group A and Group B')
119.   plt.ylabel('Values')
120.   plt.show()
121.
122.   # Output the results for new pairs
123.   print("Wilcoxon Signed Rank Test:")
124.   print("W statistic:", w_statistic)
125.   print("p-value:", p_value)
126.
127.   print("\nDifferences:", differences)
128.   print("Absolute Differences:", absolute_differences)
129.   print("Positive Ranks:", positive_ranks)
130.   print("Negative Ranks:", negative_ranks)
```

(halaman ini sengaja dikosongkan)

BIODATA PENULIS

Penulis dilahirkan di Bojonegoro, 12 Oktober 2001, merupakan anak ketiga dari 3 bersaudara. Penulis telah menempuh pendidikan formal yaitu di TK Pertiwi Bojonegoro, SDN Kadipaten 2 Bojonegoro, SMPN 1 Bojonegoro dan SMAN 1 Bojonegoro. Setelah lulus dari SMAN tahun 2020, Penulis mengikuti SBMPTN dan diterima di Departemen Teknik Informatika FTEIC - ITS pada tahun 2020 dan terdaftar dengan NRP 5025201217.

Di Departemen Teknik Informatika Penulis sempat aktif sebagai bagian admin dari laboratorium Grafika, Interaksi, dan Game (GIGa). Penulis tertarik terhadap Game Development, Game Design, dan Game Story Writing.