TUGAS AKHIR - KI141502

IMPLEMENTASI SMART TERRAIN PADA APLIKASI *AUGMENTED REALITY* UNTUK MEMBENTUK LINGKUNGAN 3D

BESTAMA ABHI PRIAMBADA
NRP 5111100146

Dosen Pembimbing
Dr. Darlis Heru Murti, S.Kom., M.Kom.
Ridho Rahman Hariadi, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2015

FINAL PROJECT – KI141502

IMPLEMENTATION OF SMART TERRAIN IN AUGMENTED REALITY APPLICATIONS TO BUILD A 3D ENVIRONMENT

BESTAMA ABHI PRIAMBADA
NRP 5111100146

Advisor
Dr. Darlis Heru Murti, S.Kom., M.Kom.
Ridho Rahman Hariadi, S.Kom., M.Sc.

INFORMATICS DEPARTMENT
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2015

LEMBAR PENGESAHAN

IMPLEMENTASI SMART TERRAIN PADA APLIKASI *AUGMENTED REALITY* UNTUK MEMBENTUK LINGKUNGAN 3D

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Interaksi Grafis dan Seni
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
BESTAMA ABHI PRIAMBADA
NRP. 5111100146

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr. Darlis Heru Murti, S.Kom., M.Kom. .....
   NIP: 197712172003121001 (pembimbing 1)
2. Ridho Rahman Hariadi, S.Kom., M.Sc. .....
   NIP: 198702132014041001 (pembimbing 2)

SURABAYA
JUNI, 2015

IMPLEMENTASI SMART TERRAIN PADA APLIKASI *AUGMENTED REALITY* UNTUK MEMBENTUK LINGKUNGAN 3D

Nama Mahasiswa : Bestama Abhi Priambada
NRP : 5111100146
Jurusan : Teknik Informatika FTIf-ITS
Pembimbing I : Dr. Darlis Heru Murti, S.Kom., M.Kom.
Pembimbing II : Ridho Rahman Hariadi, S.Kom., M.Sc.

ABSTRAK

Perkembangan teknologi saat ini berada pada masa yang sangat cepat. Banyak teknologi-teknologi baru bermunculan sehingga kehidupan tidak bisa terlepas dari teknologi. Realitas augmentasi merupakan jenis teknologi baru yang mulai mengisi kehidupan sehari-hari. Melalui teknologi ini sesuatu yang tidak nyata dapat terlihat menjadi hal yang nyata hanya dalah genggaman kamera.

Melalui realitas augmentasi ini, dapat digunakan sebuah teknologi yang masih terbilang baru namun sangat menarik untuk digunakan, yaitu Smart Terrain. Dengan menggunakan Smart Terrain, sebuah realitas augmentasi dapat dibangun dengan dinamis dan sesuai kehendak penggunanya. Interaksi juga dapat dilakukan pada Smart Terrain ini, sehingga realitas augmentasi bisa menjadi sangat interaktif dan dinamis. Hal itu lah yang menjadi dasar pemikiran penulis untuk menggunakan Smart Terrain pada pembangunan tugas akhir ini. Tujuannya adalah untuk membentuk sebuah lingkungan 3D, diharapkan juga pengembangan selanjutnya dapat menjadi sebuah permainan yang tentu sangat menarik untuk dimainkan.

Hasil uji coba memperlihatkan bahwa aplikasi telah dapat membangun lingkungan 3D yang peletakan objeknya sesuai dengan keinginan pengguna. Sebuah skenario sederhana yang

melibatkan interaksi pengguna juga telah dapat dijalankan dengan baik.

Kata kunci: 3D, interaksi, kamera, lingkungan, realitas augmentasi, Smart Terrain, teknologi, Unity, Vuforia.

IMPLEMENTATION OF SMART TERRAIN ON *AUGMENTED REALITY* APPLICATIONS TO BUILD A 3D *ENVIRONMENT*

Name	**: Bestama Abhi Priambada**
NRP	**: 5111100146**
Major	**: Informatics Department, FTIf-ITS**
Advisor I	**: Dr. Darlis Heru Murti, S.Kom., M.Kom.**
Advisor II	**: Ridho Rahman Hariadi, S.Kom., M.Sc.**

ABSTRACT

Nowadays, the development of technology becoming rapidly fast. Lots of new technology become available so our live depending so much on it. Augmented reality is one of a new kind of technology which is filling our daily live. With this kind of technology something unreal will become real in our eyes with only a camera.

Through this augmented reality, an interesting new technology can be use, that is Smart Terrain. With using Smart Terrain, an augmented reality can be build dynamically and with the will of the user. The interaction also can be done with this Smart Terrain, so that an augmented reality can be something interactive and dynamic. Based on that advantages, writer has an idea to use Smart Terrain for development of this final project. The purpose is to build a 3D environment. And for the next development, the writer hope that this technology could be use in a game, it will be so much interesting to play for.

The result of the test shown that the application can build a 3D environment which the placement of the object depend on the user. A simple scenario that uses user interaction also can be done and run perfectly.

Keywords: 3D, augmented reality, camera, environment, interaction, Smart Terrain, technology, Unity, Vuforia.

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Dengan menyebut nama Allah SWT yang Maha Pengasih lagi Maha Panyayang, kami panjatkan puja dan puji syukur atas kehadirat-Nya, yang telah melimpahkan rahmat, hidayah, dan inayah-Nya kepada kami, sehingga penulis dapat menyelesaikan tugas akhir ini dengan baik.

Melalui lembar ini, penulis ingin menyampaikan rasa hormat dan terima kasih yang setinggi-tingginya kepada pihak-pihak yang telah membantu penulis dalam penyelesaian tugas akhir ini, terutama kepada:

1. Allah SWT atas limpahan rahmat dan rezeki-Nya sehingga penulis mendapatkan kesehatan untuk menyelesaikan tugas akhir.
2. Bapak Budi Subagio dan Ibu Esti Ebhi Evolisa, orang tua penulis, yang selalu memberikan dukungan dan semangat baik serta doa yang selalu dikirimkan sehingga penulis mampu menyelesaikan tugas akhir ini dengan benar, tepat waktu, dan lancar.
3. Kedua adik penulis, Bestari Ebhi Kurnianing Ramadhanis dan Bestrigaswara Abhimukti Permana, yang selalu menghibur penulis, memberi semangat, dan memberikan doa agar penulis mampu menyelesaikan tugas akhirnya
4. Bapak Dr. Tohari Ahmad, S.Kom., MIT. yang telah menjadi dosen wali penulis selama berkuliah di Teknik Informatika sehingga penulis mendapatkan banyak saran tentang perkuliahan di Teknik Informatika.
5. Bapak Dr. Darlis Heru Murti, S.Kom., M.Kom. dan Bapak Ridho Rahman Hariadi, S.Kom., M.Sc. yang telah bersedia menjadi dosen pembimbing tugas akhir penulis sehingga penulis dapat mengerjakan tugas

akhir dengan arahan dan bimbingan yang baik dan benar.

6. Bapak Imam Kuswardayan, S.Kom., M.T. yang telah memberikan banyak saran tentang ide dari pembuatan tugas akhir ini pada tahap proposal.
7. Bapak dan Ibu dosen Teknik Informatika yang telah memberikan ilmu kepada penulis selama berkuliah di Teknik Informatika.
8. Andina Trya Ramadhayanti yang terus memberikan penulis semangat untuk mengerjakan tugas akhir dari awal sampai akhir.
9. Erick Hendra Putra Alwando dan Danang Prawira Nugraha, teman seperjuangan dari Mataram yang telah memberikan inspirasi tersendiri bagi penulis.
10. Rahmat, Devin, Erick, Didi, Faldi, dan Chandra, teman asrama penulis selama tahun pertama tinggal di Surabaya, sehingga penulis dapat berjuang merantau sampai saat ini.
11. Teman-teman kontrakan, Rahman, Tommy, Toto, Punggi, Faris, Ruslan, Andrie, Dapik, Yunus, Tev, dan Risal yang telah tinggal bersama selama tiga tahun di kota Surabaya, sehingga penulis mendapatkan kebersamaan untuk tinggal merantau di Kota Surabaya ini.
12. Teman-teman Lab IGS, yang telah berjuang bersama untuk menyelesaikan tugas akhir ini.
13. Teman-teman BPH Schematics 2013 serta panitia pejuang Schematics 2013, yang telah memberikan pengalaman yang tak akan pernah dilupakan penulis selama kuliah di Teknik Informatika.
14. Teman-teman mahasiswa Teknik Informatika 2011, yang telah berjuang bersama selama berkuliah di Teknik Informatika.

15. Serta pihak-pihak lain yang turut membantu penulis baik secara langsung maupun tidak, yang namanya tidak penulis sebutkan disini.

Penulis telah berusaha sebaik-baiknya dalam menyusun tugas akhir ini, mohon maaf apabila ada kesalahan dan kata-kata yang dapat menyinggung perasaan.

Surabaya, Juni 2015

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE SUMBER

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar tugas akhir yang meliputi latar belakang, tujuan, rumusan dan batasan masalah, metodologi pembuatan tugas akhir, serta sistematika penulisan.

1.1. Latar Belakang

Kemajuan teknologi saat ini telah berada pada kecepatan yang sangat cepat, perkembangan-perkembangan yang telah dicapai sekarang telah menjadi kenyataan dari apa yang diinginkan di masa lalu. Dimulai dengan munculnya sebuah kamera tradisional hingga teknologi canggih yang ada didalamnya, dan bahkan saat ini penggunaan kamera sudah menjadi hal yang lumrah dan biasa-biasa saja karena sudah terdapat hampir dimanapun, seperti laptop, *smartphone*, *tablet*, dan lain sebagainya.

Teknologi *augmented reality* merupakan sebuah teknologi yang memungkinkan pengguna untuk melihat sesuatu yang diinginkan untuk muncul di layar *smartphone* dalam bentuk digital melalui tangkapan kamera. Hal ini tentu sangat menarik karena pengguna dapat berinteraksi secara langsung melalui apa yang dilihatnya pada layar *smartphone*. Oleh sebab itu pembuatan aplikasi menggunakan teknologi ini menjadi suatu hal yang sangat bagus.

Namun teknologi tersebut masih belum banyak digunakan sampai saat ini, hal tersebut dikarenakan sulitnya membentuk dunia yang dinamis menggunakan *augmented reality*.

Untuk mengatasi masalah tersebut, maka digunakanlah teknologi baru bernama Smart Terrain. Smart Terrain adalah sebuah terobosan baru dari Vuforia SDK yang memungkinkan untuk membuat sebuah hal yang baru dalam pengalaman aplikasi sampai *gaming* didalam *augmented reality*. Teknologi Smart

Terrain memungkinkan pengembang untuk membangun sebuah dunia *augmented reality* pada lingkungan nyata. Hal ini dapat membantu membentuk sebuah lingkungan dinamis dimana peletakannya ditempatkan sesuai dengan keinginan pengguna.

Dalam tugas akhir ini, diharapkan dapat menjelaskan bagaimana penggunaan Smart Terrain untuk pengembangan aplikasi kedepannya, apa saja kekurangan dan kelebihannya, siapa saja yang dapat menggunakan teknologi ini, serta sejauh mana teknologi ini dapat mencakup kebutuhan *game* saat ini.

1.2. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam tugas akhir ini dipaparkan sebagai berikut:

1. Bagaimana membuat aplikasi dengan teknologi Smart Terrain?
2. Bagaimana mengambil *mesh* dari objek asli sebagai *input* untuk Smart Terrain?
3. Bagaimana membangun lingkungan 3D menggunakan Smart Terrain?
4. Bagaimana hasil sebuah augmented reality jika digunakan untuk sebuah skenario sederhana?

1.3. Batasan Permasalahan

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, antara lain:

1. Perangkat lunak berbasis *mobile*, yaitu Android
2. Perangkat yang digunakan harus memiliki kamera
3. Menggunakan bahasa pemrograman C# dengan *Game Engine* Unity *Free License*
4. Menggunakan Vuforia SDK untuk *augmented reality*
5. Hasil adalah dalam bentuk *terrain* lingkungan 3D

1.4. Tujuan

Tujuan dari pembuatan tugas akhir ini antara lain:

1. Mengeksplorasi teknologi Smart Terrain untuk pengembangan aplikasi pembangkitan lingkungan 3D
2. Mengetahui bagaimana hasil sebuah *augmented reality* jika dibangun dikombinasikan dengan Smart Terrain
3. Pemanfaatan *augmented reality* dan Smart Terrain sebagai bahan dasar untuk pembuatan *game*

1.5. Manfaat

Manfaat dari hasil pembuatan tugas akhir ini antara lain:

1. Memberikan wawasan mengenai *augmented reality* dengan menggunakan teknologi Smart Terrain
2. Membuat *augmented reality* sebagai sebuah hiburan untuk semua kalangan pengguna

1.6. Metodologi

Tahap-tahap pembuatan tugas akhir ini dilakukan menggunakan metodologi sebagai berikut:

1. Penyusunan proposal tugas akhir
   Proposal tugas akhir ini berisi tentang deskripsi pendahuluan tugas akhir yang akan dibuat. Pendahuluan terdiri atas hal yang menjadi latar belakang diusulkannya tugas akhir, rumusan masalah yang diangkat, batasan masalah, tujuan, dan manfaat dari hasil pembuatan tugas akhir. Terdapat pula detail mengenai tugas akhir yang akan dibuat pada tinjauan pustaka dan ringkasan isi tugas akhir. Selain itu terdapat merodologi, jadwal kegiatan, dan daftar pustaka.
2. Studi literatur
   Pada tahap studi literatur akan dipelajari beberapa referensi yang dibutuhkan dalam perancangan dan pembuatan aplikasi.

Referensi tersebut diantaranya adalah mengenai Unity, *augmented reality*, Vuforia, Smart Terrain, *modelling*, dan animasi.

3. Perancangan perangkat lunak
   Pada tahap ini dilakukan analisa awal dan pendefinisian kebutuhan sistem untuk mengetahui permasalahan yang sedang dihadapi. Kemudian dirumuskan dalam bentuk rancangan sistem yang dapat memberi solusi terhadap permasalahan tersebut.
4. Implementasi perangkat lunak
   Tahap implementasi merupakan tahap untuk membangun aplikasi Smart Terrain. Aplikasi ini dibangun menggunakan *framework* Unity 3D, Bahasa pemrograman C#, dibantu dengan Vuforia SDK, serta alat bantu pemodelan Blender.
5. Pengujian dan evaluasi
   Pada tahap ini akan dilakukan pengujian secara *blackbox* menggunakan perangkat *smartphone* berbasis Android, dengan pengujian tingkat pembacaan aplikasi terhadap berbagai target objek yang diberikan.
6. Penyusunan laporan tugas akhir
   Pada tahap ini dilakukan penyusunan laporan yang berisi dasar teori, dokumentasi dari perangkat lunak, dan hasil-hasil yang diperoleh selama pengerjaan tugas akhir.

1.7. Sistematika Penulisan

Buku tugas akhir ini terdiri dari beberapa bab, yang dijelaskan sebagai berikut.

BAB I PENDAHULUAN
Bab ini berisi latar belakang masalah, rumusan dan batasan permasalahan, tujuan dan manfaat pembuatan tugas akhir, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

BAB II DASAR TEORI
Bab ini membahas dasar pembuatan dan beberapa teori penunjang yang berhubungan dengan pokok pembahasan yang mendasari pembuatan tugas akhir ini.

BAB III ANALISIS DAN PERANCANGAN SISTEM
Bab ini membahas analisis dari sistem yang dibuat meliputi analisis permasalahan, deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifikasi pengguna. Kemudian membahas rancangan dari sistem yang dibuat meliputi rancangan skenario kasus penggunaan, arsitektur, data, dan antarmuka.

BAB IV IMPLEMENTASI
Bab ini membahas implementasi dari rancangan sistem yang dilakukan pada tahap perancangan. Penjelasan implementasi meliputi implementasi pembangkitan area permainan, dan antarmuka permainan.

BAB V PENGUJIAN DAN EVALUASI
Bab ini membahas pengujian dari aplikasi yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan evaluasi untuk mengetahui kemampuan aplikasi.

BAB VI KESIMPULAN DAN SARAN
Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan serta saran untuk pengembangan aplikasi selanjutnya.

.

BAB II
DASAR TEORI

Pada bab ini akan dibahas mengenai teori-teori yang menjadi dasar dalam pembuatan tugas akhir ini. Pokok permasalahan yang dibahas adalah mengenai teknologi yang mendukung pembuatan tugas akhir seperti *augmented reality*, Vuforia, Smart Terrain, dan Unity3D.

2.1. Rancang Bangun Perangkat Lunak

Rancang bangun perangkat lunak merupakan tahap-tahap teknis untuk membangun perangkat lunak yang melingkupi analisis permasalahan dan kebutuhan, perencanaan, analisis sistem, implementasi, serta aktivitas pengujian dan pemeliharaan perangkat lunak. Rancang bangun perangkat lunak diperlukan untuk menentukan konsep, strategi, dan praktik yang baik diterapkan untuk menciptakan perangkat lunak yang berkualitas tinggi, sesuai anggaran biaya, mudah dalam pemeliharaannya, serta tidak membutuhkan waktu yang lama dalam pembangunannya. Beberapa model rancang bangun perangkat lunak yang terkenal dan banyak dipakai antara lain model air terjun dan model iterasi.

2.2. Unity3D *Game Engine*

Unity merupakan sebuah ekosistem pengembangan *game*, mesin *render* yang kuat dan terintegrasi dengan satu set lengkap alat intuitif dan alur kerja yang cepat untuk membuat konten 3D interaktif. Unity merupakan suatu aplikasi yang digunakan untuk mengembangkan *game multiplatform* yang didesain dengan aplikasi yang profesional. *Editor* pada Unity dibuat dengan *user interface* yang sederhana dan mudah dipahami. Secara rinci, Unity dapat digunakan untuk membuat *game* 3D, *real-time*

animasi 3D dan 2D, visualisasi arsitektur, *augmented reality*, dan banyak lainnya.

Unity adalah *software* yang bagus untuk mengembangkan *game* 3D, untuk lebih tepatnya disebut sebagai *game engine*. Setelah rilis pertamanya pada tahun 2005, banyak pembaruan yang ditambahkan selama tahun perilisannya, sekarang Unity telah berada pada versi 5 [1]. Dan sampai saat ini telah banyak *game-game* yang dikembangkan menggunakan Unity seperti WolfQuest, Tiger Woods PGA Tour *Online*, Temple Run Oz, dan banyak *game* baru lainnya.

Fitur-fitur yang terdapat pada Unity antara lain:

a. *Rendering*
   Digunakan untuk menampilkan animasi ketika sedang dikembangkan maupun setelah selesai, graphics *engine* yang digunakan adalah Direct3D (Windows, Xbox 360), OpenGL (Mac, Windows, Linux, PS3), OpenGL ES (Android, iOs), dan *proprietary* APIs (Wii).
b. *Scripting*
   Script game engine dibuat dengan MonoDevelop, sebuah implementasi *open-source* dari .NET *Framework*. Dengan fitur ini, pengembang dapat melakukan kustomisasi *game* dan animasi melalui *code* yang terintegrasi.
c. *Asset Tracking*
   Unity juga menyertakan *Server* Unity *Asset*, sebuah solusi terkontrol untuk *developer game asset* dan *script*.
d. *Platforms*
   Unity mendukung pengembangan ke berbagai *platform*. Didalam *project*, pengembang memiliki kontrol untuk mengirim ke perangkat *mobile*, *web browser*, *desktop*, dan juga *console*. Unity juga mengijinkan spesifikasi kompresi tekstur dan pengaturan resolusi di setiap *platform* yang didukung
e. *Asset Store*
   Unity *Asset* Strore adalah sebuah resource yang hadir di Unity *editor*. *Asset store* terdiri dari koleksi lebih dari 4400 *asset*

packages, beserta model 3D, tekstur dan material, partikel, music dan efek suara, tutorial dan projek, *scripting package*, ekstensi *editor* dan servis *online* [2].

f. *Physics*
Unity mendukung *physics* yang berguna sebagai sifat dari objeknya, seperti massa, *collision*, gravitasi, dan lain sebagainya.Unity3D

Unity merupakan sebuah *game engine* yang dikembangkan oleh Unity Technologies, unity sendiri dapat menciptakan *game* ke dalam beberapa sistem operasi sekaligus. Antara lain: Windows Phone, Android, iOS, Windows 8, OSX, Blackberry 10, Playstation 3, Playstation 4, XBOX, dan sebagainya. *Game* - *game* yang dapat dibuat oleh unity ini bisa dalam bentuk 3d ataupun 2d, tergantung pada pengembang *game* tersebut. Unity mampu untuk mengubah gambar statis menjadi animasi yang dapat dimainkan, maka dari itu banyak partner yang menggunakan unity. Antara lain: Micro*soft*, Sony, Qualcomm, Blackberry, Samsung, dan Nintendo.

Dari segi produk unity terdiri dari dua macam, yakni Unity *Free* dan Unity Pro, tentunya jika kita ingin menggunakan Unity Pro maka akan ada sejumlah uang yang digunakan untuk membelinya.

2.3. *Augmented Reality*

Augmented reality adalah sebuah teknologi yang menggabungkan benda maya dua dimensi ataupun tiga dimensi ke dalam sebuah lingkungan nyata tiga dimensi, lalu memproyeksikan benda-benda maya tersebut dalam waktu nyata [3].

Benda-benda berbentuk maya yang sering dilihat dalam layar *smartphone* bisa dibentuk seakan-akan benda tersebut nyata, karena mengggunakan kamera sebagai perantaranya. Dengan ditambahkan interaksi, maka realitas yang dihasilkan akan bertambah.

Dengan adanya teknologi *augmented reality* ini, dapat membantu di berbagai bidang untuk mempermudah aktifitas, salah satunya adalah di dunia hiburan. Salah satu bentuk contoh *augmented reality* menggunakan *smartphone* bisa dilihat pada Gambar 2.1.

Gambar 2.1 Contoh *Augmented reality* [3]

2.4. Vuforia SDK

Vuforia adalah *Augmented reality Software Development Kit* (SDK) untuk perangkat mobile yang memungkinkan pembuatan aplikasi *Augmented reality*. Vuforia menggunakan teknologi *Computer Vision* untuk mengenali dan melacak gambar planar (*Image Target*) dan objek 3D sederhana, seperti kotak, secara *real-time* [4]. Kemampuan registrasi citra ini memungkinkan pengembang untuk mengembangkan sebuah objek dengan posisi dan orientasi objek virtual, seperti model 3D dan media lainnya, dalam kaitannya dengan gambar dunia nyata ketika hal ini dilihat melalui kamera dari perangkat *mobile*. Objek virtual kemudian melacak posisi dan orientasi dari gambar secara

BAB III
ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini akan membahas tahapan-tahapan analisis permasalahan dan perancangan dari sistem yang dibangun. Terdiri dari analisis perancangan pengerjaan tugas akhir, serta perancangan dari hasil analisis tersebut untuk menyelesaikan permasalahan tersebut.

3.1. Analisis Perangkat Lunak

Pada subbab ini akan dibahas mengenai analisis permasalahan dari aplikasi yang dibangun, meliputi deskripsi perangkat lunak, spesifikasi kebutuhan perangkat lunak, dan identifikasi pengguna.

3.1.1. Deskripsi Umum Perangkat Lunak

Perkembangan teknologi saat ini membuat sesuatu yang di masa dahulu tidak mungkin, menjadi mungkin di masa sekarang. Penggunaan kamera sebagai dasar untuk bermain *game* menjadi hal yang seakan biasa di masa sekarang. Dan dengan adanya teknologi *augmented reality* dan Smart Terrain, pengembangan *game* yang berbasis kamera menjadi sangat terbantu.

Smart Terrain lebih bagus saat digunakan dengan benda-benda yang diam dengan pencahayaan yang stabil. Permukaan benda seharusnya tidak mengkilap dan transparan, karena benda seperti itu bisa mengganti posisi awal dari pengguna. Benda sebagai objek yang digunakan untuk Smart Terrain bisa jadi sekecil mangkok sup, atau juga sebesar kotak sereal. Bentuknya haruslah mempunyai bidang yang tetap dan permukaannya harus mempunyai *pattern* dan detail yang bisa digunakan oleh *tracker* Smart Terrain untuk mengenali dan membaca objek. Jumlah objek yang dibaca secara optimal adalah lima objek.

Smart Terrain membangun ulang, mengenali, dan melacak objek fisik dan permukaannya. Objek tersebut direpresentasikan

dalam bentuk 3D *mesh* di dalam Unity. Kumpulan dari permukaan dan *mesh* dari objek merupakan *input* untuk membuat *terrain* tersebut, Gambar 3.1 menunjukkan benda-benda yang menjadi objek untuk *input* pada Smart Terrain.

Gambar 3.1 *Props* [7]

Gambar 3.2 Hasil Smart Terrain [5]

Hasil pembangkitan *augmented reality* dengan menggunakan Smart Terrain dapat dilihat pada Gambar 3.2.

3.1.2. Spesifikasi Kebutuhan Perangkat Lunak

Pada subbab ini akan membahas kebutuhan yang diperlukan oleh perangkat lunak tugas akhir ini, meliputi kebutuhan fungsional dan kebutuhan non-fungsional.

3.1.2.1. Kebutuhan Fungsional Perangkat Lunak

Pada aplikasi ini terdapat beberapa kebutuhan fungsional yang harus dipenuhi untuk mendukung jalannya aplikasi, antara lain:

a) Deteksi *props*

Aplikasi mampu untuk melakukan pendeteksian terhadap objek nyata didalam *stage*. Objek nyata ini bisa berupa apa saja, kotak, botol, buku, dan lain-lain. Kebutuhan ini menjadi dasar dari aplikasi Smart Terrain. Objek nyata yang dideteksi juga harus memenuhi kriteria *props* yaitu ukuran dimulai sekecil mangkok sambal hingga sebesar kotak mineral, bentuknya haruslah mempunyai bidang yang tetap, serta permukaannya harus mempunyai *pattern* dan detail yang bisa digunakan oleh *tracker* Smart Terrain untuk mengenali dan membaca objek. Jumlah objek yang dibaca secara optimal adalah lima objek.

b) Pembangkitan *terrain*

Pembangkitan *terrain* dilakukan setelah deteksi *props* selesai dilakukan. Smart Terrain akan membangun ulang, mengenali, dan melacak objek fisik dan permukaannya. Objek akan direpresentasikan dalam bentuk 3D *mesh* dalam Unity. Dan berdasarkan *props* tersebut akan dibangun lingkungan 3D yang dinamis. Pada awalnya, perilaku Smart Terrain akan melakukan pembangkitan secara otomatis dan langsung ketika *props* dideteksi pertama kali. Karena hal tersebut mengakibatkan mesh tidak dapat dilihat maka pembangkitan dimodifikasi menjadi

manual, yaitu pembangkitan dilakukan setelah pengguna memilih pilihan untuk melakukan pembangkitan, sehingga *mesh* dari *props* dapat terlihat sebelum proses pembangkitan dilakukan.

c) Interaksi karakter

Kebutuhan ini diperlukan untuk mengenali objek-objek yang ada. Akan disediakan sebuah karakter yang akan berjalan sesuai *tap* pengguna di layar *smartphone*. Fungsi ini menggunakan *raycast* dimana akan mendapatkan lokasi *tap* pengguna. Interaksi ini juga berguna untuk mengetahui *collider* dari objek yang dibangun.

3.1.2.2. Kebutuhan Non-Fungsional

Pada aplikasi ini terdapat beberapa kebutuhan tambahan yang akan menambah performa aplikasi bila dipenuhi. Kebutuhan tersebut adalah sebagai berikut:

1. Pencahayaan ruangan

Aplikasi ini adalah aplikasi yang membaca objek nyata dari *stage* yang disediakan. Pencahayaan yang bagus akan membantu membaca objek dengan bagus pula. Didalam ruangan dengan cahaya yang minim akan menyulitkan Smart Terrain untuk mendeteksi objek. Namun dengan cahaya terang dari beberapa model lampu akan menimbulkan *flicker* yang mengganggu kinerja pula.

2. Penggunaan kamera resolusi tinggi

Aplikasi ini sangat bergantung dari kamera, karena kamera yang melakukan semua hal dimulai dari deteksi, memindai, dan membaca objek nyata. Resolusi yang rendah tentu berpengaruh terhadap kualitas gambar yang dihasilkan, sehingga disarankan memakai kamera dengan resolusi minimal 8MP untuk menjaga kualitas dari aplikasi. Adanya flash juga akan membantu kualitas dari aplikasi.

3.1.3. Identifikasi Pengguna

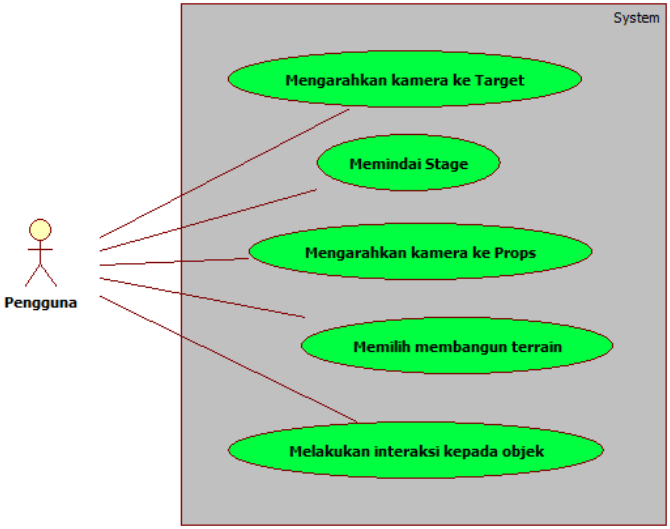
Dalam aplikasi tugas akhir ini, hanya terdapat satu pengguna yang terlibat langsung, yaitu orang yang akan menjalankan aplikasi ini.

3.2. Perancangan Perangkat Lunak

Pada subbab ini akan membahas bagaimana rancangan dari aplikasi tugas akhir ini. Meliputi diagram kasus penggunaan, definisi aktor, definisi kasus penggunaan, arsitektur, rancangan antarmuka, dan rancangan proses aplikasi.

3.2.1. Model Kasus Penggunaan

Diagram kasus penggunaan aplikasi ini berpacu pada kebutuhan yang didefinisikan. Model diagram kasus penggunaan aplikasi ini dapat dilihat pada Gambar 3.3.



Gambar 3.3 Diagram Kasus Penggunaan

3.2.2. Definisi Aktor

Aktor yang terdapat pada aplikasi ini dapat dilihat pada Tabel 3.1.

Tabel 3.1 Definisi Aktor

No	Nama	Deskripsi
1	Pengguna	Merupakan aktor utama dan satu-satunya dalam aplikasi ini. Bertugas untuk menjalankan aplikasi, melakukan pemindaian, dan seluruh fungsionalitas yang ada dalam sistem

3.2.3. Definisi Kasus Penggunaan

Pada bagian ini akan dijelaskan detail mengenai kasus penggunaan yang telah disebutkan sebelumnya. Detail tersebut dapat dilihat pada Tabel 3.2.

Tabel 3.2 Definisi Kasus Penggunaan

No.	**Kode Kasus Penggunaan**	**Nama Kasus Penggunaan**	**Keterangan**
1	UC-01	Mengarahkan kamera ke target	Pengguna bisa mengarahkan kamera ke mana saja, namun untuk dapat mengaktifkan fungsi Smart Terrain, pengguna harus mengarahkan kamera ke target yang disediakan
2	UC-02	Mengarahkan kamera ke *props*	Pengguna mengarahkan kanera ke *props* (objek nyata) untuk memindainya

No.	Kode Kasus Penggunaan	Nama Kasus Penggunaan	Keterangan
			sehingga didapatkan *mesh* dari objek tersebut
3	UC-03	Memindai *stage*	Pengguna memindai *stage* secara keseluruhan sehingga didapatkan *primary surface* sesuai dengan keinginan pengguna
4	UC-04	Memilih membangun *terrain*	Setelah *stage* terpindai, pengguna memilih untuk melakukan pembangunan *terrain*, dengan demikian *terrain* akan terbentuk, objek yang akan dibangun pada aplikasi ini adalah sebuah rumah, batang pohon berjumlah sesuai *props*, batu, tumbuhan jalar, sebuah kotak, karakter, dan senjata.
5	UC-05	Melakukan interaksi kepada objek	Penggguna dapat mengontrol karakter yang ada, karakter akan dapat berjalan sesuai dengan lokasi *tap* pengguna pada layar *smartphone*

3.2.3.1. Mengarahkan Kamera ke Target

Spesifikasi kasus penggunaan mengarahkan kamera ke target dapat dilihat pada Tabel 3.3. Pada spesifikasi dijelaskan bagaimana pengguna memulai aplikasi ini serta menjalankan sebuah *augmented reality* sebelum fungsi Smart Terrain dijalankan.

Tabel 3.3 Spesifikasi Kasus Penggunaan Mengarahkan Kamera ke Target

Nama Kasus Penggunaan	**Mengarahkan Kamera ke Target**
Nomor	UC-01
Deskripsi	Pengguna bisa mengarahkan kamera ke mana saja, namun untuk dapat mengaktifkan fungsi Smart Terrain, pengguna harus mengarahkan kamera ke target yang disediakan
Aktor	Pengguna
Kondisi Awal	Kamera sudah menyala
Alur Normal	1. Halaman utama muncul sebagai tanda aplikasi dijalankan 2. Pengguna memilih *start* 3. Kamera akan menyala dan pengguna dapat mengarahkan kamera kemanapun 4. Pengguna mengarahkan kamera ke target 5. Setelah terarah dengan baik aplikasi akan menjalankan fungsi Smart Terrain
Alur Alternatif	-
Kondisi Akhir	Aplikasi menjalankan fungsi Smart Terrain

3.2.3.2. Mengarahkan Kamera ke *Props*

Spesifikasi kasus penggunaan mengarahkan kamera ke *props* dapat dilihat pada Tabel 3.4.

Tabel 3.4 Spesifikasi Kasus Penggunaan Mengarahkan Kamera ke *Props*

Nama Kasus Penggunaan	**Mengarahkan Kamera ke *Props***
Nomor	UC-02
Deskripsi	Pengguna mengarahkan kamera ke *props* (objek nyata) untuk memindainya sehingga didapatkan *mesh* dari objek tersebut
Aktor	Pengguna
Kondisi Awal	Target telah terdeteksi
Alur Normal	1. Target sudah dideteksi sebelumnya oleh pengguna 2. Pengguna mengarahkan kamera ke *props* (objek nyata) disekitar target 3. *Mesh* akan terbentuk sebagai hasil pindaian objek 4. Pengguna memutari objek sehingga *mesh* yang terbentuk sempurna
Alur Alternatif	-
Kondisi Akhir	*Mesh props* terbentuk dengan sempurna

3.2.3.3. Memindai *Stage*

Spesifikasi kasus penggunaan memindai *stage* dapat dilihat pada Tabel 3.5.

Tabel 3.5 Spesifikasi Kasus Penggunaan Memindai *Stage*

Nama Kasus Penggunaan	**Memindai *Stage***
Nomor	UC-03
Deskripsi	Pengguna memindai *stage* secara keseluruhan sehingga didapatkan *primary surface* sesuai dengan keinginan pengguna
Aktor	Pengguna
Kondisi Awal	Target telah terdeteksi
Alur Normal	1. Target sudah dideteksi sebelumnya oleh pengguna 2. Pengguna secara perlahan mundur menjauhi Target ketika fungsi Smart Terrain sudah berjalan 3. Akan terbentuk *primary surface* yang merupakan hasil pindaian kamera
Alur Alternatif	-
Kondisi Akhir	Terbentuk *primary surface*

3.2.3.4. Memilih Membangun Terrain

Spesifikasi kasus penggunaan memilih membangun *terrain* dapat dilihat pada Tabel 3.6.

Tabel 3.6 Spesifikasi Kasus Penggunaan Memilih Membangun Terrain

Nama Kasus Penggunaan	**Memilih Membangun Terrain**
Nomor	UC-04
Deskripsi	Setelah *stage* terpindai, pengguna memilih untuk melakukan

Nama Kasus Penggunaan	**Memilih Membangun Terrain**
	pembangunan *terrain*, dengan demikian *terrain* akan terbentuk
Aktor	Pengguna
Kondisi Awal	*Stage* dan *props* telah terpindai
Alur Normal	1. Pengguna akan melihat pilihan untuk menyelesaikan pindaian 2. Pengguna memilih pilihan "*done*" tersebut 3. Fungsi Smart Terrain akan berjalan dan melakukan *render* objek maya berdasarkan objek nyata tersebut 4. Karakter kemudian juga akan muncul setelah proses *render* selesai
Alur Alternatif	-
Kondisi Akhir	Objek maya akan muncul

3.2.3.5. Melakukan Interaksi kepada Objek

Spesifikasi kasus penggunaan melakukan interaksi kepada objek dapat dilihat pada Tabel 3.7.

Tabel 3.7 Spesifikasi Kasus Penggunaan Interaksi Kepada Objek

Nama Kasus Penggunaan	**Melakukan Interaksi Kepada Objek**
Nomor	UC-05
Deskripsi	Penggguna dapat mengontrol karakter yang ada, karakter akan dapat berjalan sesuai dengan lokasi *tap* pengguna pada layar *smartphone*

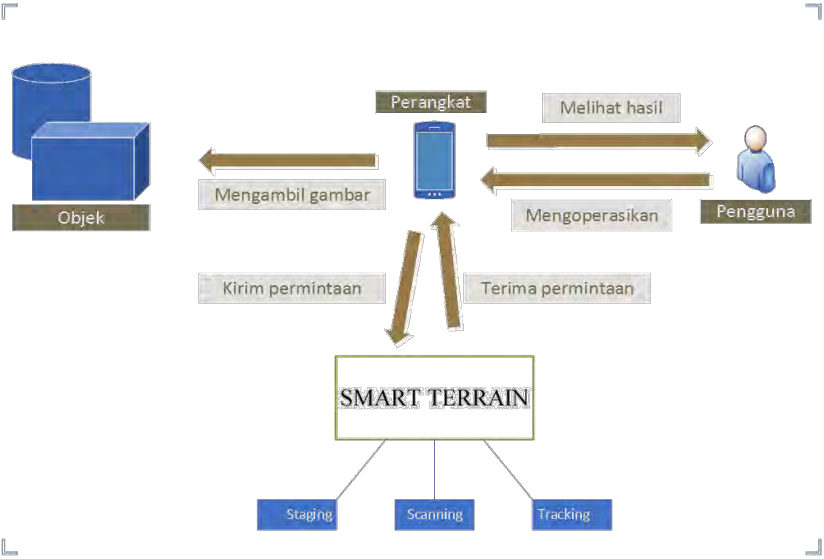
Nama Kasus Penggunaan	**Melakukan Interaksi Kepada Objek**
Aktor	Pengguna
Kondisi Awal	Karakter telah muncul
Alur Normal	1. Setelah proses *render*, karakter akan muncul 2. Pengguna dapat melakukan *tap* pada bidang manapun yang berada didalam *primary surface* 3. Karakter akan berjalan ke titik *tap* pengguna
Alur Alternatif	-
Kondisi Akhir	Karakter akan menuju lokasi *tap* pengguna

3.2.4. Arsitektur Umum Sistem

Aplikasi untuk tugas akhir ini menggunakan Unity3D dan Vuforia SDK, dan fungsi Smart Terrain terdapat pada Vuforia SDK. Arsitektur aplikasi bisa dilihat pada Gambar 3.4.

Adapun tiga fase cara kerja Smart Terrain adalah sebagai berikut:

1. Fase *staging*, dimana pengguna menggunakan kamera untuk menangkap area yang akan digunakan sebagai *terrain*, menambahkan objek-objek tersebut dan menginisialisasikannya
2. Fase *scanning*, dimana area dan objek yang ada di pengaturan ditangkap dan akan dibangun ulang oleh *tracker* dari Smart Terrain
3. Fase *tracking*, dimana *terrain* dibentuk secara *real-time* dalam bentuk *augmented reality* didalam *scene* Unity



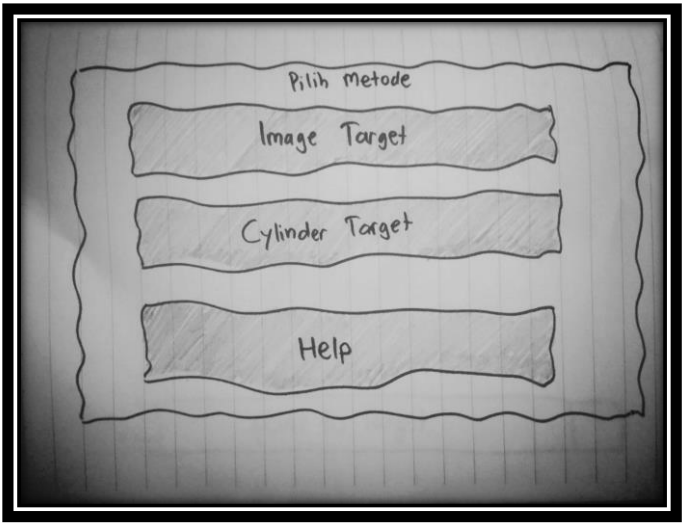
Gambar 3.4 Arsitektur Sistem

3.2.5. Rancangan Antarmuka Aplikasi

Pada subbab berikut akan membahas gambaran umum kepada pengguna bagaimana aplikasi ini berinteraksi dengan pengguna, bagaimana tampilan dari aplikasi ini agar mudah dipahami dan digunakan, sehingga akan muncul kesan *user experience* yang baik dan mudah.

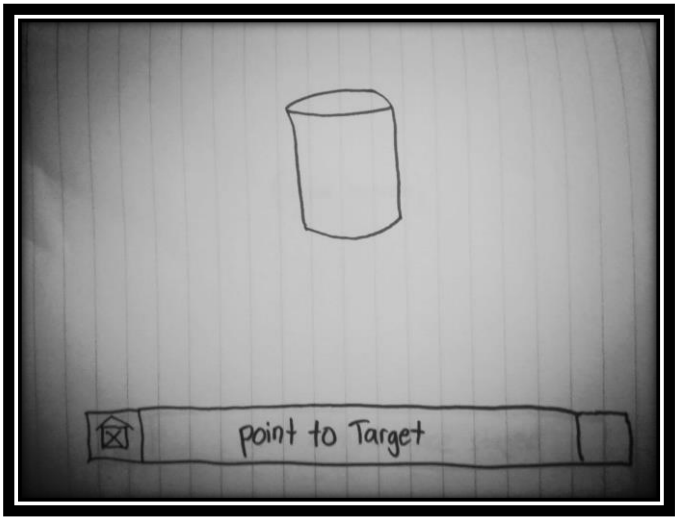
3.2.5.1. Rancangan Antarmuka Halaman Utama

Pada halaman ini aplikasi akan menampilkan halaman utama dari aplikasi, yakni menampilkan pilihan metode dan bantuan. Rancangan antarmuka halaman utama bisa dilihat pada Gambar 3.5.



Gambar 3.5 Rancangan Antarmuka Halaman Utama

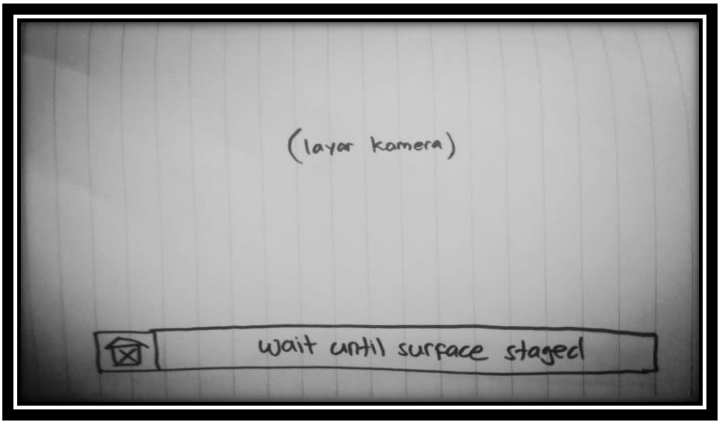
3.2.5.2. Rancangan Antarmuka Mencari Target



Gambar 3.6 Rancangan Antarmuka Mencari Target

Rancangan antarmuka ini merupakan antarmuka saat pengguna akan mengarahkan kamera ke target. Ketika Target telah terdeteksi maka antarmuka akan berubah secara otomatis. Terdapat pula tombol *home* untuk kembali ke halaman utama. Rancangan antarmuka mencari target dapat dilihat pada Gambar 3.6.

3.2.5.3. Rancangan Antarmuka Menunggu Smart Terrain



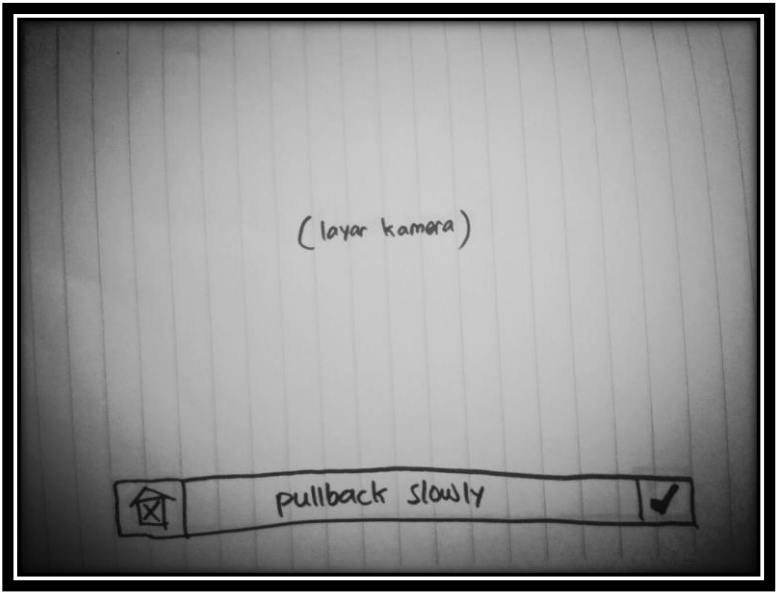
Gambar 3.7 Rancangan Antarmuka Menunggu Smart Terrain

Pada halaman ini akan menampilkan halaman singkat ketika pengguna telah berhasil mengarahkan kamera ke target dan fungsi Smart Terrain belum dijalankan. Perpindahan halaman akan dilakukan secara otomatis. Rancangan antarmuka ini dapat dilihat pada Gambar 3.7.

3.2.5.4. Rancangan Antarmuka Memindai *Stage*

Halaman ini akan ditampilkan pada saat fungsi Smart Terrain telah dijalankan, pengguna mengarahkan kamera ke *stage* dan *props* sehingga akan terbentuk *primary surface* dan *mesh props*. Terdapat pilihan *home* untuk kembali ke halaman utama

dan *done* untuk menyelesaikan pemindaian *stage*. Rancangan antarmuka halaman ini bisa dilihat pada Gambar 3.8.



Gambar 3.8 Rancangan Antarmuka Memindai *Stage*

3.2.5.5. Rancangan Antarmuka Halaman Akhir

Halaman ini adalah halaman akhir ketika proses *render*ing telah selesai dan semua objek dan karakter telah muncul. Terdapat dua pilihan yaitu *home* dan *restart* untuk mengulang proses dari awal deteksi target.

Pada antarmuka ini pengguna dapat menginteraksikan karakter dengan melakukan *tap* di layar *smartphone*. Pengguna dapat melakukan *tap* dimanapun sesuai keinginan pengguna, namun *tap* yang akan dideteksi hanya yang berada dalam jangkauan *primary surface* saja. Rancangan antarmuka ini dapat dilihat pada Gambar 3.9.

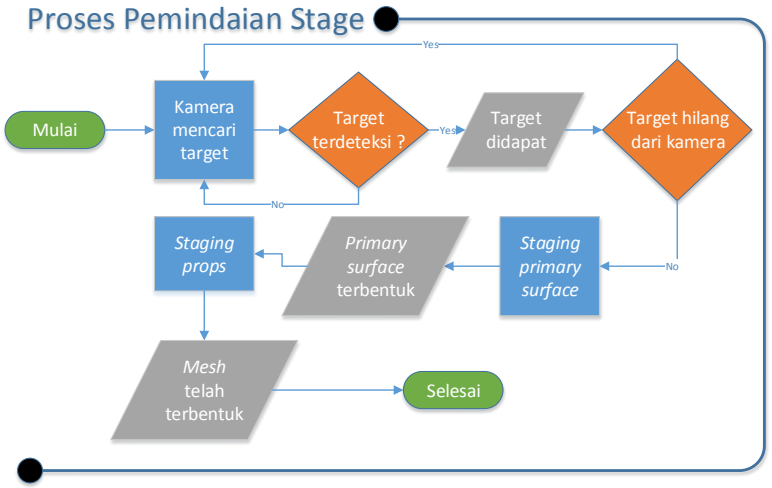
Gambar 3.9 Rancangan Antarmuka Halaman Akhir

3.2.6. Rancangan Proses Aplikasi

Pada rancangan proses aplikasi ini akan dijelaskan mengenai proses yang terjadi dalam sistem untuk memenuhi fungsionalitas yang ada pada aplikasi. Proses ini akan dijelaskan dalam bentuk alur, dan bertujuan untuk memastikan proses aplikasi berjalan dengan baik dan benar.

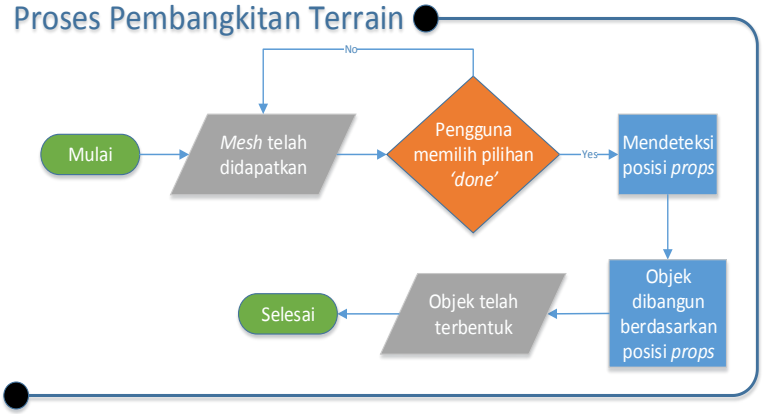
3.2.6.1. Rancangan Proses Pemindaian *Stage*

Proses ini adalah adalah salah satu bagian vital dari aplikasi. Proses dimulai ketika target telah ditemukan dan fungsi Smart Terrain pertama kali berjalan. Proses ini akan menghasilkan *primary surface* yang merupakan bidang utama dari Smart Terrain ini. Dan juga akan mendeteksi *props* yang ada pada dunia nyata dan kemudian diproyeksikan menjadi object *mesh* pada *scene*. Rancangan proses ini dapat dilihat pada Gambar 3.10.



Gambar 3.10 Rancangan Proses Pemindaian *Stage*

3.2.6.2. Rancangan Proses Pembangkitan Terrain



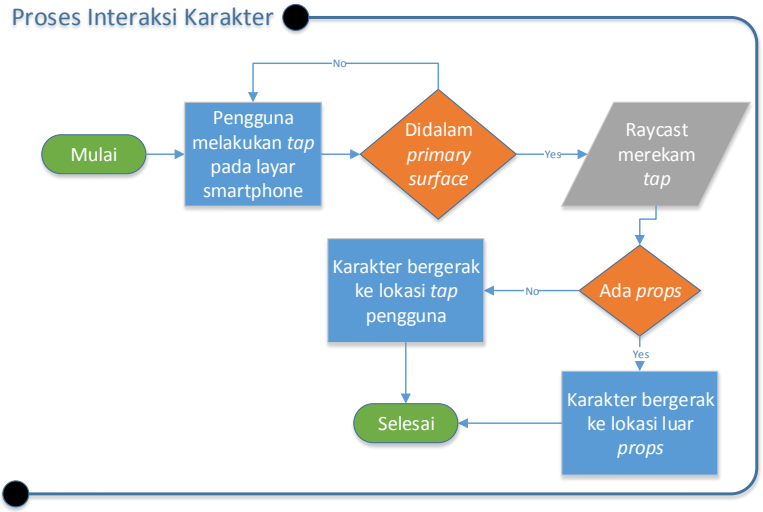
Gambar 3.11 Rancangan Proses Pembangkitan Terrain

Proses pembangkitan *terrain* akan dijalankan setelah proses *staging* selesai dan pengguna memilih pilihan *done*. Fungsi

Smart Terrain akan membangun objek maya berdasarkan *props* yang telah dideteksi sebelumnya. Rancangan proses pembangkitan *terrain* ini dapat dilihat pada Gambar 3.101.

3.2.6.3. Rancangan Proses Interaksi Karakter

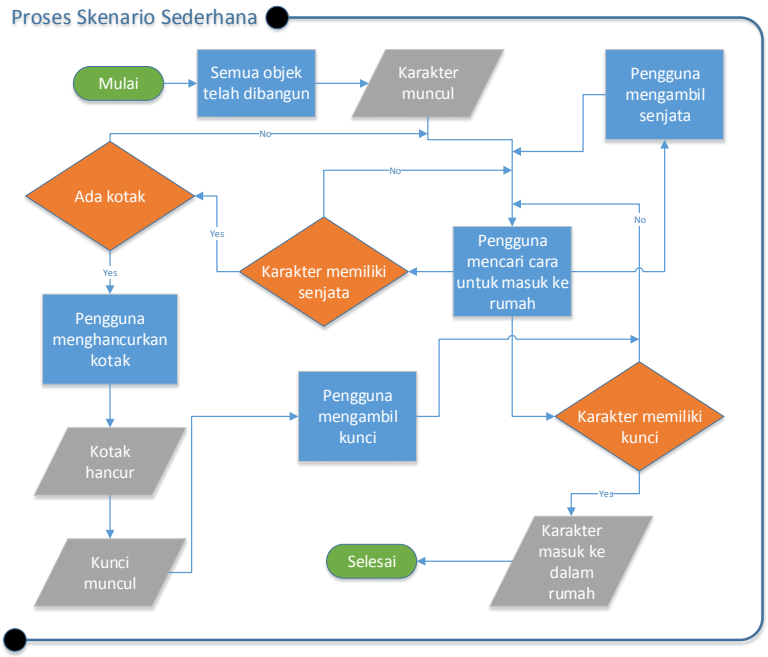
Interaksi karakter bertujuan untuk mengetahui bagaimana property dari *props* dan batasan dari *primary surface*. Karakter yang dipakai disini adalah orang. Pengguna dapat melakukan *tap* di layar *smartphone* untuk membuat karakter berjalan ke titik *tap* dari pengguna. Rancangan proses ini dapat dilihat pada Gambar 3.12. Proses ini menggunakan fungsi *raycast* untuk menangkap *tap* dari pengguna, dan *tap* yang akan ditangkap hanya yang berada pada *primary surface* saja. Jika *tap* diluar dari *primary surface* maka tidak akan ditangkap oleh fungsi *raycast*. Setelah itu, karakter akan berjalan menuju koordinat *raycast* tersebut, dan karakter juga tidak akan bisa berjalan melewati *props* yang ada.



Gambar 3.12 Rancangan Proses Interaksi Karakter

3.2.6.4. Rancangan Proses Skenario Sederhana

Rancangan skenario sederhana ini digunakan untuk menjalankan permainan sederhana yang tersedia pada aplikasi ini. Proses ini berjalan sesuai dengan apa yang dilakukan oleh pengguna. Jenis skenarionya adalah sebuah *puzzle* kecil yang harus dipecahkan oleh pengguna.



Gambar 3.13 Rancangan Proses Skenario Sederhana

Pada Gambar 3.13 digambarkan alur dari skenario sederhana pada aplikasi. Secara singkat dijelaskan bahwa pengguna harus membawa karakter untuk masuk ke dalam rumah. Untuk masuk ke dalam rumah karakter harus mempunyai kunci, sedangkan karakter pada awalnya belum mempunyai kunci. Kunci sendiri tersembunyi dibalik kotak kubus dan pengguna

harus menghancurkannya untuk mengambil isi dari kotak tersebut, namun tidak bisa dengan tangan kosong, untuk itu pengguna harus mengambil sebuah senjata yang tergeletak didekat karakter.

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Bab ini membahas tentang implementasi dari perancangan aplikasi ini. Setiap kelas pada semua modul akan dibahas di bab ini. Namun, pada hasil akhir mungkin saja terjadi perubahan kecil. Bahasa pemrograman yang digunakan adalah bahasa pemrograman C# dengan *environment* Unity3D dan tambahan Vuforia SDK.

4.1. Lingkungan Pembangunan

Dalam membangun aplikasi ini digunakan beberapa perangkat pendukung baik perangkat keras maupun perangkat lunak. Lingkungan pembangunannya akan dijelaskan sebagai berikut.

4.1.1. Lingkungan Pembangunan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan aplikasi ini adalah sebuah laptop Sony VAIO E Series 14P dan sebuah *smartphone* Asus Padfone S. Spesifikasi dari laptop adalah sebagai berikut:

- Prosesor Intel® Core™ i7-3632QM CPU @ 2.2GHz
- Memori (RAM) 8,00 GB
- Kartu grafis AMD Radeon™ HD 7670M 2GB

Sedangkan spesifikasi untuk *smartphone* adalah sebagai berikut:

- Prosesor Qualcomm® Snapdragon™ 801 @ 2.27GHz
- Memori (RAM) 2,00GB
- Kamera belakang 13 MP dengan *flash*

4.1.2. Lingkungan Pembangunan Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan untuk pengembangan aplikasi ini adalah sebagai berikut:

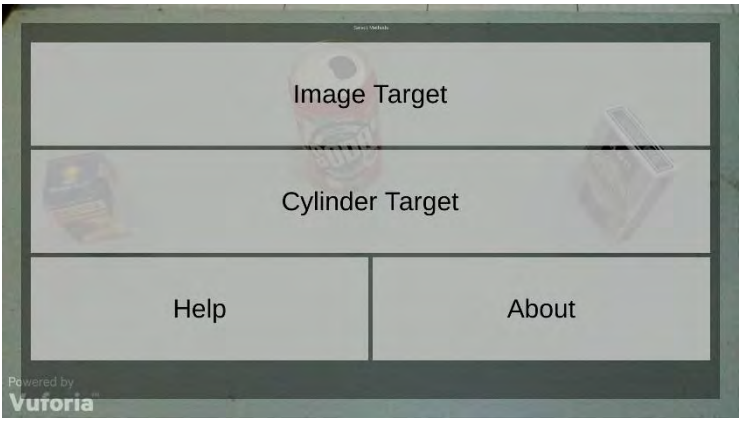
- Micros*oft* Windows 8.1 OS 64-bit
- Unity3D versi 5.0
- Vuforia SDK versi 4.0.105
- Micos*oft* Visual Studio Professional 2013
- Blender
- StarUML
- Android Kitkat 4.4 OS
- Android SDK

4.2. Implementasi Antarmuka

Pada subbab ini akan dibahas mengenai hasil implementasi dari perancangan antarmuka. Lingkungan pada antarmuka ini adalah menggunakan *smartphone* Android. Dalam implementasinya, menggunakan Unity dengan bahasa pemrograman C#.

4.2.1. Halaman Utama

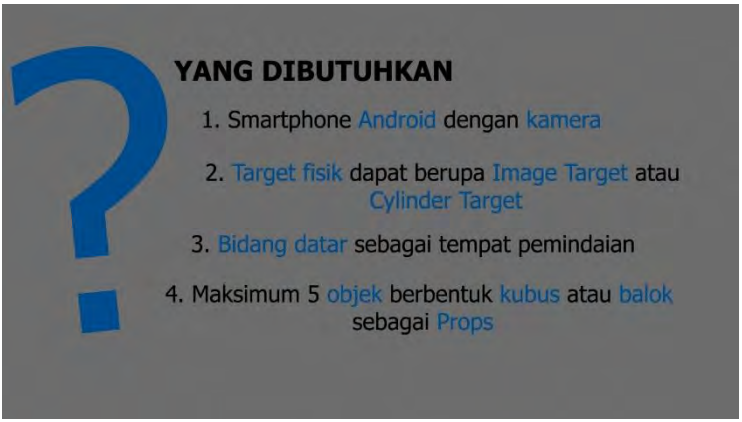
Antarmuka halaman utama adalah antarmuka yang pertama kali membutuhkan interaksi dari penggunanya saat aplikasi berjalan. Pada antarmuka ini terdapat pilihan yang dapat dipilih oleh pengguna. Pengguna juga bisa menggunakan tombol *soft key* atau *hard key* pada *smartphone* untuk keluar dari aplikasi. Pada antarmuka ini, kamera dari *smartphone* sudah dipakai sehingga latar belakang dari antarmuka adalah hasil tangkapan kamera tersebut. Detail dari antarmuka halaman utama dapat dilihat pada Gambar 4.1.



Gambar 4.1 Halaman Utama

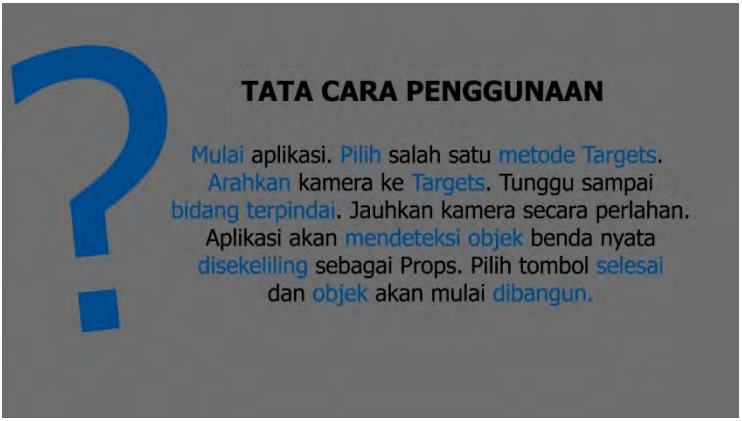
4.2.2. Halaman Bantuan

Antarmuka halaman bantuan adalah antarmuka yang dapat dipilih oleh pengguna untuk mendapatkan info mengenai aplikasi ini. Terdapat dua panduan pada antarmuka ini yaitu kebutuhan aplikasi dan tata cara penggunaan. Untuk implementasi antarmuka kebutuhan aplikasi dapat dilihat pada Gambar 4.2.



Gambar 4.2 Halaman Bantuan Kebutuhan

Sedangakan untuk implementasi antarmuka tata cara penggunaan dapat dilihat pada Gambar 4.3.



Gambar 4.3 Halaman Bantuan Tata Cara

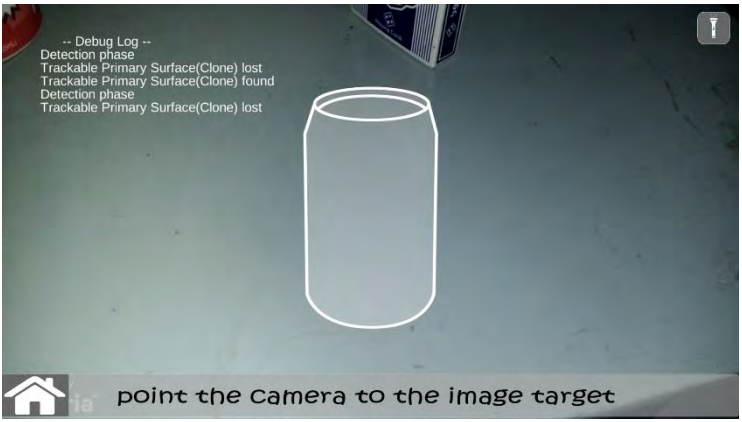
4.2.3. Halaman Profil

Halaman profil adalah sebuah media bagi penulis untuk memperkenalkan diri. Pada implementasi ini pengguna dapat melihat sekilas identitas penulis sebagai pengembang aplikasi. Detail implementasi antarmuka halaman profil dapat dilihat pada Gambar 4.4.

Gambar 4.4 Halaman Profil

4.2.4. Halaman Mencari Target

Antarmuka halaman mencari target adalah pada saat pengguna memilih salah satu metode targets pada halaman utama, baik itu *Image Target* maupun *Cylinder Target*. Pada implementasi antarmuka ini, diberikan sebuah *overlay* yang berguna sebagai panduan kepada pengguna untuk mengarahkan kamera ke target. Pada antarmuka ini dan seterusnya terdpata pilihan untuk menyalakan lampu *flash* untuk membantu proses pemindaian pada ruangan yang minim cahaya. Juga terdapat *log* yang berguna seperti *console* pada perangkat lunak pengembangan. Detail implementasi antarmuka ini dapat dilihat pada Gambar 4.5.



Gambar 4.5 Halaman Mencari Target

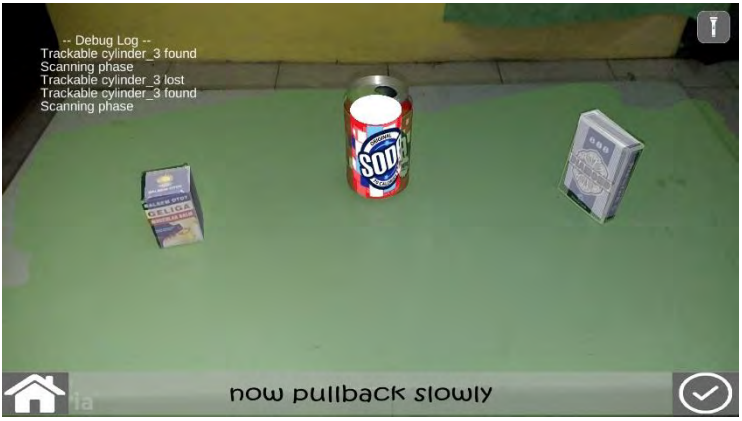
4.2.5. Halaman Menunggu Smart Terrain

Antarmuka ini merupakan sebuah transisi antara pemindaian target dengan dimulainya fungsi Smart Terrain. Pada implementasinya pengguna tidak diharuskan untuk melakukan apapun, namun dapat berpindah posisi untuk mempercepat proses Smart Terrain. Detail implementasi dapat dilihat pada Gambar 4.6.

Gambar 4.6 Halaman Menunggu Smart Terrain

4.2.6. Halaman Memindai *Stage*

Antarmuka ini dijalankan setelah fungsi Smart Terrain berjalan. Pengguna dapat berpindah-pindah posisi agar proses pemindaian *stage* menghasilkan pindaian yang bagus, termasuk *props* dan *primary surface*. Detail implementasi antarmuka halaman ini dapat dilihat pada Gambar 4.7.



Gambar 4.7 Halaman Memindai *Stage*

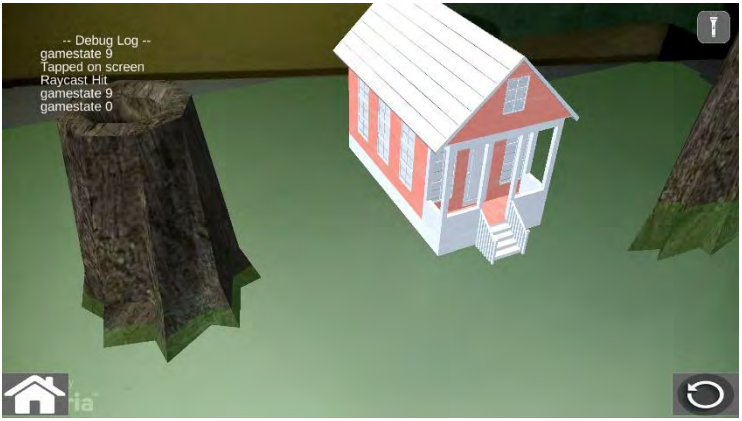
4.2.7. Halaman Permainan Sederhana

Antarmuka ini adalah implementasi dari sebuah skenario dari lingkungan hasil dari Smart Terrain ini. Setelah semua objek dibangun, pengguna dapat mengontrol sebuah karakter dengan melakukan *tap* pada layar *smartphone*. Karakter akan berjalan menuju *tap* pengguna. Pada antarmuka ini pengguna mempunyai misi untuk masuk ke dalam sebuah rumah, namun ada beberapa hal yang harus dilakukan oleh pengguna agar karakter yang dikontrol dapat masuk ke dalam rumah tersebut. Detail implementasi antarmuka ini dapat dilihat pada Gambar 4.8.

Gambar 4.8 Halaman Permainan Sederhana

4.2.8. Halaman Akhir

Antarmuka halaman akhir merupakan antarmuka setelah pengguna berhasil masuk ke dalam rumah pada antarmuka sebelumnya. Pada implementasi pada halaman akhir, karakter sudah tidak dapat dikontrol lagi, yang ada hanyalah sebuah lingkungan 3D yang masih dapat dilihat oleh pengguna. Detail implementasinya dapat dilihat pada Gambar 4.9.



Gambar 4.9 Halaman Akhir

4.3. Implementasi Aplikasi

Pada subbab ini akan dibahas mengenai implementasi aplikasi dari kasus penggunaan ke dalam baris kode. Dijelaskan pula kode-kode penunjang agar aplikasi ini berjalan dengan sebagaimana mestinya. Implementasi ini menggunakan Unity dengan bahasa pemrograman C#.

4.3.1. Insialisasi Smart Terrain

Aplikasi ini menggunakan teknologi *augmented reality* pada dasarnya dan diteruskan dengan Smart Terrain. Smart Terrain sendiri adalah sebuah fitur dari Vuforia SDK yang bisa diaktifkan ketika inisialisasi *augmented reality* sudah dilakukan.

Pada Gambar 4.10 Dapat dilihat bahwa inisialisasi Smart Terrain hanya dilakukan dengan mencentang pilihan Smart Terrain pada *Target Behaviour*. Kemudian akan terbentuk Smart Terrain sederhana dengan menggunakan kode implementasi bawaan dari Vuforia.

Gambar 4.10 Inisialisasi Smart Terrain

Sebuah Smart Terrain diatur oleh sebuah *event handler*. Pada aplikasi ini juga menggunakan sebuah *event handler* dengan beberapa modifikasi sehingga mempunyai perilaku berbeda dengan Smart Terrain sebelumnya. Pada Gambar 4.11 Merupakan inspeksi dari kelas Smart Terrain.

Pada fungsi awal Smart Terrain, akan didaftarkan fungsi-fungsi penanganan untuk inisialisasi, pembangkitan *props*, dan pembangkitan *surface*, seperti pada Kode Sumber 4.1

Gambar 4.11 Kelas Smart Terrain

```
void Start()
{
debugLog                                                    =
GameObject.FindObjectOfType(typeof(DebugLogController))
as DebugLogController;

mReconstructionBehaviour                                    =
GetComponent<ReconstructionBehaviour>();
if (mReconstructionBehaviour)
{

mReconstructionBehaviour.RegisterInitializedCallback(On
Initialized);

mReconstructionBehaviour.RegisterPropCreatedCallback(On
PropCreated);

mReconstructionBehaviour.RegisterSurfaceCreatedCallback
(OnSurfaceCreated);
}
kastil.SetActive(false);
debugLog.InsertLog("Smart Terrain started");
}
```

Kode Sumber 4.1 Kode Inisialisasi Smart Terrain

4.3.2. Implementasi Kelas GUI

Aplikasi ini menggunakan *behaviour* yang berbeda dibandingkan dengan Smart Terrain bawaan dari Vuforia. Smart Terrain bawaan Vuforia menggunakan cara otomatis ketika pendeteksian *props* maka akan membentuk objek secara langsung. Sedangkan pada aplikasi ini akan dilakukan secara manual, hal itu dimaksudkan agar aplikasi dapat memindai *props* secara sempurna sebelum pembangkitan objek *props* dilakukan. Untuk itu digunakan beberapa fase yang membedakan perlakuan aplikasi ini sekaligus penanganan UI juga.

4.3.2.1. Fase Deteksi

Fase ini adalah fase dimana aplikasi memasuki tahap pencarian target. Target bisa berupa *Image Target* maupun *Cylinder Target*. Pada kode Kode Sumber 4.2 dapat dilihat bahwa fungsi ini merupakan bagian dari fungsi OnGUI yang menggunakan switch sehingga bisa dipindah-pindah fasenya dengan mudah. Pada fase ini pembangkit dari Smart Terrain dimatikan. Ketika target terdeteksi akan langsung berpindah ke fase berikutnya.

```
// detection phase - when you search for target
case GUIStates.OVERLAY_OUTLINE:
debugLog.InsertLog("Detection phase");

uiInput.UpdateTitle(HEADER_MESSAGE.POINT_DEVICE);

smartSurface.GetComponent<Renderer>().enabled     =
false;

uiInput.DrawCylinderTargetOutline();
uiInput.DrawBackButton();

uiInput.DrawFlashButton(cameraState);
if
(cylinderTrackableEventHandler.cylinderDetected)
{
   state = GUIStates.WAITING;
}
break;
```

Kode Sumber 4.2 Kode Fase Deteksi

4.3.2.2. Fase Menunggu

Fase menunggu adalah saat target sudah terdeteksi namun fungsi pemindaian Smart Terrain belum berjalan. Waktu yang dibutuhkan untuk memulai pemindaian tergantung kondisi dari *stage* dan cahaya. Detailnya dapat dilihat pada Kode Sumber 4.3.

```
// waiting phase - when target found but smart
terrain no active yet
case                                GUIStates.WAITING:
debugLog.InsertLog("Waiting phase");

uiInput.UpdateTitle(HEADER_MESSAGE.WAIT);
uiInput.DrawBackButton();

uiInput.DrawFlashButton(cameraState);

smartSurface.GetComponent<Renderer>().enabled     =
false;

if (primarySurfaceStagged)
{
   state = GUIStates.SCANNING;
}
else                                                if
(cylinderTrackableEventHandler.cylinderDetected  ==
false)
{
   state = GUIStates.OVERLAY_OUTLINE;
}
break;
```

Kode Sumber 4.3 Kode Fase Menunggu

4.3.2.3. Fase Pemindaian

Pada fase ini, Smart Terrain sudah memulai melakukan pemindaian *stage*. Fase ini akan menghasilkan daftar *props* yang ditemukan serta *primary surface* dari bidang datar.

```
// scanning phase - when smart terrain on and
surface tracking
case GUIStates.SCANNING:
debugLog.InsertLog("Scanning phase");

uiInput.UpdateTitle(HEADER_MESSAGE.PULLBACK_SLOWLY);

smartSurface.GetComponent<Renderer>().enabled      =
```

```
primarySurfaceStagged;
uiInput.DrawBackButton();
uiInput.DrawDoneButton();

uiInput.DrawFlashButton(cameraState);
break;
```

Kode Sumber 4.4 Kode Fase Pemindaian

Pada fase ini, pembangkit akan diaktifkan hanya untuk *primary surface*, sehingga dapat terlihat perbedaan antara bidang yang sudah terpindai dan belum terpindai. Detailnya dapat dilihat pada Kode Sumber 4.4.

4.3.2.4. Fase Pembangkitan

Fase ini merupakan tahap akhir dari Smart Terrain dimana *props* yang ditemukan pada tahap sebelumnya akan dibangkitkan menjadi sebuah objek. Pada tahap ini juga fungsi untuk pemindaian dinonaktifkan sehingga tidak akan melakukan pemindaian lagi. Objek yang dibangun akan menggunakan sifat dari *props* yang dideteksi. Objek yang dibangun pada aplikasi ini adalah sebuah rumah pada posisi target, batang pohon yang terletak pada posisi *props* dan sesuai ukuran *props* diikuti juga dengan objek batu dan tumbuhan jalar, sebuah kotak kubus yang terletak pada posisi *props* pertama, dan seorang karakter disamping rumah diikuti dengan sebuah senjata yang tergeletak didepan rumah. Detailnya dapat dilihat pada Kode Sumber 4.5.

```
// rendering phase - user tap done button and props
will rendered
case GUIStates.RENDERING:
debugLog.InsertLog("Rendering phase");
if     ((reconstructionBehaviour     !=     null)     &&
(reconstructionBehaviour.Reconstruction != null))
{

cylinderSmartTerrainEventHandler.ShowPropClones();
```

```
cylinderSmartTerrainEventHandler.kunci.gameObject.SetAc
tive(false);

reconstructionBehaviour.Reconstruction.Stop();
state = GUIStates.PLAY;
}
break;
```

Kode Sumber 4.5 Kode Fase Pembangkitan

4.3.2.5. Fase Permainan Sederhana

Setelah pembangkitan selesai dilakukan, sebuah karakter akan muncul dan dapat dikendalikan oleh pengguna. Karakter ini berbentuk orang yang dapat berjalan sesuai lokasi *tap* pengguna pada layar *smartphone*. Permainan sederhana ini merupakan sebuah skenario yang harus dimainkan oleh pengguna. Skenarionya permainan dijelaskan sebagai alur pada Gambar 3.13. Untuk kode sumbernya dapat dilihat pada Kode Sumber 4.6.

```
// user interaction phase - character will appear
case GUIStates.PLAY:
if (spawn)
{                    person.gameObject.SetActive(true);
}

debugLog.InsertLog("gamestate " + gameState);
if(gameState == 0)
{
   uiInput.UpdateTitle(HEADER_MESSAGE.DEFAULT);
}
else if(gameState == 1)
{
   uiInput.UpdateGameTitle(GAME_MESSAGE.FIND_WAY);
}
else if(gameState == 2)
{
uiInput.UpdateGameTitle(GAME_MESSAGE.TAP_FRONTDOOR);
}
else if(gameState == 3)
{
```

```
   uiInput.UpdateGameTitle(GAME_MESSAGE.TAP_BOX);
}
else if (gameState == 4)
{
uiInput.UpdateGameTitle(GAME_MESSAGE.WEAPON_RADIUS);
   uiInput.DrawPickButton("weapon");
}
else if (gameState == 5)
{
uiInput.UpdateGameTitle(GAME_MESSAGE.WEAPON_PICKED);
}
else if (gameState == 6)
{
 uiInput.UpdateGameTitle(GAME_MESSAGE.BOX_RADIUS);
 uiInput.DrawShootButton();
}
else if (gameState == 7)
{
uiInput.UpdateGameTitle(GAME_MESSAGE.BOX_DESTROYED);
}
else if (gameState == 8)
{
uiInput.UpdateGameTitle(GAME_MESSAGE.KEY_RADIUS);
uiInput.DrawPickButton("key");
}
else if (gameState == 9)
{                     uiInput.UpdateGameTitle(GAME_MESSA
GE.TAP_FRONTDOOR_KEY);
                    uiInput.DrawEnterHouseButton();
}
uiInput.DrawBackButton();
uiInput.DrawResetButton();
                uiInput.DrawFlashButton(cameraState);
break;
```

Kode Sumber 4.6 Kode Fase Permainan Sederhana

4.3.2.6. Fungsi Tambahan Lampu *Flash*

Pendeteksian *props* dan *primary surface* sangat bergantung pada kondisi *stage* dan cahaya, maka dari itu dibuatlah sebuah fungsi tambahan untuk menyalakan lampu *flash*

dari *smartphone*. Fungsi ini menggunakan perlakuan kamera secara langsung untuk mengaktifkan lampu *flash* kamera. Detailnya dapat dilihat pada Kode Sumber 4.7.

```
void uiInput_TappedFlashOnButton()
{         CameraDevice.Instance.SetFlashTorchMode(false)
;
cameraState = false;
debugLog.InsertLog("Flash turned off");
}

void uiInput_TappedFlashOffButton()
{
CameraDevice.Instance.SetFlashTorchMode(true);
cameraState = true;
debugLog.InsertLog("Flash turned on");
}
```

Kode Sumber 4.7 Kode Lampu *Flash*

4.3.3. Modifikasi Fungsi Pembangkitan *Props*

Pembangkitan *props* pada Smart Terrain bawaan Vuforia menggunakan cara otomatis, sehingga ketika *props* dideteksi pertama kali, langsung terbentuk objek. Hal itu tentu tidak menghasilkan kualitas objek yang bagus karena *props* tidak terpindai dengan sempurna. Untuk itu pada aplikasi ini dilakukan modifikasi agar tidak menggunakan cara otomatis.

Pada fase pembangkitan terdapat fungsi bernama *ShowPropClones* yang merupakan tahap pembangkitan objek berdasarkan *props* yang telah dideteksi. Fungsi ini juga merupakan kunci utama dari aplikasi ini. Objek akan dideteksi berdasarkan posisi dan ukuran dari *props*. Akan dibangkitkan juga beberapa objek tambahan untuk permainan sederhana pada fungsi ini. Detailnya dapat dilihat pada Kode Sumber 4.8.

```
public void ShowPropClones()
{
if (!propsCloned)
```

```
{
PropAbstractBehaviour[]              props              =
GameObject.FindObjectsoftype(typeof(PropAbstractBehavio
ur)) as PropAbstractBehaviour[];
foreach (PropAbstractBehaviour prop in props)
{
Transform                   BoundingBox                 =
prop.transform.FindChild("BoundingBoxCollider");
BoxCollider                    collider                 =
BoundingBox.GetComponent<BoxCollider>();
collider.isTrigger = false;
                prop.SetAutomaticUpdatesDisabled(true);
Renderer                    propRenderer                =
prop.GetComponent<MeshRenderer>();
if (propRenderer != null)
{
Destroy(propRenderer);
}

Hills spawn = Instantiate(HillsPrefab) as Hills;
spawn.name = "Hills";
spawn.transform.localPosition           =           new
Vector3(prop.transform.localPosition.x,
prop.transform.localPosition.y          +          260,
prop.transform.localPosition.z);
spawn.transform.localScale                          =
prop.Prop.BoundingBox.HalfExtents;
spawn.transform.localRotation = Quaternion.identity;

if(keyCount == 0)
{
kotak = Instantiate(crate) as GameObject;
kotak.name = "Kotak";
kotak.transform.localPosition           =           new
Vector3(prop.transform.localPosition.x,
prop.transform.localPosition.y,
prop.transform.localPosition.z - 300);

boom = Instantiate(explode) as GameObject;
boom.name = "Explosion";
```

```
boom.transform.localPosition                                    =
kotak.transform.localPosition;

kunci = Instantiate(key) as GameObject;
kunci.name = "Kunci";
kunci.transform.localPosition                                   =
kotak.transform.localPosition;
keyCount++;
}
}

kastil.SetActive(true);
weapon.SetActive(true);
propsCloned = true;
debugLog.InsertLog(props.Length + " props cloned");
}
}
```

Kode Sumber 4.8 Fungsi *Show Props*

4.3.4. Implementasi Kelas Karakter

Karakter pada aplikasi ini akan memainkan peran sebagai tokoh yang dikendalikan oleh pengguna. Dengan menggunakan *raycast*, pengguna dapat menggerakkan karakter sesuai lokasi *tap* pada *smartphone*.

4.3.4.1. Penggunaan *Raycast*

Raycast adalah kelas dari Unity untuk mengambil lokasi *tap* pada layar *smartphone* untuk dilakukan beberapa perlakuan. Pada aplikasi ini *raycast* akan langsung mendapatkan koordinat dari *tap* pada *primary surface*. Pada tahap ini juga *raycast* berfungsi sebagai pendukung fase permainan sederhana. Terdapat sedikit modifikasi pada fungsi *raycast* dari *raycast* pada umumnya karena kamera pada aplikasi ini menggunakan ARCamera. Detailnya dapat dilihat pada Kode Sumber 4.9.

```
private void HandleSingleTap()
{
GameObject                  gameobject                  =
```

```
QCARManager.Instance.ARCameraTransform.gameObject;
Camera[]                    camera                    =
gameobject.GetComponentsInChildren<Camera>();
Ray                         ray                       =
camera[0].ScreenPointToRay(Input.mousePosition);
RaycastHit hitInfo;
if(Physics.Raycast(ray, out hitInfo))
{
if            (hitInfo.collider.CompareTag("House")
&& !gui.gotKey)
{
debugLog.InsertLog("house tapped!");
lookRotationDirection     =     hitInfo.point     -
transform.position;
RotateUpdate();
gui.gameState = 2;
}

// front door, have key
else  if  (hitInfo.collider.CompareTag("House")  &&
gui.gotKey)
{
lookRotationDirection     =     hitInfo.point     -
transform.position;
RotateUpdate();
gui.gameState = 9;
}
// in shoot radius, no weapon

else    if    (hitInfo.collider.CompareTag("Crate")
&& !weaponOnHand.activeSelf)
{
debugLog.InsertLog("in crate radius");
lookRotationDirection     =     hitInfo.point     -
transform.position;
RotateUpdate();
gui.gameState = 3;
}

// in box radius, can do shoot
```

```
else if (hitInfo.collider.CompareTag("Crate") &&
weaponOnHand.activeSelf)
{
debugLog.InsertLog("in shoot radius");
lookRotationDirection = hitInfo.point -
transform.position;
RotateUpdate();
gui.gameState = 6;
}

else if (hitInfo.collider.CompareTag("Weapon"))
{
debugLog.InsertLog("in weapon radius");
lookRotationDirection = hitInfo.point -
transform.position;
RotateUpdate();
gui.gameState = 4;
}

// in key radius
else if (hitInfo.collider.CompareTag("Key")
&& !gui.gotKey)
{
debugLog.InsertLog("in key radius");
lookRotationDirection = hitInfo.point -
transform.position;
RotateUpdate();
gui.gameState = 8;
}

else
{
movementTarget = hitInfo.point;
lookRotationDirection = hitInfo.point -
transform.position;
}
Debug.Log("hit!");
debugLog.InsertLog("Raycast Hit");
}
if(personAppear)
```

```
{
interactive = true;
}
}
```

Kode Sumber 4.9 Fungsi *Raycast*

4.3.4.2. Pergerakan Karakter

Karakter pada aplikasi ini dapat dikendalikan oleh pengguna, maka dari itu karakter dapat berjalan dan menghadap ke tempat tujuan. Untuk pergerakan karakter pada aplikasi ini digunakan *CharacterController* untuk memudahkan pergerakan. Detailnya dapat dilihat pada Kode Sumber 4.10.

```
private void Move()
{
Vector2 currentVectorToTarget =
 (new Vector2(movementTarget.x, movementTarget.z) - new
Vector2(transform.position.x, transform.position.z));

if(!characterMoving && currentVectorToTarget.magnitude
>= MIN_DISTANCE)
{
characterMoving = true;
animator.SetBool("run", true);
animator.SetBool("aim", false);
}

if(characterMoving && currentVectorToTarget.magnitude <
MIN_DISTANCE)
{
characterMoving = false;
if (weaponOnHand.activeSelf)
{
animator.SetBool("run", false);
animator.SetBool("aim", true);
}
else
{
animator.SetBool("run", false);
```

```
animator.SetBool("aim", false);
}
}

if(characterMoving)
{
Vector2 movementVector =
currentVectorToTarget.normalized * movementSpeed *
Time.deltaTime;
Vector3 direction = new Vector3(movementVector.x, 0,
movementVector.y);
Debug.Log(direction);
characterController.Move(direction);
RotateUpdate();
}
}

private void RotateUpdate()
{
rotation =
Quaternion.LookRotation(lookRotationDirection.normalize
d);
rotation = Quaternion.Euler(0, rotation.eulerAngles.y,
0);
transform.rotation =
Quaternion.Slerp(transform.rotation, rotation,
Time.deltaTime * 8);
}
```

Kode Sumber 4.10 Fungsi Pergerakan Karakter [8]

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada aplikasi yang dikembangkan. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsional secara keseluruhan. Ada beberapa skenario yang dilakukan untuk pengujian ini. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini.

5.1. Lingkungan Pembangunan

Perangkat keras yang digunakan dalam pembangunan aplikasi ini adalah sebuah laptop Sony VAIO E Series 14P dan sebuah *smartphone* Asus Padfone S. Spesifikasi dari laptop adalah sebagai berikut:

- Prosesor Intel® Core™ i7-3632QM CPU @ 2.2GHz
- Memori (RAM) 8,00 GB
- Kartu grafis AMD Radeon™ HD 7670M 2GB

Sedangkan spesifikasi untuk *smartphone* adalah sebagai berikut:

- Prosesor Qualcomm® Snapdragon™ 801 @ 2.27GHz
- Memori (RAM) 2,00GB
- Kamera belakang 13 MP dengan *flash*

5.2. Skenario Pengujian

Skenario pengujian yang dilakukan adalah sebagai berikut:

1. Pengujian skenario 1. Berupa pengujian terhadap deteksi dasar AR dengan menggunakan *Cylinder Target*
2. Pengujian skenario 2. Berupa pengujian terhadap deteksi *props* benda polos

3. Pengujian skenario 3. Berupa pengujian terhadap performa aplikasi saat mendeteksi *props* sebanyak satu buah
4. Pengujian skenario 4. Berupa pengujian terhadap performa aplikasi saat mendeteksi *props* sebanyak lima buah
5. Pengujian skenario 5. Berupa pengujian terhadap performa apllikasi saat mendeteksi *props* sebanyak enam buah
6. Pengujian skenario 6. Berupa pengujian terhadap kesesuaian lokasi dan ukuran objek yang dibangun dengan *props*
7. Pengujian skenario 7. Berupa pengujian terhadap permainan sederhana, yaitu gerakan dan rotasi karakter berdasarkan *tap* pengguna
8. Pengujian skenario 8. Berupa pengujian terhadap permainan sederhana, yaitu *collider props* dan perlakuannya terhadap karakter
9. Pengujian skenario 9. Berupa pengujian oleh pengguna langsung

5.2.1. Pengujian Skenario 1 dan Evaluasi

Pada pengujian skenario 1 ini akan dilakukan pengujian terhadap pendeteksian target sebagai fungsi dasar AR. Keberhasilan deteksi ini sangat penting karena dasar AR inilah yang akan menjadikan Smart Terrain berfungsi. Tabel skenario pengujian 1 dapat dilihat pada Tabel 5.1.

Tabel 5.1 Skenario Pengujian 1

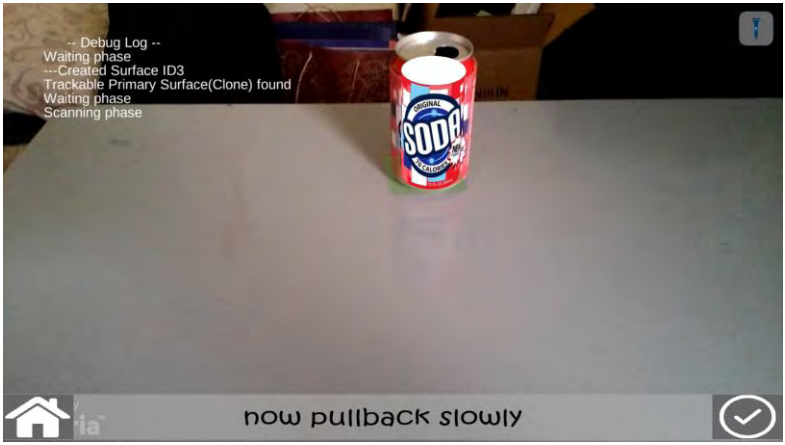
Nama Skenario Pengujian	**Pengujian Deteksi Dasar AR**
Kode	SP-01
Tujuan Pengujian	Mendeteksi apakah target dapat dideteksi dengan mudah
Kondisi Awal	Pengguna memilih menu *Cylinder*

Nama Skenario Pengujian	**Pengujian Deteksi Dasar AR**
	Target pada halaman utama
Data *Input*	-
Prosedur Pengujian	1. Kamera di depan target 2. Kamera di atas t*arget* dengan sudut 45 derajat
Hasil yang Diharapkan	Target dapat dideteksi dalam waktu 5 detik
Hasil yang Diperoleh	Target berhasil dideteksi kurang dari 5 detik

Pada awal pengujian, arahkan kamera langsung ke target. Target akan langsung dideteksi dengan cepat oleh aplikasi. Hal ini merupakan hal yang sangat bisa terjadi dalam keadaan *stage* yang bagus dan pencahayaan yang cukup. Jika pendeteksian dirasa lambat maka dapat mengaktifkan lampu *flash* dari *smartphone*. Pendeteksian didepan target dapat dilihat pada Gambar 5.1.

Gambar 5.1 Pengujian Target Depan

Karena bentuk dari target adalah silinder, maka pengambilan kamera harus dilakukan dari depan Target, jika dari samping dan belakang tidak akan bisa terdeteksi. Dan sudut yang paling bagus untuk pengambilan dari depan adalah 45 derajat. Pengambilan pada sudut 45 derajat dapat dilihat pada Gambar 5.2.



Gambar 5.2 Pengujian Target Atas 45

5.2.2. Pengujian Skenario 2 dan Evaluasi

Pada pengujian skenario 2 ini akan dilakukan pengujian terhadap pendeteksian *prop* dengan menggunakan objek sebagian besar berwarna putih atau polos. Skenario pengujian 2 dapat dilihat pada Tabel 5.2.

Tabel 5.2 Skenario Pengujian 2

Nama Skenario Pengujian	**Pengujian Pendeteksian *Props* Polos**
Kode	SP-02
Tujuan Pengujian	Membuktikan bahwa *props* yang bisa dipakai adalah yang memiliki tekstur yang cukup unik

Nama Skenario Pengujian	**Pengujian Pendeteksian *Props* Polos**
Kondisi Awal	Target dan *props* sudah ditempatkan pada *stage*
Data *Input*	-
Prosedur Pengujian	Pengguna mengarahkan kamera ke *stage* untuk melakukan pemindaian
Hasil yang Diharapkan	Objek polos tidak akan bisa terpindai dengan sempurna
Hasil yang Diperoleh	*Props* dengan warna polos tidak bisa dideteksi dengan sempurna

Sesuai dengan kemampuan Smart Terrain ini, benda yang bisa dipindai adalah yang mempunyai tekstur yang cukup unik. Benda polos atau putih atau hitam penuh tidak akan bisa terpindai dengan sempurna. Pada pengujian ini terdapat tiga *props* yang ada dalam *stage*, dua *props* yang mempunyai tekstur unik bisa dipindai dengan baik, sedangkan *props* dengan warna polos tidak bisa terpindai dengan baik. Hasil pengujian dapat dilihat pada Gambar 5.3.

Gambar 5.3 *Props* Polos

5.2.3. Pengujian Skenario 3

Pada skenario 3 ini akan dilakukan pengujian terhadap pemindaian *props* sebanyak satu buah benda. Hal ini bertujuan untuk melihat performa aplikasi saat melakukan pemindaian *props*. Dengan hanya satu buah benda, seharusnya aplikasi dapat melakukan pemindaian tanpa adanya penurunan performa. Untuk dapat mengukur performa, digunakan satuan pengukuran FPS (*frames per second*) dimana standar yang baik adalah 30 FPS. Untuk membantu pengukuran, maka digunakan aplikasi bernama GameBench pada perangkat Android, aplikasi ini dapat mengukur FPS secara langsung dari aplikasi yang dijalankan. Skenario pengujian 3 dapat dilihat pada Tabel 5.3.

Tabel 5.3 Skenario Pengujian 3

Nama Skenario Pengujian	**Pengujian Pendeteksian Sebanyak Satu Buah *Props***
Kode	SP-03
Tujuan Pengujian	Mendeteksi apakah aplikasi mampu memindai objek dengan baik
Kondisi Awal	Target dan sebuah *props* sudah ditempatkan pada *stage*
Data *Input*	-
Prosedur Pengujian	Pengguna mengarahkan kamera ke *stage* untuk melakukan pemidaian
Hasil yang Diharapkan	Aplikasi dapat memindai *props* dengan baik
Hasil yang Diperoleh	Aplikasi berhasil memindai *props* dengan performa yang baik

Tabel 5.4 Percobaan Satu *Props*

Percobaan ke-	**Hasil FPS yang Didapatkan**
1	60
2	53

Percobaan ke-	Hasil FPS yang Didapatkan
3	55
4	47
5	61

Pada pengujian ketiga ini, dilakukan beberapa kali percobaan untuk dapat mengukur nilai FPS dari aplikasi. Tabel percobaan pengukuran aplikasi dapat dilihat pada Tabel 5.4.

Seperti yang diharapkan, aplikasi ini dapat memindai sebuah benda dengan baik dan performa yang baik. Berdasarkan percobaan, FPS yang didapatkan adalah dengan rata-rata 55.2. Proses percobaan pengujian skenario 3 ini dapat dilihat pada Gambar 5.4.

Gambar 5.4 Satu *Props*

5.2.4. Pengujian Skenario 4

Pada pengujian skenario 4 ini akan dilakukan pengujian terhadap pemindaian *props* sebanyak lima buah benda. Untuk dapat mengukur performa, digunakan satuan pengukuran FPS (*frames per second*) dimana standar yang baik adalah 30 FPS.

Untuk membantu pengukuran, maka digunakan aplikasi bernama GameBench pada perangkat Android, aplikasi ini dapat mengukur FPS secara langsung dari aplikasi yang dijalankan. Skenario pengujian 4 dapat dilihat pada Tabel 5.5.

Tabel 5.5 Skenario Pengujian 4

Nama Skenario Pengujian	**Pengujian Pendeteksian Sebanyak Lima Buah *Props***
Kode	SP-04
Tujuan Pengujian	Mendeteksi apakah aplikasi mampu memindai objek dengan jumlah banyak
Kondisi Awal	Target dan lima buah *props* sudah ditempatkan pada *stage*
Data *Input*	-
Prosedur Pengujian	Pengguna mengarahkan kamera ke *stage* untuk melakukan pemidaian
Hasil yang Diharapkan	Aplikasi dapat memindai kelima *props* dengan baik namun dengan penurunan performa
Hasil yang Diperoleh	Aplikasi berhasil memindai kelima *props* namun dengan performa yang menurun, rata-rata FPS yang tercatat adalah 4.8

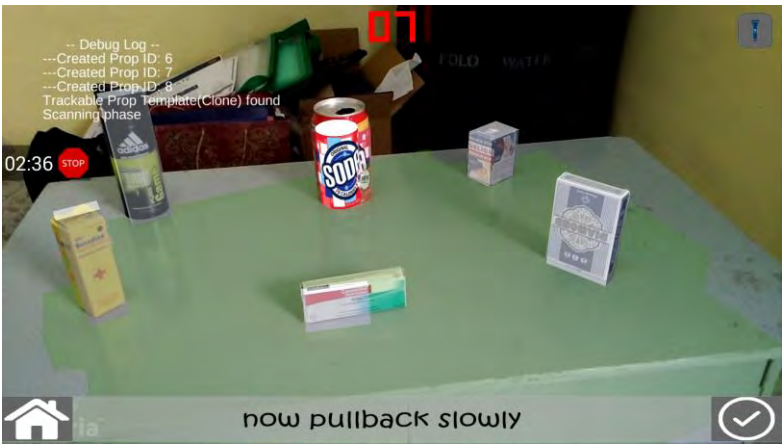
Pada pengujian skenario 4 ini, dilakukan beberapa kali pengujian untuk mengukur FPS pada aplikasi. Tabel percobaan dapat dilihat pada Tabel 5.6.

Tabel 5.6 Percobaan Lima *Props*

Percobaan ke-	**Hasil FPS yang Didapatkan**
1	7
2	6
3	4

4	6
5	1

Berdasarkan beberapa pengujian didapatkan hasil dengan FPS yang rendah, dengan nilai rata-rata 4.8. Dengan hasil ini, aplikasi tidak bisa digunakan dengan lancar walaupun kelima *props* telah berhasil dipindai.



Gambar 5.5 Lima *Props*

Dari pengujian yang dilakukan, aplikasi telah berhasil mendeteksi lima buah *props* pada *stage*. Namun, performa aplikasi sangat menurun, bahkan dengan *smartphone* berprosesor tinggi. Hasil pengujian dapat dilihat pada Gambar 5.5.

5.2.5. Pengujian Skenario 5

Pada pengujian skenario 5 ini akan dilakukan pengujian terhadap performa aplikasi saat melakukan pemindaian sebanyak enam buah benda nyata. Untuk dapat mengukur performa, digunakan satuan pengukuran FPS (*frames per second*) dimana standar yang baik adalah 30 FPS. Untuk membantu pengukuran, maka digunakan aplikasi bernama GameBench pada perangkat

Android, aplikasi ini dapat mengukur FPS secara langsung dari aplikasi yang dijalankan. Skenario pengujian 5 dapat dilihat pada Tabel 5.7.

Tabel 5.7 Skenario Pengujian 5

Nama Skenario Pengujian	**Pengujian Pendeteksian Sebanyak Enam Buah *Props***
Kode	SP-05
Tujuan Pengujian	Mendeteksi apakah aplikasi mampu memindai objek dengan jumlah lebih dari yang disarankan
Kondisi Awal	Target dan enam buah *props* sudah ditempatkan pada *stage*
Data *Input*	-
Prosedur Pengujian	Pengguna mengarahkan kamera ke *stage* untuk melakukan pemidaian
Hasil yang Diharapkan	Performa aplikasi menurun drastis dan *props* tidak berhasil dipindai dengan baik
Hasil yang Diperoleh	Aplikasi berhasil memindai kelima *props* walaupun hasil pemindaian buruk dan performa yang menurun, rata-rata FPS yang tercatat adalah 3.4

Pada skenario 5 ini, dilakukan beberapa kali pengujian untuk mengukur FPS dari aplikasi. Tabel percobaan dapat dilihat pada Tabel 5.8.

Tabel 5.8 Percobaan Enam *Props*

Percobaan ke-	**Hasil FPS yang Didapatkan**
1	2
2	5
3	4

4	5
5	1

Berdasarkan beberapa pengujian didapatkan hasil dengan FPS yang sangat rendah, dengan nilai rata-rata 3.4. Dengan hasil ini, aplikasi tidak bisa digunakan dengan lancar walaupun keenam *props* telah berhasil dipindai.

Gambar 5.6 Enam *Props*

Dari pengujian yang dilakukan, aplikasi berhasil memindai *props* tetapi hasil pengujian yang dilakukan tidak sempurna. Performa perangkat pun berkurang dengan sangat drastis. Dengan pemindaian tersebut, aplikasi banyak terdapat *lag* yang tentu sangat mengganggu. Hal ini sejalan dengan rekomendasi dari Vuforia dimana *props* yang disediakan tidak lebih dari lima buah. Proses pengujian dapat dilihat pada Gambar 5.6.

5.2.6. Pengujian Skenario 6

Pada pengujian skenario 6 ini akan dilakukan pengujian terhadap kesesuaian pembangunan objek berdasarkan *props* yang ada. Skenario dan hasil pengujian dapat dilihat pada Tabel 5.9.

Tabel 5.9 Skenario Pengujian 6

Nama Skenario Pengujian	**Pengujian Kesesuaian Objek dengan *Props***
Kode	SP-06
Tujuan Pengujian	Menguji apakah aplikasi mampu membangun objek berdasarkan lokasi dan ukuran *props*
Kondisi Awal	*Props* telah dipindai pada *stage*
Data *Input*	-
Prosedur Pengujian	Pengguna memilih tombol *done* setelah berhasil memindai *props*
Hasil yang Diharapkan	Aplikasi mampu membangun objek sesuai dengan lokasi dan ukuran *props*
Hasil yang Diperoleh	Aplikasi berhasil membangun objek sesuai dengan lokasi dan ukuran *props*

Gambar 5.7 Ukuran *Props*

Dari hasil pengujian didapat bahwa aplikasi telah berhasil mengimplementasikan fungsional dari Smart Terrain. Hal ini merupakan tujuan utama dari pembuatan aplikasi. Hasil pengujian dapat dilihat pada Gambar 5.7.

5.2.7. Pengujian Skenario 7

Pada pengujian skenario 7 ini akan dilakukan pengujian terhadap karakter dan *raycast* pada permainan sederhana yang disediakan oleh aplikasi. Skenario dan hasil pengujian dapat dilihat pada Tabel 5.10.

Tabel 5.10 Skenario Pengujian 7

Nama Skenario Pengujian	**Pengujian Gerakan dan Rotasi Karakter**
Kode	SP-07
Tujuan Pengujian	Menguji apakah karakter telah sesuai dengan fungsional atau tidak
Kondisi Awal	*Props* sudah dibangun pada *stage*
Data *Input*	-
Prosedur Pengujian	Pengguna melakukan *tap* pada layar *smartphone* di dalam *primary surface*
Hasil yang Diharapkan	Karakter menghadap titik *tap* dan menuju titik tersebut
Hasil yang Diperoleh	Karakter berhasil menghadap dan menuju titik tersebut.

Pada pengujian ini, karakter telah berhasil menghadap dan menuju titik *tap* pengguna. Hal ini menunjukkan fungsional *raycast* aplikasi telah berhasil. Namun terdapat *delay* pada animasi dimana perpindahan animasi tidak tepat pada titik tersebut. Pengujian skenario ini dapat dilihat pada Gambar 5.8.

Gambar 5.8 Gerakan Karakter

5.2.8. Pengujian Skenario 8

Pada pengujian skenario 6 ini masih dilakukan pengujian terhadap karakter pada lingkungan permainan sederhana. Pengujian ini juga berhubungan dengan pembangunan *collider* pada objek yang dibangun. Hasil pengujian dapat dilihat pada Tabel 5.11.

Tabel 5.11 Pengujian Skenario 8

Nama Skenario Pengujian	**Pengujian *Collider* pada *Props***
Kode	SP-08
Tujuan Pengujian	Menguji apakah *collider* sudah terbentuk sehingga karakter tidak bisa melewati objek yang dibangun pada *props*
Kondisi Awal	Objek telah dibangun pada *props* di *stage*
Data *Input*	-
Prosedur Pengujian	Pengguna melakukan *tap* pada salah

Nama Skenario Pengujian	Pengujian *Collider* pada *Props*
	satu objek *props* yang telah dibangun
Hasil yang Diharapkan	Karakter tidak bisa berjalan melewati objek tersebut
Hasil yang Diperoleh	Karakter tidak bisa berjalan melewati objek

Gambar 5.9 *Collider Props*

Pada pengujian ini karakter ditabrakkan pada salah satu *props* yang telah dibangun, hasilnya karakter tidak bisa melawati objek tersebut. Hal tersebut berarti *collider* yang dibangun pada objek telah berfungsi dengan sebagaimana mestinya. Hasil pengujian dapat dilihat pada Gambar 5.9.

5.2.9. Pengujian Skenario 9

Pada pengujian skenario 7 ini dilakukan pemakaian pengguna langsung. Hasilnya berupa nilai dengan *range* 1 (sangat tidak bagus) sampai 5 (sangat bagus). Hasil pengujian dapat dilihat pada Tabel 5.12.

Tabel 5.12 Skenario Pengujian 9

Nama Skenario Pengujian	**Pengujian Aplikasi Kepada Pengguna Langsung**
Kode	SP-09
Tujuan Pengujian	Menguji apakah aplikasi sudah berjalan dengan baik dan tanpa kendala
Kondisi Awal	Pengguna menjalankan aplikasi
Data *Input*	-
Prosedur Pengujian	Pengguna dapat memilih menu mana saja sesuai keinginan pengguna
Hasil yang Diharapkan	Nilai rata-rata 3.00 dari 5.00
Hasil yang Diperoleh	Nilai rata-rata 3.24 dari 5.00

Pengujian ini dibebaskan pada pengguna untuk menggunakan aplikasi sesuai keinginan pengguna, dan memakai *props* sesuai dengan keinginan pengguna pula. Ada empat pengguna yang menjadi penguji aplikasi ini. Detail nilai per atribut pengujian dapat dilihat pada Tabel 5.13, Tabel 5.14, Tabel 5.15, Tabel 5.16, dan Tabel 5.17.

Tabel 5.13 Nilai Tampilan Aplikasi

Tampilan Aplikasi		
No.	**Nama Pengguna**	**Nilai**
1	Mahen	4/5
2	Andrie	4/5
3	Rizka	4/5
4	Didik	4/5

Tabel 5.14 Nilai Kecepatan Pemindaian

Kecepatan Pemindaian		
No.	**Nama Pengguna**	**Nilai**

1	Mahen	3/5
2	Andrie	4/5
3	Rizka	2/5
4	Didik	4/5

Tabel 5.15 Nilai Pembangunan Objek

Kesesuaian Hasil Pembangunan Objek		
No.	**Nama Pengguna**	**Nilai**
1	Mahen	3/5
2	Andrie	4/5
3	Rizka	3/5
4	Didik	5/5

Tabel 5.16 Nilai Bebas Kendala

Bebas Kendala		
No.	**Nama Pengguna**	**Nilai**
1	Mahen	3/5
2	Andrie	2/5
3	Rizka	3/5
4	Didik	4/5

Tabel 5.17 Nilai Permainan Sederhana

Permainan Sederhana		
No.	**Nama Pengguna**	**Nilai**
1	Mahen	4/5
2	Andrie	3/5
3	Rizka	4/5
4	Didik	4/5

Berdasarkan hasil nilai per atribut, maka didapatkanlah hasil rata-rata. Hasil rata-rata nilai dari aplikasi dapat dilihat pada Tabel 5.18.

Tabel 5.18 Nilai Aplikasi

Nama Atribut Pengujian	**Nilai Rata-rata (rentang nilai 1-5)**
Tampilan aplikasi	4/5
Kecepatan pemindaian	3.25/5
Kesesuaian hasil pembangunan objek terhadap *props* (posisi dan ukuran)	3.75/5
Bebas kendala	3/5
Permainan sederhana	3.75/5

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan tugas akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari proses pengerjaan selama perancangan, implementasi, dan proses pengujian aplikasi yang dilakukan, dapat diambil kesimpulan sebagai berikut.

1. Aplikasi telah dapat mendeteksi *props* pada *stage* sebanyak lima buah
2. *Props* yang dibangun haruslah yang memiliki tekstur dan *pattern* cukup unik, benda dengan warna polos dan transparan tidak akan bisa dipindai dengan sempurna
3. Sampai saat ini, aplikasi ini harus menggunakan target sebagai metode deteksi awal
4. Aplikasi telah dapat membangun lingkungan 3D yang peletakan objeknya sesuai kehendak pengguna
5. Aplikasi telah dapat menjalankan permainan sederhana berdasarkan objek-objek yang dibangun
6. Smart Terrain mempunyai potensi untuk dijadikan sebuah pengembangan *game* dengan menggunakan *augmented reality*, karena walaupun fitur yang ada sekarang sangat terbatas dan perlu banyak modifikasi, aspek kedinamisam sudah cukup sangat membantu dalam sebuah *game*

6.2. Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang. Di antaranya adalah sebagai berikut.

1. Agar memberikan kesan yang baik pada target, maka bisa menggunakan kaleng yang dicetak langsung (bukan ditempel)
2. Melalui teknologi Smart Terrain ini, pembuatan *game* akan menjadi sangat menarik jika dibangun menggunakan teknologi ini
3. Untuk pengembangan aplikasi sebaiknya dikhususkan untuk perangkat dengan layar besar, seperti *tablet*. Karena dengan layar yang besar, pengguna akan lebih leluasa pula menjalankan aplikasi tersebut
4. Untuk pengembangan aplikasi sebaiknya jangan terpaku dengan versi terbaru dari SDK, karena jika dalam tahap pengembangan mengganti ke versi yang baru akan banyak ketidaksesuaian yang harus diperbaiki lagi, hal itu tentu lebih menyulitkan untuk pengembang

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diambil selama pengerjaan tugas akhir serta saran-saran tentang pengembangan yang dapat dilakukan terhadap tugas akhir ini di masa yang akan datang.

6.1. Kesimpulan

Dari proses pengerjaan selama perancangan, implementasi, dan proses pengujian aplikasi yang dilakukan, dapat diambil kesimpulan sebagai berikut.

1. Aplikasi telah dapat mendeteksi *props* pada *stage* sebanyak lima buah
2. *Props* yang dibangun haruslah yang memiliki tekstur dan *pattern* cukup unik, benda dengan warna polos dan transparan tidak akan bisa dipindai dengan sempurna
3. Sampai saat ini, aplikasi ini harus menggunakan target sebagai metode deteksi awal
4. Aplikasi telah dapat membangun lingkungan 3D yang peletakan objeknya sesuai kehendak pengguna
5. Aplikasi telah dapat menjalankan permainan sederhana berdasarkan objek-objek yang dibangun
6. Smart Terrain mempunyai potensi untuk dijadikan sebuah pengembangan *game* dengan menggunakan *augmented reality*, karena walaupun fitur yang ada sekarang sangat terbatas dan perlu banyak modifikasi, aspek kedinamisam sudah cukup sangat membantu dalam sebuah *game*

6.2. Saran

Berikut saran-saran untuk pengembangan dan perbaikan sistem di masa yang akan datang. Di antaranya adalah sebagai berikut.

1. Agar memberikan kesan yang baik pada target, maka bisa menggunakan kaleng yang dicetak langsung (bukan ditempel)
2. Melalui teknologi Smart Terrain ini, pembuatan *game* akan menjadi sangat menarik jika dibangun menggunakan teknologi ini
3. Untuk pengembangan aplikasi sebaiknya dikhususkan untuk perangkat dengan layar besar, seperti *tablet*. Karena dengan layar yang besar, pengguna akan lebih leluasa pula menjalankan aplikasi tersebut
4. Untuk pengembangan aplikasi sebaiknya jangan terpaku dengan versi terbaru dari SDK, karena jika dalam tahap pengembangan mengganti ke versi yang baru akan banyak ketidaksesuaian yang harus diperbaiki lagi, hal itu tentu lebih menyulitkan untuk pengembang

DAFTAR PUSTAKA

[1] Unity, "Unity 5.0," Unity3D, [Online]. Available: http://unity3d.com. [Accessed 25 4 2015].

[2] Unity3D, "Unity Asset Store," Unity, [Online]. Available: https://www.assetstore.unity3d.com/en/#!/search/simple%20particle%20pack. [Accessed 14 6 2015].

[3] K. Bonsor, "How Augmented Reality Works," [Online]. Available: http://computer.howstuffworks.com/augmented-reality.htm. [Accessed 15 November 2014].

[4] Qualcomm Vuforia, "Unity Extension - Vuforia v3.0," [Online]. Available: https://developer.vuforia.com/resources/sdk/unity. [Accessed 5 Desember 2014].

[5] Qualcomm, "Qualcomm Vuforia: Revealing a Whole New World," [Online]. Available: https://www.qualcomm.com/products/vuforia. [Accessed 20 November 2014].

[6] Vuforia, "Vuforia 4.0," Vuforia, [Online]. Available: http://www.developer.vuforia.com. [Accessed 22 4 2015].

[7] Qualcomm Vuforia, "Smart Terrain," [Online]. Available: https://developer.vuforia.com/resources/dev-guide/smart-terrain. [Accessed 11 Desember 2014].

[8] S. Petrovic, "3rd Person Basic Movement Rotation," Scott Petrovic, [Online]. Available: http://www.scottpetrovic.com/blog/2009/11/unity3d-3rd-person-basic-movementrotation-wsource/. [Accessed 4 6 2015].

BIODATA PENULIS

Penulis, Bestama Abhi Priambada dilahirkan di Mataram, 4 Oktober 1993, merupakan anak pertama dari tiga bersaudara yang dibesarkan pula di kota Mataram.

Penulis menempuh pendidikan formal di SDN 41 Mataram (1999-2005), SMPN 2 Mataram (2005-2008), dan SMAN 1 Mataram (2008-2011). Pada tahun 2011, penulis menempuh pendidikan S1 di jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember, Surabaya, Jawa Timur.

Di jurusan Teknik Informatika, penulis mengambil bidang minat Interaksi Grafika dan Seni dan memiliki kompetensi di beberapa subjek seperti Sistem *Game*, Realitas Virtual, Perancangan Perangkat Lunak, dan Pemrograman Perangkat Mobile. Selama berada di dunia kampus, penulis juga aktif sebagai asisten dosen untuk mata kuliah Analisis dan Perancangan Sistem. Selain itu penulis juga aktif di bidang non akademik. Organisasi yang pernah diikuti penulis adalah menjadi staff Departmen Hubungan Luar Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) periode 2012-2013. Penulis juga pernah menjabat sebagai ketua Schematics 2013 pada tahun 2013 yang merupakan acara tingkat nasional yang diselenggarakan oleh Teknik Informatika. Penulis dapat dihubungi melalui alamat email bestama.abhi@windowslive.com