RANCANG BANGUN PENAMPIL BERBASIS *AUGMENTED REALITY* APLIKASI KAMUS KAMERA *REALTIME* BERBASIS ANDROID

SKRIPSI

Untuk Memenuhi sebagian persyaratan mencapai gelar Sarjana Komputer

IRFANDI ACHMAD

115060800111047

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN NASIONAL

UNIVERSITAS BRAWIJAYA

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

MALANG

2015

LEMBAR PERSETUJUAN

RANCANG BANGUN PENAMPIL BERBASIS AUGMENTED REALITY APLIKASI KAMUS KAMERA REALTIME BERBASIS ANDROID

SKRIPSI

LABORATORIUM

PEMROGRAMAN APLIKASI PERANGKAT BERGERAK

Diajukan untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun oleh :

IRFANDI ACHMAD

NIM. 115060800111047

Skripsi ini telah diperiksa dan disetujui oleh dosen pembimbing

Dosen Pembimbing I	**Dosen Pembimbing II**
Dr.Eng. Herman Tolle, ST., MT.	**Wibisono Sukmo Wardhono, S.T, M.T**
NIP.19740823 200012 1 001	**NIK. 82040406110091**

LEMBAR PENGESAHAN

LEMBAR PERNYATAAN
ORISINALITAS SKRIPSI

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah SKRIPSI ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No.20 Tahun 2003, Pasal 25 Ayat 2 dan Pasal 70).

Malang, 18 Agustus 2015

Mahasiswa

Irfandi Achmad

NIM. 115060800111047

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Tuhan yang Maha Esa, yang telah melimpahkan berkat dan anugerah-Nya kepada penulis sehingga penulis dapat menyelesaikan skripsi ini yang berjudul "**Rancang Bangun Penampil Berbasis Augmented Reality Aplikasi Kamus Kamera Realtime Berbasis Android**"

Pada penyusunan Skripsi ini tidak semata-mata hasil kerja penulis sendiri, melainkan juga berkat dan bimbingan dan dorongan dari pihak-pihak yang telah membantu, baik secara materi maupun non materi. Maka dari itu penulis ingin mengucapkan banyak terima kasih yang tak terhingga serta penghargaan yang setinggi-tingginya kepada orang-orang yang telah membantu penulis secara langsung maupun tidak langsung kepada yang terhormat:

1. Bapak Dr. Eng. Herman Tolle, S.T., M.T. dan Bapak Wibisono Sukmo Wardhono, S.T., M.T. selaku dosen pembimbing skripsi yang telah membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
2. Bapak Drs. Marji, M.T selaku Ketua Prodi Informatika / Ilmu Komputer yang telah memberikan kesempatan kepada penulis untuk menyelesaikan skripsi ini.
3. Ayah, Ibu, dan seluruh keluarga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan do'a dan semangat demi terselesaikannya skripsi ini.
4. Segenap bapak dan ibu dosen program studi Informatika / Ilmu Komputer beserta seluruh staff administrasi yang telah membantu selama kegiatan perkuliahan.
5. Seluruh Civitas Akademika Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Prodi Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.

6. Teman – teman Teknik Informatika angkatan 2010 yang telah memberikan masukan dan inspirasi kepada penulis selama menempuh studi dan menyelesaikan skripsi ini.
7. Sintia Farach Dhiba, atas semua kasih sayang yang (semoga selalu) ada, dukungan yang bermanfaat, dan pengertiannya.
8. Seluruh pihak yang telah membantu kelancaran penulisan tugas akhir yang tidak dapat disebutkan satu persatu.

Penulis menyadari bahwa penulisan skripsi ini jauh dari kata sempurna. Oleh karena itu, penulis mengharapkan masukan berupa saran dan kritik dari semua pihak demi tercapainya kesempurnan dalam skripsi ini. Akhir kata semoga penulisan skripsi ini dapat memberikan manfaat bagi semua pihak.

Malang, Juni 2015

Penulis

repository.ub.ac.id

ABSTRAK

Irfandi Achmad, 2015. Rancang Bangun Penampil Berbasis Augmented Reality Aplikasi Kamus Kamera Realtime Berbasis Android. Skripsi Program Studi Informatika / Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen Pembimbing : Dr.Eng. Herman Tolle, ST., MT. dan Wibisono Sukmo Wardhono, S.T, M.T.

Aplikasi OCR(Optical Character Recognition) translation dapat dimanfaatkan ketika seseorang mengunjungi luar negeri untuk menerjemahkan teks gambar yang tidak dipahami. OCR merupakan sistem yang mampu mengenali teks pada suatu citra. Citra atau gambar tersebut diambil oleh OCR melalui kamera. Ketika proses dari pengenalan karakter(OCR) selesai aplikasi menampilkan hasil terjemahan pada bidang yang terdeteksi. Penelitian ini bertujuan untuk mengembangkan penampil pada aplikasi kamus kamera realtime dengan memanfaatkan kamera smartphone sebagai media untuk menangkap gambar nyata menggunakan teknologi augmented reality. Penelitian ini menyajikan algoritma menerjemahkan teks, menampilkan hasil terjemahan, dan mengkoreksi kata salah. penelitian ini memanfaatkan Vuforia SDK untuk mendeteksi kata pada gambar. Aplikasi dapat berjalan baik pada jarak optimal 30-150cm, berjalan baik pada tingkat kecerahan gambar -40% sampai 20% dari kecerahan normal, dan berjalan baik pada sudut pandang 90˚ sampai 40˚. Aplikasi ini memiliki tingkat akurasi menerjemahkan dan menampilkah hasil terjemahan mencapai 80% tingkat keberhasilannya.

Kata kunci : Augmented Reality, OCR, dan penerjemah.

UNIVERSITAS BRAWIJAYA

repository.ub.ac.id

ABSTRACT

Irfandi Achmad, 2015. Design Based Augmented Reality Applications Camera Dictionary Android-Based Real Time. Skripsi Program Studi Informatika / Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen Pembimbing : Dr.Eng. Herman Tolle, ST., MT. and Wibisono Sukmo Wardhono, S.T, M.T.

OCR Aplication (Optical Character Recognition) translation can be used when someone visits abroad to translate text images that are not understood. OCR is a system that can recognize text in an image. The image or the image taken by the OCR through the camera. When the process of character recognition (OCR) completed application displays the translation on the field that is detected. This research aims to develop a viewer in the application dictionary realtime camera by utilizing smartphone camera as a medium to capture real images using augmented reality technology. This study presents an algorithm to translate text, displays the translation results, and correct the wrong word. This study utilizes Vuforia SDK to detect words on the picture. Applications can be run either at the optimum distance 30-150cm, runs good at image brightness levels up to 20% - 40% of normal brightness, and good walking on the viewing angle 90˚ to 40˚. This application has a high degree of accuracy and menampilkah translate translation results reached 80% success rate.

Keyword : Augmented Reality, OCR, and translation.

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

LEMBAR PERSETUJUAN ..... **II**
LEMBAR PENGESAHAN ..... **III**
LEMBAR PERNYATAAN ..... **IV**
KATA PENGANTAR ..... **V**
ABSTRAK ..... **VII**
DAFTAR ISI ..... **IX**
DAFTAR GAMBAR ..... **XII**
DAFTAR TABEL ..... **XIV**
DAFTAR SOURCE CODE ..... **XV**
BAB I **1**
1.1 LATAR BELAKANG 1
1.2 RUMUSAN MASALAH 2
1.3 BATASAN MASALAH 2
1.4 TUJUAN PENELITIAN 3
1.5 MANFAAT PENELITIAN 3
1.6 SISTEMATIKA PENULISAN LAPORAN 3
BAB II **5**
2.1 KAJIAN PUSTAKA 5
2.2 *AUGMENTED REALITY* 5
2.2.1 Cara Kerja Augmented Reality *6*
2.2.2 Marker *6*
2.2.3 Markerless *7*
2.3 *OPTICAL CHARACTER RECOGNITION* (*OCR*) 8
2.4 VUFORIA 9
2.4.1 Cara Kerja Vuforia Text Recognition *9*
2.4.1 Word-List *10*
2.5 OPENGL 10
2.5.1 Open GL Primitive dan Attribute *11*
2.4.2 OpenGL Text Rendering *12*
2.6 LEVENSHTEIN ALGORITHM 12
2.7 SISTEM OPERASI *ANDROID* 13
2.5.1 Sejarah Sistem Operasi Android *13*
2.5.2 Fitur-fitur Android *13*
2.6 *UNIFIED MODELLING LANGUAGE (UML)* 14

2.6.1 Class Diagram *14*
2.6.2 Use case *15*
2.6.3 Squance Diagram *15*
2.6.4 Activity Diagram *15*
2.6.5 Entity Diagram Relationship(ERD) *15*
BAB III **16**
3.1 STUDI LITERATURE 17
3.2 ANALISA KEBUTUHAN 17
3.3 PERANCANGAN SISTEM 18
3.4 IMPLEMENTASI 18
3.5 PENGUJIAN 18
3.6 KESIMPULAN DAN SARAN 19
BAB IV **21**
4.1 ANALISIS KEBUTUHAN 21
4.1.1 Gambaran Umum Perangkat Lunak *21*
4.1.2 Identifikasi Aktor *22*
4.1.3 Analisis Kebutuhan Fungsional *22*
4.1.4 Analisis Kebutuhan Non-Fungsional *23*
4.1.5 Diagram Use Case *23*
4.1.6 Skenario Use Case *24*
4.2 PERANCANGAN 27
4.2.1 Perancangan Arsitektural *27*
4.2.2 Activity Diagram *28*
4.2.3 Algoritma Mendeteksi teks *31*
4.2.4 Algoritma Pengoreksian kata salah *32*
4.2.5 Algoritma Menerjemahkan Teks *33*
4.2.6 Penggunaan Library Vuvoria pendekteksi teks *35*
4.2.7 Squence diagram *38*
4.2.8 Class Diagram *41*
4.2.9 Perancangan Basis Data *44*
4.2.10 Perancangan Page Flow *45*
BAB V **47**
5.1 SPESIFIKASI SISTEM 47
5.1.1 Spesifikasi Linkungan Sistem Perangkat Lunak *48*
5.1.2 Spesifikasi Lingkungan Sistem Perangkat Keras *48*
5.2 BATASAN – BATASAN IMPLEMENTASI 48
5.3 IMPLEMENTASI *USE CASE* 49
5.3.1 Implementasi Fitur Mendeteksi Kata *49*
5.3.2 Implementasi Fitur Menerjemahkan teks *50*
5.3.3 Implementasi Fitur Mengkoreksi Kata *52*

BAB VI 55
6.1 DATA UJI 55
6.2 PENGUJIAN 61
6.2.1 Pengujian Tingkat Kecerahan 61
6.2.2 Pengujian Jarak Optimal 63
6.2.3 Pengujian Sudut Pandangan 64
6.2.4 Pengujian Akurasi 64
6.2.5 Pengujian Validasi 65
6.3 ANALISIS PENGUJIAN 72
6.3.1 Analisis Hasil Pengujian Tingkat Kecerahan 72
6.3.2 Analisis Hasil Pengujian Jarak Optimal 72
6.3.3 Analisis Hasil Pengujian Sudut Pandang 73
6.3.4 Analisis Hasil Pengujian Akurasi 74
6.3.6 Analisis Hasil Pengujian Validasi 74
BAB VII 75
7.1 KESIMPULAN 75
7.2 SARAN 76
DAFTAR PUSTAKA XVI

DAFTAR GAMBAR

GAMBAR 2. 1 DIAGRAM ALIR SISTEM *AUGMENTED REALITY* SEDERHANA[9]. 6
GAMBAR 2. 2 CONTOH MAKER PADA *AUGMENTED REALITY* [9]. 7
GAMBAR 2. 3 CONTOH EDGE YANG TERDETEKSI[9]. 8
GAMBAR 2. 4 BAGIAN OBJEK YANG TERDETEKSI SEBAGAI CORNER[9]. 8
GAMBAR 2. 5 STRUKTUR WORDLIST PADA VUVORIA SDK[9]. 10
GAMBAR 2. 6 OBJEK *PRIMITIVE POINT* DAN JENIS *LINE*(BACA DARI KIRI KE KANAN)[21]. 12
GAMBAR 2. 7 ALGORITMA *LEVENSHEIN*[3]. 12
GAMBAR 2. 8 CONTOH SEDERHANA *CLASS DIAGRAM*[22]. 15
GAMBAR 3. 1 DIAGRAM ALIR METODOLOGI PENELITIAN 16
GAMBAR 4. 1 VISUALISASI GAMBARAN UMUM APLIKASI 22
GAMBAR 4. 2 DIAGRAM *USE CASE* APLIKASI KAMUS PENERJEMAH *REALTIME* BERBASIS *AUGMENTED REALITY*. 24
GAMBAR 4. 3 RANCANGAN ARSITEKTURAL SISTEM 27
GAMBAR 4. 4 *ACTIVITY DIAGRAM* MENDEKTEKSI TEKS 29
GAMBAR 4. 5 *ACTIVITY DIAGRAM* KOREKSI KATA SALAH 30
GAMBAR 4. 6 *ACTIVITY DIAGRAM* MENERJEMAHKAN TEKS 31
GAMBAR 4. 7 ALGORITMA MENDETEKSI KATA 32
GAMBAR 4. 8 ALGORITMA KOREKSI KATA 32
GAMBAR 4. 9 ALGORITMA MENERJEMAHKAN TEKS 33
GAMBAR 4. 10 ALGORITMA MENAMBAHKAN HASIL TERJEMAHAN 34
GAMBAR 4. 11 ALGORITMA MENAMPILKAN HASIL TERJEMAHAN 35
GAMBAR 4. 12 *DIAGRAM* SKUENSIAL MENDETEKSI TEKS 39
GAMBAR 4. 13 DIAGRAM SKUENSIAL MENGOREKSI KATA SALAH. 40
GAMBAR 4. 14 DIAGRAM SKUENSIAL MENERJEMAHKAN TEKS 41
GAMBAR 4. 15 CLASS DIAGRAM PENGGUNA 42
GAMBAR 4. 16 *CLASS DIAGRAM TRANSLATEDWORDS* 43
GAMBAR 4. 17 *CLASS DIAGRAM FILEOPERATION* 43
GAMBAR 4. 18 *ENTITY DIAGRAM* SISTEM 45
GAMBAR 4. 19 *PAGE FLOW DIAGRAM* SISTEM 46
GAMBAR 5. 1 POHON IMPLEMENTASI 47
GAMBAR 5. 2 TAMPILAN DARI FITUR MENERJEMAHKAN TEKS 52
GAMBAR 5. 3 TAMPILAN FITUR MENGKOREKSI KATA SALAH 54
GAMBAR 6. 1 VARIASI TINGKAT KECERAHAN DATA UJI KE-1 62
GAMBAR 6. 2 SKENARIO PENGUJIAN JARAK 63
GAMBAR 6. 3 SKENARIO PENGUJIAN SUDUT PANDANGAN 64
GAMBAR 6. 4 HASIL PENGUJIAN KECERAHAN 72
GAMBAR 6. 5 HASIL PENGUJIAN JARAK 73

GAMBAR 6. 6 HASIL PENGUJIAN SUDUT PANDANG..... 73
GAMBAR 6. 7 HASIL PENGUJIAN AKURASI..... 74

DAFTAR TABEL

TABEL 2. 1 TABEL DISTRIBUSI ANDROID[4]..... 13
TABEL 4. 1 TABEL IDENTIFIKASI AKTOR 22
TABEL 4. 2 TABEL KEBUTUHAN FUNGSIONAL 22
TABEL 4. 3 SPESIFIKASI KEBUTUHAN NON-FUNGSIONAL 23
TABEL 4. 4 SKENARIO *USE CASE* MENDEKTEKSI TEKS 24
TABEL 4. 5 SKENARIO USE CASE MENGOREKSI KATA SALAH. 25
TABEL 4. 6 SKENARIO *USE CASE* MENERJEMAHKAN DAN MENAMPILKAN HASIL TERJAMAHAN 26
TABEL 4. 7 TABEL PENGGUNAAN LIBRARY VUFORIA 35
TABEL 4. 8 STRUKTUR *CLASS TRANSLATEDWORDS* 43
TABEL 4. 9 STRUKTUR *CLASS TRANSLATEDWORD* 43
TABEL 6. 1 DATA UJI 56

DAFTAR SOURCE CODE

SOURCE CODE 5. 1 FITUR MENDETEKSI KATA 50
SOURCE CODE 5. 2 FITUR MENERJEMAHKAN TEKS 51
SOURCE CODE 5. 3 FITUR MENGKOREKSI KATA 53

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pariwisata di Indonesia mengalami perkembangan yang pesat pada tahun 2014[24]. Perkembangan tersebut tentunya dibarengi dengan pesatnya perkembangan teknologi bidang kepariwisataan abad ini[25]. Perkembangan teknologi tersebut dapat dimanfaatkan untuk sektor kepariwisataan Indonesia yang umumnya kondisi wisatanya masih menggunakan papan petunjuk berbahasa Indonesia untuk memberikan informasi kepada wisatawan. Salah satu teknologi yang dapat dimanfaatkan untuk mempermudah wisatawan asing untuk mengenali papan petunjuk tersebut adalah *OCR* (*optical character recognition*) translation yang digunakan untuk menerjemahkan tanda atau menu. *OCR translation* dapat dimanfaatkan untuk menerjemahkan gambar teks dari bahasa Indonesia kedalam bahasa inggris. *OCR* dapat diibaratkan sebagai mata yang mampu mengenali lingkungan dan memberikan input ke otak. *OCR* merupakan sistem yang mampu mengenali teks pada suatu citra. Citra atau gambar tersebut diambil oleh *OCR* melalui kamera. *OCR* melakukan image processing dari citra yang didapatkan[15]. Sistem pengenalan teks *realtime* membutuhkan penampil realtime untuk menampilkan hasil terjemahan secara realtime yang baik untuk membantu wisatawan menerjemahkan papan petunjuk.

Selain itu, sistem pengenalan teks realtime juga membutuhkan tingkat fleksibilitas yang tinggi agar dalam hampir semua kondisi user dapat menerjemahkan kata melalui sistem ini. Fleksibilitas tersebut dijawab dengan memanfaatkan teknologi mobile yang memiliki tingkat fleksibilitas yang tinggi[16].

Teknologi *mobile*, *smartphone* khususnya menjadi produk yang tidak bisa kita pisahkan dalam hidup manusia sekarang ini. Dalam komunikasi sehari-hari *smartphone* menjadi perangkat yang diunggulkan karena fitur-fiturnya sangat mendukung. *Smartphone* memiliki banyak keuntungan yaitu *fleksibel* dapat dibawa kemana-mana dan kapan saja serta memiliki '*user-friendly*

interface'. Sistem operasi *Android* menjadi system operasi *smartphone* yang paling marak diminati oleh pengguna *smartphone*[1].

Kemajuan terbaru dalam komputasi *mobile*, computer grafis, nirkabel, dan teknologi sensor memungkinkan pengembangan *Augmented Reality*(*AR*) pada aplikasi *smartphone*[11]. AR merupakan terobosan untuk meredakan operasi yang kompleks. AR merupakan campuran realitas *virtual* dan aktual dengan realitas waktu yang sama[2]. Teknologi *Augmented Reality* banyak dimanfaatkan pada berbagai bidang seperti, kesehatan, militer, pendidikian, pariwisata, ekonomi, dan lain –lain. Teknologi *augmented reality* yang diterapkan pada perangkat mobile menawarkan banyak kemudahan kepada user.

Skripsi ini bertujuan untuk merancang dan mengimplementasikan aplikasi dengan memanfaatkan kamera *smartphone* sebagai media untuk menerjemahkan bahasa Indonesia ke bahasa inggris secara *realtime* menggunakan teknologi *augmented reality*.

Berdasarkan latar belakang tersebut skripsi ini diajukan dengan judul skripsi "Rancang Bangun Penampil berbasis *Augmented Reality* Aplikasi Kamus Kamera *Realtime* berbasis *Android*".

1.2 Rumusan Masalah

Dari penjelasan latar belakang diatas, rumusan masalah di dalam penelitian ini adalah:

1. Bagaimana rancangan aplikasi penampil berbasis *augmented reality* aplikasi kamus kamera *realtime*?
2. Bagaimana mengimplementasikan aplikasi penampil berbasis *augmented reality* aplikasi kamus kamera *realtime*?
3. Bagaimana tingkat perfomansi sistem berdasarkan pencahayaan, jarak, dan sudut pandangan objek?

1.3 Batasan Masalah

Berdasarkan latar belakang serta rumusan masalah diatas, penelitian ini dibatasi sebagai berikut:

1. Berbasis system operasi Android versi 4.0 keatas secara *offline*.

2. Bahasa pemrograman yang digunakan untuk membuat aplikasi perangkat bergerak adalah bahasa java.
3. Aplikasi ini hanya bisa menerjemahkan dari bahasa Indonesia ke inggris.
4. Sistem operasi yang digunakan untuk merancang dan mengimplementasikan aplikasi ini adalah Windows 8.1.
5. Aplikasi ini menggunakan Vuforia SDK 4.0
6. Aplikasi ini dapat mengenali tulisan dengan font sans serif dan hampir semua serif. Aplikasi ini juga mendukung *plain text*, *bold*, *italic*, *underline* text dan kombinasi tersebut.
7. Aplikasi ini hanya mampu menerjemahkan maksimal 2 kata dalam satu periode pengenalan gambar.
8. Aplikasi ini mengkoreksi kata dengan mengunakan metode *levensthein*.

1.4 Tujuan Penelitian

Berdasarkan rumusan masalah diatas, tujuan di dalam penelitian ini adalah:

1. Menghasilkan rancangan dan mengimplementasi aplikasi penampil berbasis *augmented reality* aplikasi kamus kamera *realtime* yang baik berdasarkan pengujian tingkat kecerahan, jarak, dan sudut pandangan objek.

1.5 Manfaat Penelitian

Beberapa manfaat yang diperoleh dari penelitian ini adalah;

a) Bagi pengguna aplikasi
    1. Mempermudah wisatawan asing dalam menerjemahkan bahasa Indonesia kedalam bahasa internasional yaitu bahasa inggris.

1.6 Sistematika Penulisan Laporan

Sistematika penulisan skripsi ini antara lain :

Bab I : Pendahuluan

Pada bab ini berisi tentang latar belakang masalah, rumusan masalah, tujuan penelitian, batasan masalah, manfaat penelitian, dan sistematika penulisan skripsi.

Bab II : Tinjauan Pustaka

Pada bab ini diulas mengenai teori yang mendukung aplikasi Penampil berbasis *Augmented Reality* Aplikasi Kamus Kamera *Realtime* berbasis *Android* pada rancangan, implementasi, hingga pengujian.

Bab III : Metodologi Penelitian

Pada bab ini dijelaskan tentang gambaran umum mengenai perancangan aplikasi yang meliputi: *observasi litaratur*, penggunaan dasar teori, analisisi kebutuhan perangkat lunak, metode pengambilan data, serta pemilihan bentuk pengujian yang tepat.

Bab IV : Perancangan

Pada bab ini akan dibahas tentang tahapan-tahapan perancangan system yang akan dibuat.

Bab V : Implementasi

Pada bab ini akan diulas tentang implementasi berdasarkan metodologi dan perancangan.

Bab VI : Pengujian

Pada bab ini akan dipaparkan tantang pengujian terhadap hasil ringkasan dan aplikasi.

Bab VII : Kesimpulan dan saran

Pada bab ini akan dijelaskan kesimpulan dan saran terhadap penelitian ini.

BAB II

TINJAUAN PUSTAKA

2.1 Kajian Pustaka

Generasi teknologi perangkat bergerak saat ini (*smartphone*) menggunakan *hardware* yang cukup untuk pengembangan pada penelitian sebelumnya yang berkaitan dengan *augmented reality* translator. Pengguna aplikasi dibuat bisa berinteraksi secara langsung dengan dunia nyata. Pada penelitian Victor Fragroso menjelaskan *augmented reality* translator dapat dikembangakan menggunakan kamera *smartphone* dan *touchscreen* yang dikombinasikan dengan *OCR*(*optical character recognition*) dan layanan terjemahan *online*(Google Translation API)[14].

Pemanfaatan kamera *smartphone* dan *touchscreen*, digunakan untuk interaksi *Augmented Reality* dengan menekan screen *smartphone* secara langsung untuk menghasilkan terjemahan. Tujuan penelitian kedua ini adalah mengembangkan algoritma yang effisien untuk secara akurat mendeteksi lokasi dan orientasi teks dalam aliran kamera[19].

2.2 *Augmented Reality*

Augmented Reality (*AR*) merupakan salah satu teknologi baru di bidang multimedia. *AR* didefinisikan sebagai teknologi yang dapat menggabungkan dunia nyata dengan dunia maya, bersifat interaktif menurut waktu nyata (*realtime*), serta berbentuk animasi *3D* (Azuma, 1997). Dengan kata lain, *AR* merupakan teknologi yang mampu menggabungkan objek maya dalam dua dimensi (*2D*) atau tiga dimensi (*3D*) ke dalam sebuah lingkungan nyata, kemudian memproyeksikan objek-objek tersebut secara *realtime*[7]. *Augmented Reality* adalah jenis virtual *reality* yang bertujuan untuk menduplikasi lingkungan dunia ke dalam komputer[18].

Dalam perkembangannya, banyak aplikasi yang menyatakan dirinya menggunakan *AR*. Maka dari itu, pendefinisian teknologi *Augmented Reality*

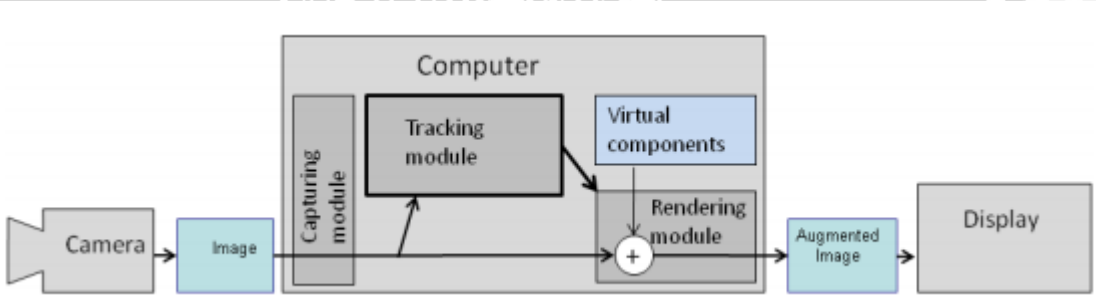
(*AR*) dapat sangat beragam. Berikut adalah beragam pendefinisian Teknologi *AR*:

1. Teknologi *AR* merupakan penggabungan dunia nyata dengan *computer grafis*.
2. Teknologi *AR* menyediakan interaksi dengan dunia nyata secara *realtime*.
3. Melacak objek secara *realtime*
4. *AR* menyediakan pengenalan gambar atau objek.
5. *AR* menyediakan konteks data secara *realtime*.

Dalam penggunaanya, *AR* memanfaatkan kamera yang digunakan secara realime. Kemudian *AR* mengenali objek dengan menggunakan *marker* atau tidak (*markerless*)[5].

2.2.1 Cara Kerja *Augmented Reality*

Seperti yang ada pada gambar 2.2, Modul penangkap menangkap gambar dari kamera. Modul pelacakan (tracking modul) menghitung lokasi yang benar dan orientasi dari *overlay layer*. Modul *render* digunakan untuk menggabungkan gambar asli dan komponen *virtual* kemudian merender *augmented image* ke *display*. Modul pelacakan merupakan inti dari sistem *Augmented Reality* karena pada modul ini terdapat fungsi untuk mendeteksi gambar yang di-*augmented*[9].



Gambar 2. 1 diagram alir sistem *augmented reality* sederhana[9].

2.2.2 Marker

Marker merupakan penanda yang dikenali oleh *AR* untuk memasukkan objek yang dikehendaki kedalam *marker*. *Marker* dapat berupa foto, objek

nyata, atau gambar buatan dengan pola unik. Pada pelacakan berbasis *marker* posisi kamera dan orientasi kamera dhitung dengan marker yang telah ditetapkan[8].Tujuan pelacakan berbasis *marker* adalah mendapatkan garis besar *marker* dan kemudian menyimpulkan lokasi sudut *marker* pada sebuah gambar[9]. Contoh dari marker dapat dilihat pada gambar 2.3.

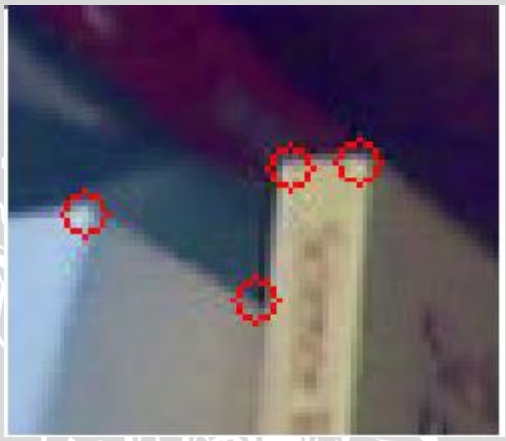
Gambar 2. 2 Contoh maker pada *augmented reality* [9].

2.2.3 Markerless

Penggunakan *marker* merupakan pilihan yang baik, namun dalam kenyataannya ada tempat dimana kita tidak menginginkan penggunaan *marker* atau keadaan yang tidak memungkinkan untuk *marker*. *Markerless* merupakan solusi untuk mengatasi masalah tersebut. Markerless *Augmented Reality* merupakan pengenalan objek *AR* tanpa menggunakan *marker* atau penanda. Dalam pelacakan berbasis *markerless, AR* menggunakan posisi antara pengguna/kamera dan dunia nyata tanpa refrensi apapun hanya menggunakan titik fitur alami seperti *edge*, *corner*, *garis* atau *model 3D*[8].

Dalam pendeteksian markerless diperlukan algoritma dinamakan *tracking algorithm*. *Tracking algorithm* dibagi menjadi tiga fitur yaitu *feature points*(*e.g. corners*), *feature descriptors (e.g. SIFT)* dan *edges*. *Feature point* merupakan area kecil pada gambar yang memiliki pose yang jelas dan posisi yang jelas sesuai gambar 2.5. *Feature descriptor* menggambarkan karakteristik dari gambar. Sedangkan *edge* merupakan garis besar dari objek yang dideteksi kadang *edge* juga muncul pada garis yang bukan objek. *Edge* dicocokan berdasarkan profile dan orientasi objek seperti yang ditunkukkan pada gambar 2.4 [9].

Gambar 2. 3 contoh edge yang terdeteksi[9].



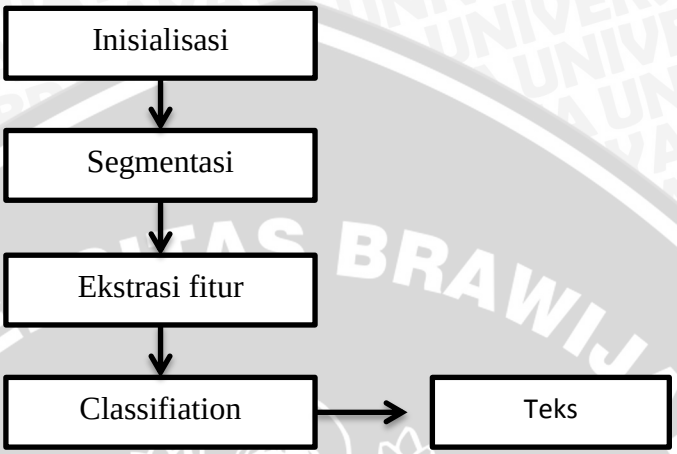
Gambar 2. 4 bagian objek yang terdeteksi sebagai corner[9].

2.3 *Optical Character Recognition* (*OCR*)

Oprical *Character Recognition* atau *OCR* didefiniskan sebagai analisis elektronik dari suatu gambar untuk mengidentifikasi area dari gambar yang berisi informasi tekstual dan mengekstraksi atau mengenali teks dari gambar yang diberikan[15].

Pada pengenalan teks dikenal 4 proses umum yaitu inisialisasi, segmentasi, ektrasi fitur, dan classification. Pada Gambar 2.6 dijelaskan proses pengenalan teks diawali dengan inisialisasi. Proses inisialisasi yang dimaksud adalah mempersiapkan citra digital yang akan dideteksi. Pada proses segmentasi dilakukan proses pemecahan gambar teks menjadi gambar tiap karakter. Selanjutnya pada proses Ekstrasi fitur dilakukan penyimpanan hasil

segmentasi yang mana hasil segmentasi tiap karakter disimbolkan dengan simbol yang unik. Sedangkan pada proses Classification dilakukan proses mencocokan semua fitur ektrasi yang disimpan dengan segmen uji yang ada pada program. Kemudian setelah itu mengeluarkan hasil teks[20].



Gambar 2. 6 Diagram algoritma pengenalan teks[20].

2.4 Vuforia

Vuforia adalah *Augmented Software Development Kit* (*SDK*) yang dikembangkan oleh qualcomm untuk perangkat *mobile* yang memungkinkan pembuatan aplikasi *Augmented Reality*. Vuforia mendukung platform *iOS* dan *Android* dalam pengembangannya. Objek yang mampu dikenali Vuforia *SDK* adalah *3D* objek, gambar, objek silinder/botol, teks, kubus, dan *frame marker*.

2.4.1 Cara Kerja Vuforia Text *Recognition*

Dalam pengenalan text Vuforia dikenal penggunakan Tracker class yang mampu mengenali dan melacak teks dan membuka API untuk proses initializing, deinitializing, starting and stopping subsistem dari text *recognition* Vuforia.

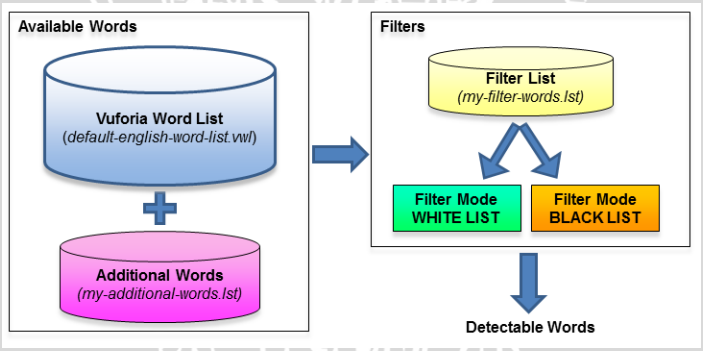
Vuforia mendeteksi dan melacak elemen teks dengan mengamati bagian persegi panjang dari pandangan kamera yang disebut dengan Rest of Interest (ROI). Dua keunikan dari ROI adalah deteksi ROI dan memprediksikan ROI. Jika ditinjau dari sisi performa jika objek full camera maka pendektesian ROI

akan memakan waktu lebih lama sebaliknya proses memprediksikan ROI tidak akan terkena dampak apapun akibat dari objek full kamera.

Ketika text *recognition* difungsikan. Vuforia mencoba untuk mencocokan kata-kata yang ditemukan dalam ROI dengan kata-kata pada refrensi yang dikenal dengan word-list[6].

2.4.1 Word-List

Dalam pengenalan teks, Vuforia memerlukan sebuah dataset yang dikenal dengan "wordlist". *Wordlist* merupakan kumpulan kata yang merepresentasikan teks yang akan dideteksi. *Wordlist* berguna sebagai penyedia daftar kata yang dapat di deteksi oleh Vuforia *text recognation*. Saat ini jumlah kata yang dimiliki *wordlist* Vuforia mencapai 100.000 kata dalam bahasa inggris untuk. *File wordlist* direpresentasikan dengan *ekstensi .vwl* dan pengguna dapat menambahkan kata. Seperti pada gambar 2.5 wordlist memiliki fitur filter dibagi menjadi 2 yaitu black-list dan white-list. *Black-list filtering* memblokir kata tertentu untuk dideteksi sedangkan *white-list filters* memperbolehkan kata tertentu untuk dideteksi[6].



Gambar 2. 5 Struktur wordlist pada Vuvoria SDK[9].

2.5 OpenGL

OpenGL merupakan standart antarmuka perangkat lunak untuk grafis *hardware* yang memungkinkan programmer untuk membuat citra 2 Dimensi yang interaktif ataupun grafis 3 dimensi pada berbagai sistem. OpenGL dapat membuat gambar yang berkualitas tinggi. OpenGL mempermudah dalam membuat model geometri, merubah posisi penglihatan, mengontrol warna dan

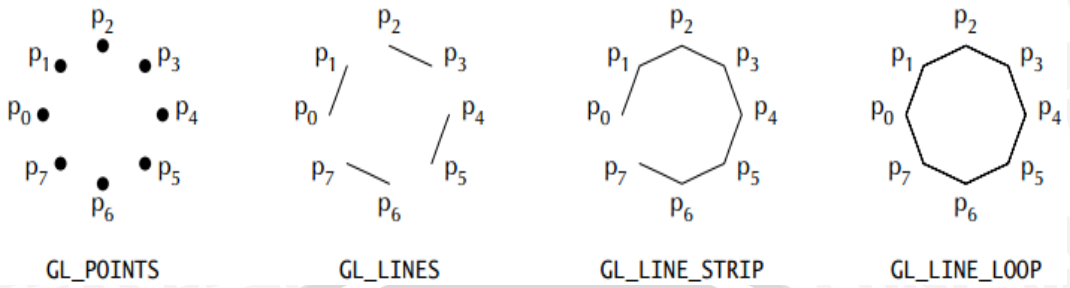
pencahayaan objek ,dan memanipulasi *pixel* dan texture map pada suatu gambar[12].

OpenGL melakukan proses *rendering*. *Rendering* merupakan proses membuat gambar dari model pada computer. Model-model atau objek-objek tersebut dibangun dari objek primitif *geometric* seperti titik, garis, dan *polygon* yang dispesifikan dengan *vertices*. Hasil gambar yang direnderer terdiri dari *pixel-pixel* yang digambar pada *screen*. *Pixel* adalah elemen terkecil yang diletakkan oleh display *hardware* pada screen. Informasi tentang *pixel* (misalkan warna yang digunakan) diorganisasikan memori kedalam bitplane. Bitplane merupakan area didalam memori yang menyimpan satu bit informasi untuk setiap *pixel* pada screen. Kumpulan dari bitplane diorganisasikan kedalam framebuffer yang menyimpan semua informasi yang dibutuhkan display graphics untuk mengkontrol warna dan intensitas semua *pixel* pada screen[13].

2.5.1 *Open GL Primitive* dan *Attribute*

Primitif yang pada OpenGL adalah suatu entitas atomik yang bekerja pada sistem grafis openGL salah satunya adalah *polygon*. *Polygon* merupakan geometris *primitive*. *Polygon* dibentuk dengan kumpulan vertex-vertex. Dasar dari tiga tipe objek *primitive* adalah *point*, *line segment*, dan *polygon*. Tipe-tipe dari objek inilah yang membedakan penggambaran dari suatu vertices yang sama. Berikut adalah tipe objek *primitive points* dan *line segment*[21]:

1 *Point*: *point* merupakan objek primitif yang paling sederhana. Pada gambar 2.7 menggambarkan *point* tersebut.
2 *Line-segment*: *line segment* akan membentuk *line* yang mampu membentuk objek suatu nyata dan realistic. Pada gambar 2.7 menjelaskan macam-macam *line segment*.



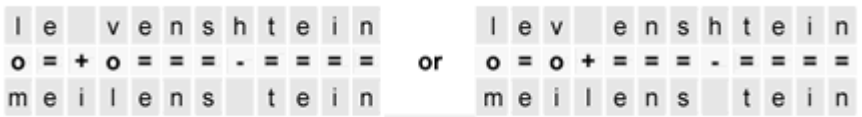
Gambar 2. 6 Objek *primitive point* dan jenis *line*(baca dari kiri ke kanan)[21].

2.4.2 OpenGL Text Rendering

Sebuah penggunaan baru untuk texture rendering adalah merender text [21]. Karakter disimpan pada tekstur 2D. Ketika karakter dirender, *polygon* yang dibentuk sesuai ukuran dipetakan dengan gambar karakter. Teks dapat digambar secara effisien dengan menyimpan karakter-karakter teks didalam sebuah tekstur.

2.6 Levenshtein Algorithm

Algoritma *Levenshtein* merupakan algoritma untuk menghitung jumlah operasi edit diantara dua string. Algoritma ini merupakan cara yang dinamis untuk menghitung perbedaan dua string. Pengertian yang lain Algoritma *Levenshtein* adalah jumlah minimal dari operasi dasar yang diperlukan untuk mengubah String S1 ke dalam S2 Sebuah matrik A digunakan untuk menghitung (m,n) untuk mencari jarak diantara m karakter-karakter string dengan n yang merupakan karakter-karakter dari string yang lain. Seperti contoh pada Gambar 2.6 simbol o(lingkaran) menandakan ketidakcocokan, samadengan(=) menunjukkan kecocokan karakter, dan negatif (-) menggambarkan ada karakter yang hilang[3].



Gambar 2. 7 Algoritma *Levenshein*[3].

2.7 Sistem Operasi *Android*

Android merupakan sistem operasi mobile yang berbasis pada modifikasi sistem operasi linux. Perusahaan yang pertama kali mengembangkan sistem operasi mobile ini adalah Android.Inc. Nama perusahaan itulah yang akhirnya digunakan sebagai nama dari sistem operasi tersebut (Sistem operasi Android)[4].

2.5.1 Sejarah Sistem Operasi Android

Pada Tahun 2005 Google membeli Android Inc. sekaligus mengambil alih proses pengembangan sistem operasi android dan team developer Android. Google menginginkan Android untuk menjadi sistem operasi gratis atau open source. Android telah dikembangkan dan diupdate beberapa kali. Tabel dibawah ini memperlihatkan versi Android beserta distribusi android tersebut[4].

Versi	Code Name	API	distribusi
2.2	Froyo	8	0.4 %
2.3.3 - 2.3.7	GingerBread	10	6.4%
4.0.3 – 4.0.4	Ice Cream Sandwich	15	5.7%
4.1.x	Jelly Bean	16	16.5%
4.2.x		17	18.6 %
4.3		18	5.6 %
4.4	Kitkat	19	41.4%
5.0	Lollipop	21	5.0%
5.1		22	0.4%

Tabel 2. 1 Tabel distribusi android[4].

2.5.2 Fitur-fitur Android

Android merupakan sistem operasi mobile open source dan dapat dimanufaktur untuk dikustomisasi sehingga tidak ada konfigurasi yang pasti

dalam *hardware* dan softwarenya. Fitur-fitur Android digaris besarkan sebagai berikut[4]:

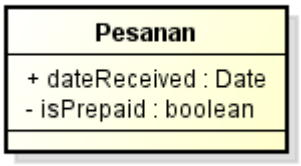
- **Dukungan *Hardware*** : Sensor accelerometer, Camera, Kompas Digital, sensor proximity, Dan GPS (Global Positioning System).
- **Multi Touch** : Mendukung layar dengan dukungan multi touch.
- **Multitasking** : Kemampuan untuk melaksanakan tugas secara bersamaan.

2.6 *Unified Modelling Language (UML)*

Unified Modeling Language atau *UML* adalah notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO). Pada kondisi yang nyata, pengertian *UML* menurut pendapat orang-orang saling berbeda[22].

2.6.1 *Class Diagram*

Class menggambarkan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat di antara mereka. *Class diagram* juga menunjukkan properti dan operasi sebuah *class* dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut. *Class diagram* menggunakan istilah fitur untuk menggambarkan property dan operasi sebuah *class*. Pada gambar …. menunjukkan properti *class diagram* sederhana. *dateRecieved* merupakan nama fungsi yang dimiliki class pesanan. *Date* menunjukkan tipe kembalian dari fungsi *dateRecieved*. Simbol positif (+) yang terdapat pada gambar *class diagram* tersebut menggambarkan jika fungsi *dateReceived* dapat diakses oleh kelas lain (bersifat public). Simbol *negative* (-) menggambarkan fungsi tersebut hanya dapat diakses oleh kelas itu sendiri[22].



Gambar 2. 8 Contoh sederhana *Class diagram*[22].

2.6.2 *Use case*

Use case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. Diagram *use case* menggambarkan interaksi antara pengguna sisttem dengan sistem itu sendiri. *Use case* terdiri atas scenario-skenario. Skenario meruapakan rangkaian langkah-langkah yang menjabarkan sebuah interaksi antara pengguna sistem dengan sistem. Dalam bahasan *use case* pengguna disebut sebagai aktor. Aktor dapat menggunakan banyak *use case*[22].

2.6.3 *Squance Diagram*

Squence diagram menjabarkan *behavior* sebuah skenario. Di dalam diagram *sequence* digambarkan urutan *behavior* yang dikerjakan dan pesan yang dikeluarkan oleh *behavior* tersebut. *Squence diagram* menunjukkan interaksi dengan menampilkan setiap partipisan dengan garis alir secara vertika dan pengurutan pesan dari atas ke bawah[22].

2.6.4 *Activity Diagram*

Activity diagram merupakan teknik untuk menggambarkan logika *procedural*, proses bisinis, dan jalan kerja. Dalam berberapa hal *activity diagram* memainkan peran yang mirip dengan diagram alir (flowchart), tetapi perbedaan prinsip antara diagram ini dan notasi diagram alir adalah diagram ini mendukung *behavior* yang parallel[22].

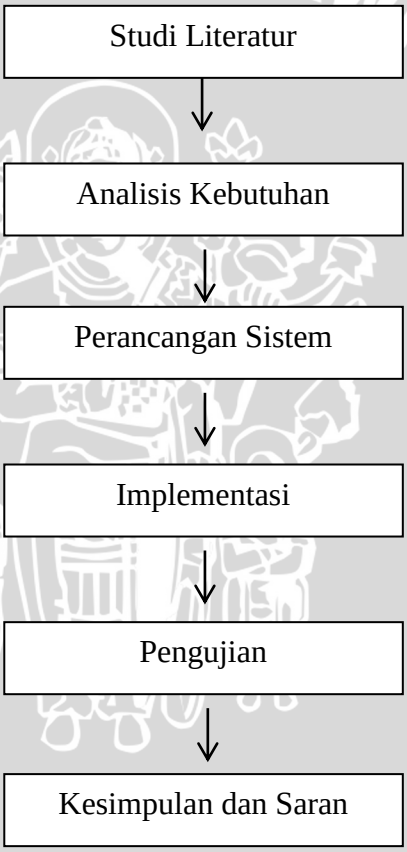
2.6.5 Entity Diagram Relationship(ERD)

Entity Relatioinship Diagram disingkat dengan ERD digunakan dalam analisis yang terstruktur dan pemodelan yang bersifat konseptual. ERD menerjemahkan masalah yang ada kedalam skema database. ERD melihat masalah-masalah terdiri dari entitas proses bisnis, hubungan proses bisnis tersebut, dan atribut yang menggambarkannya[26].

BAB III

METODOLOGI PENELITIAN

Pada bab ini akan dijelaskan langkah-langkah yang akan dilakukan dalam pengerjaan skripsi, yaitu studi literatur, analisis kebutuhan, perancangan system, implementasi, dan pengujian dari aplikasi perangkat lunak yang akan dibuat. Kesimpulan dan saran diikutsertakan sebagai catatan atas aplikasi dan kemungkinan atah pengembangna aplikasi selanjutnya. Berikut diagram alir runtutan pengerjaan skripsi ini:



Gambar 3. 1 Diagram Alir Metodologi Penelitian

Stuktur tahapan yang digambarkan diatas merupakan tahapan pengerjaan penelitian aplikasi kamera penerjemah *realtime* yang dimulai dari Studi literatur,

Analisis kebutuhan, Perancangan sistem, Implementasi, Pengujian, dan kesimpulan & Saran.

3.1 Studi Literature

Studi literature merupakan penjelasan yang mengenai dasar teori yang dipakai sebagai bahan pendukung dalam penelitian ini. Studi literature diperlukan mengingat pentingnya kualitas penelitian ini dipengaruhi oleh studi literature yang penulis pelajari.Teori-teori pendukung tersebut meliputi :

1. *Augmented Reality*
2. Sistem Operasi Android
3. Android SDK
4. Vuforia SDK : Text *Recognition*
5. OpenGL Android
6. OCR (*Optical Character Recognition*)
7. Algoritma Levenshtein
8. UML(*Unified Modelling Language*)

3.2 Analisa Kebutuhan

Kegiatan analisis kebutuhan perangkat lunak meliputi analisis spesifikasi perangkat lunak. Metode analisis menggunakan bahasa pemodelan UML *(Unified Modeling Language). Use Case Diagram* digunakan untuk mendeskripsikan kebutuhan-kebutuhan dan fungsionalitas sistem dari perspektif *user*. Analisis kebutuhan dilakukan dengan mengidentifikasi semua kebutuhan (*requirements*) sistem yang kemudian akan dimodelkan dalam diagram *use case.* Kebutuhan fungsional yang nantinya akan disediakan oleh aplikasi ini antara lain adalah:

1. Sistem mampu mendeteksi teks pada gambar secara realtime.
2. Sistem dapat menerjemahkan teks hasil deteksi dan menampilkannya secara realtime.
3. Aplikasi dapat mengkoreksi kata salah sehingga mampu menerjemahkan dengan benar ketika terdapat kesalahan kata tertentu.

3.3 Perancangan Sistem

Perancangan aplikasi dilakukan setelah semua kebutuhan system didapatkan melalui tahap analisis kebutuhan. Perancangan aplikasi dibuat berdasarkan Object Oriented Analysis dan Object Oriented Design menggunakan pemodelan UML(Unified Modeling Language).

Perancangan subsistem aplikasi dilakukan dengan mengidentifikasi kelas-kelas dan interface-interface yang dibutuhkan yang dimodelkan di dalam class diagram. Interaksi antara antar elemen (objek) dan dengan urutan waktu eksekusi objek dimodelkan kedalam squance diagram. Selain itu ada perancangan basis data yang menggunakan ER diagram perancangan model interface dengan menggunakan page flow.

3.4 Implementasi

Implementasi penelitian ini mengacu pada perancangan yang telah dibuat. Bahasa pemrograman yang digunakan dalam implementasi aplikasi ini adalah Java dengan memanfaatkan platform android. Dalam implementasi proses pengenalan teks aplikasi ini memanfaatkan Vuforia SDK. Sedangkan untuk datanya disimpan dengan format teks file yang diimplementasikan secara merata pada tiap aplikasi sehingga aplikasi ini tidak memerlukan backend/administrator. Aplikasi ini tidak memerlukan update data karena kata yang diterjemahkan dan hasil terjemahan bersifat konstan. Selanjutnya pada tahap akhir dilakukan implementasi simulasi pada *hardware* secara langsung.

3.5 Pengujian

Strategi yang digunakan dalam pengujian aplikasi ini adalah pengujian performa aplikasi berdasarkan tingkat kecerahan, jarak, dan sudut pandangan. Selain itu digunakan pengujian akurasi untuk mengetahui tingkat keberhasilkan aplikasi dalam mendeteksi dan menampilkan hasil terjemahan. Pengujian tingkat kecerahan digunakan untuk mengetahui tingkat kecerahan optimal yang dibutuhkan aplikasi untuk mendeteksi teks. Pengujian jarak digunakan aplikasi untuk mengetahui jarak optimal yang diperlukan untuk mendeteksi teks. Pengujian sudut pandangan digunakan untuk mengetahui

sudut pandang optimal yang diperlukan kamera aplikasi untuk mendeteksi teks. Selain itu juga diperlukan pengujian validasi untuk mengetahui kesesuaian hasil implementasi terhadap rancangan sistem. Dalam pengujian tingkat kecerahan, jarak optimal dan sudut pandangan akan menggunakan 20 data gambar yang dalam keadaan normal berhasil dideteksi. Pengujian akurasi menggunakan 35 data gambar yang belum tentu bisa dideteksi.

Pengujian tingkat kecerahan melakukan 10 variasi tiap gambar yaitu dengan mengubah tingkat kecerahan gambar normal menjadi 10%, 20%, 40%, 60%, 80%, -10%, -20%, -40%, -60% dan -80% dari kecerahan normal. Kemudian dari variasi tersebut aplikasi mendeteksi masing masing variasi tersebut untuk menghasilkan tingkat kecerahan optimal yang dibutuhkan aplikasi.

Pengujian jarak dilakukan dengan memvariasikan tiap data gambar dengan pemberian jarak dari aplikasi ke objek yang berbeda-beda. Jarak yang digunakan adalah 15 cm, 20cm, 25cm, 30cm, 35cm, 45cm, 50cm, 55cm, 100cm, 150cm, 200cm, 250cm, dan 300cm. Kemudian dari variasi tersebut aplikasi mendeteksi masing masing variasi tersebut untuk mengetahui jarak optimal yang dibutuhkan aplikasi untuk mengenali dan menampilkan hasil terjemahan.

Pengujian sudut pandangan dilakukan dengan memvariasikan tiap data gambar dengan pemberian sudut horizontal yang berbeda pada aplikasi untuk mendeteksi objek yang berbeda-beda. Sudut yang digunakan adalah 90˚, 80˚, 70˚, 60˚, 50˚, 40˚, 30˚, 20˚, dan 10 ˚. Kemudian dari variasi tersebut aplikasi mendeteksi masing masing variasi tersebut untuk mengetahui sudut pandangan optimal yang dibutuhkan aplikasi untuk mengenali dan menampilkan hasil terjemahan.

3.6 Kesimpulan dan Saran

Tahap kesimpulan dan saran hanya bisa dilakukan setelah tahap perancangan perangkat lunak, implamentasi perangkat lunak, dan pengujian perangkat lunak telah selesai dikerjakan. Kesimpulan berisi analisa terhadap sistem perangkat lunak dan hasil analisis dari pengujian yang telah dilakukan.

Setelah kesimpulan didapatkan maka tahap selanjutnya adalah pengambilan saran yang bermanfaat untuk memperbaiki celah dan kesalahan yang ada pada penelitian sehingga dapat menyempurnakan penelitian yang telah dibuat. Kemudian penelitian ini dapat dijadikan sebagai landasan ataupun refrensi dalam penelitian dikemudian hari.

BAB IV

ANALISIS DAN PERANCANGAN

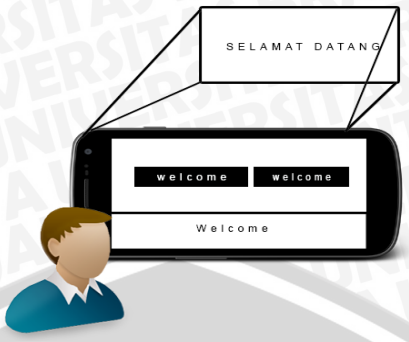
Pada bab ini dibahas tentang analisis dan perancangan perangkat lunak aplikasi kamus penerjemah secara *realtime* memanfaatkan teknologi *Augmented Reality* berbasis *Android*. Proses perancangan aplikasi ini meliputi dua tahap.Tahap pertama adalah analisis kebutuhan. Di dalam analisis kebutuhan terdapat berapa tahapan yaitu gambaran umum aplikasi, identifikasi aktor, analisis kebutuhan fungsional, kemudian dimodelkan kedalam *use case*. Sedangkan pada tahap perancangan terdiri dari perancangan arsitektural, perancangan *activity diagram*, perancangan *sequence diagram*, perancangan *class diagram* ,perancangan basis data dan perancangan *page flow*.

4.1 ANALISIS KEBUTUHAN

4.1.1 Gambaran Umum Perangkat Lunak

Konsep dari "Rancang Bangun Penampil berbasis *Augmented Reality* Aplikasi Kamus Kamera *Realtime* berbasis Android" adalah menampilkan hasil terjemahan teks bahasa indonesia ke bahasa inggris kedalam posisi teks yang terdeteksi secara *realtime* menggunakan teknologi *augmented reality*. Posisi teks didapatkan dari proses pengolahan citra digital.

Pada Prototype aplikasi ini akan digunakan sebuah *smartphone* Android dengan menggunakan kamera sebagai perangkat masukan citra yang berupa gambar video *realtime*. Output dari aplikasi ini berupa daerah yang terdeteksi dan hasil terjemahan dari daerah kata yang terdeteksi kemudian ditampilkan di layar android. Visualisasi dari gambaran umum perangkat lunak dapat dilihat pada Gambar 4.1.



Gambar 4. 1 Visualisasi Gambaran umum aplikasi

Pada system yang diusulkan ini, skenario sistem adalah sebagai berikut. User menempatkan posisi dari kamera sesuai dengan teks yang akan diterjemahkan. Kemudian aplikasi akan menampilkan hasil terjemahan teks secara *realtime*. Perangkat lunak ini memiliki fitur utama yaitu menerjemahkan teks bahasa Indonesia ke dalam bahasa inggris secara *realtime*.

4.1.2 Identifikasi Aktor

Pada tahap ini dilakukan proses identifikasi terhadap aktor-aktor yang berperan atau berinteraksi dengan aplikasi ini. Pada Tabel 4.1 memperlihatkan aktor-aktor yang terlibat berserta penjelasannya masing-masing.

Tabel 4. 1 Tabel identifikasi Aktor

Aktor	Deskripsi
Pengguna	Pengguna merupakan pengguna aplikasi yang bertujuan untuk menerjemahkan teks dari bahasa Indonesia ke bahasa inggris.

4.1.3 Analisis Kebutuhan Fungsional

Pada tabel berikut ditunjukkan daftar kebutuhan fungsional yang memuat fitur utama yang dimiliki oleh aplikasi tersebut.

Tabel 4. 2 Tabel kebutuhan Fungsional

Kode	**Kebutuhan Fungsional**	**Use Case**

F01	Aplikasi mampu mendeteksi teks bahasa Indonesia secara *realtime* menggunakan kamera	Mendeteksi Teks
F02	Aplikasi mampu menerjemahkan teks yang telah terdeteksi kedalam bahasa inggris	Menerjemahkan & menampilkan teks
F03	Aplikasi mampu mendeteksi pengejaan kata yang salah	Mengoreksi salah kata

Daftar hasil analisis kebutuhan fungsional dapat dilihat pada Tabel 4.2 yang memiliki dua fitur utama yang harus dipenuhi pada aplikasi ini.

4.1.4 Analisis Kebutuhan Non-Fungsional

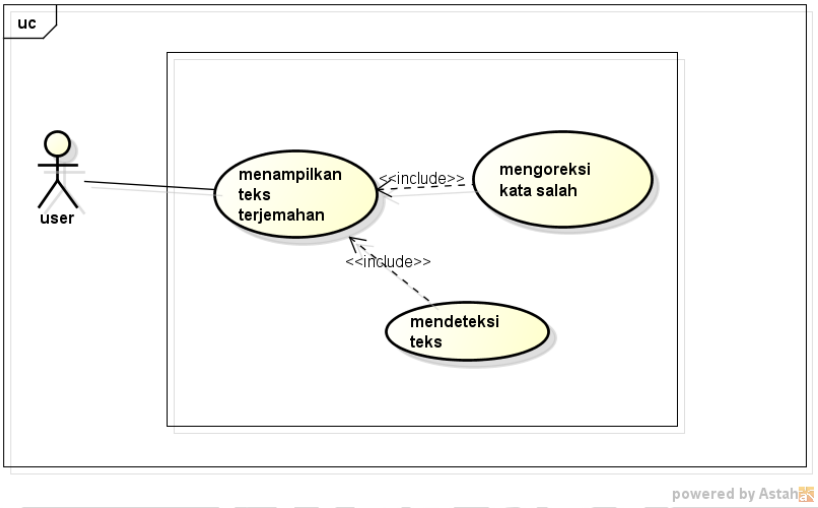
Analisis kebutuhan non-fungsional adalah analisis untuk mengetahui spesifikasi yang dibutuhkan oleh sistem. Ada beberapa parameter dan dekripsi kebutuhan yang akan digunakan dalam pengembangan, yaitu Compability dan Usatibility. Pada Tabel 4.3 menjelaskan kebutuhan non-fungsional yang dimiliki sistem yaitu Aplikasi dijalankan pada sistem operasi Android dengan kondisi offline. Kondisi offline tersebut berarti aplikasi tidak membutuhkan tambahan data diluar perangkat.

Tabel 4. 3 Spesifikasi Kebutuhan Non-Fungsional

Parameter	**Deskripsi Kebutuhan**
Compability	Aplikasi dijalankan pada sistem operasi Android dengan kondisi offline.

4.1.5 Diagram Use Case

Diagram use case merupakan diagram UML(Unified Modeling Language) yang menggambarkan fungsionalitas dari sebuah sistem. Gambar 4.2 menunjukkan use case dari aplikasi kamus penerjemah *realtime*. Pada use case tersebut digambarkan sistem memiliki aktor yaitu user (pengguna) yang mampu menerjemahkan teks dan menampilkan hasil terjemahan melalui sistem ini.



Gambar 4. 2 Diagram *use case* aplikasi kamus penerjemah *realtime* berbasis *augmented reality*.

4.1.6 Skenario Use Case

Tabel 4.3 merupakan penjabaran dari *use case* mendeteksi teks.

Tabel 4. 4 Skenario *use case* mendekteksi teks

Kode use case	F01
Nama *use case*	Mendekteksi teks
Aktor	*User*
Tujuan	Untuk mendapatkan teks dari gambar teks yang dipindai oleh kamera.
Deskripsi	Fitur yang ada dalam aplikasi yang berguna untuk memindai gambar teks yang akan diterjemahkan.
Pra-Kondisi	User mentrigger sistem untuk melakukan pemindaian melalui tombol yang disediakan
Pos-Kondisi	User akan mendapatkan bagian gambar teks yang mampu dikenali sebagai teks.
Alur-utama	Aktifitas

	1. User membuka aplikasi kamus penerjemah *realtime*. 2. Aplikasi akan menampilkan menu pindai teks yang berguna untuk memindai teks. 3. Aplikasi memindai gambar. 4. Aplikasi mengeluarkan hasil bagian gambar yang mampu dikenali sebagai teks.

Pada Tabel 4.4 menjelaskan proses aplikasi mendeteksi kata yang salah penulisan.

Tabel 4. 5 Skenario use case mengoreksi kata salah.

Kode *use case*	F03
Nama *use case*	Mengoreksi kata salah
Aktor	*User*
Tujuan	Untuk mendapatkan teks yang sesuai dengan dataset yang ada walaupun terjadi salah penulisan pada kata yang sebenarnya.
Deskripsi	Fitur yang ada dalam aplikasi berguna untuk mendeteksi kesalahan penulisan dan memperbaikinya.
Pra-Kondisi	User telah melakukan pemindaian teks dan teks telah terdeteksi pada suatu gambar tertentu. Dari hasil pemindaian tersebut teks yang terdeteksi sedang dalam kondisi salah penulisan pada kenyataanya.
Pos-Kondisi	User mendapatkan teks yang sesuai dengan dataset hasil.

Alur-utama	Aktifitas 1. User telah menjalankan fitur mendeteksi teks dan mendapatkan teks yang akan diterjemahkan. 2. Aplikasi akan mendeteksi kesalahan terjemahan menggunakan algoritma .... untuk menentukan terjadinya kesalahan penulisan teks. 3. Aplikasi mengembalikan nilai teks dalam kondisi benar.

Pada tabel berikutnya (Tabel 4.4) menjelaskan proses aplikasi dalam menerjemahkan dan menampilkan hasil terjemahan.

Tabel 4. 6 Skenario *use case* menerjemahkan dan menampilkan hasil terjamahan

Kode *use case*	F02
Nama *use case*	Menerjemahkan dan menampilkan teks hasil terjemahan
Aktor	*User*
Tujuan	Untuk mendapatkan hasil terjemahan teks yang terdeteksi pada proses deteksi teks.
Deskripsi	Fitur yang ada dalam aplikasi berguna untuk menerjemahkan teks.
Pra-Kondisi	User telah melakukan pemindaian teks dan teks telah terdeteksi pada suatu gambar tertentu.
Pos-Kondisi	User mendapatkan hasil terjemahan teks yang terdeteksi sebagai teks.
Alur-utama	Aktifitas 1. User telah menjalankan fitur mendeteksi teks dan mendapatkan teks yang akan diterjemahkan. 2. Aplikasi akan menerjemahkan teks

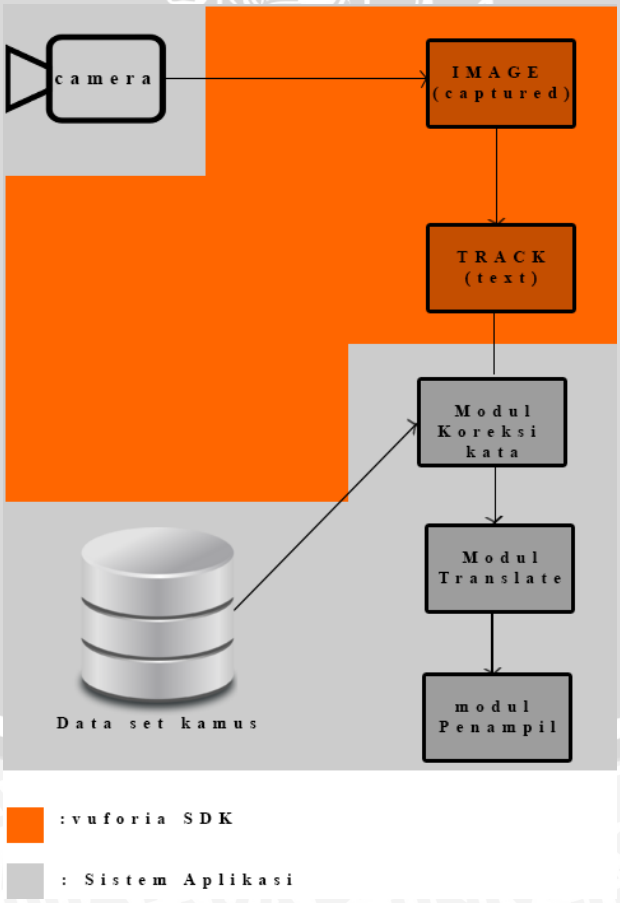
	hasil pemindaian sebelumnya. 3. Aplikasi menampilkan teks hasil terjemahan.

4.2 Perancangan

Pada tahap ini dilakukan proses perancangan sistem berdasarkan analisis kebutuhan yang dilakukan pada tahap sebelumnya. Pada tahap perancangan akan dimulai dengan merancang arsitektural sistem, *activity diagram*, *sequence diagram*, *class diagram*, basis data, dan *page flow*.

4.2.1 Perancangan Arsitektural

Perancangan arsitektural menjelaskan gambaran umum kinerja dari sistem yang dibuat secara keseluruhan. Berikut adalah rancangan arsitektural dari sistem yang akan dibuat.



Gambar 4. 3 Rancangan arsitektural sistem

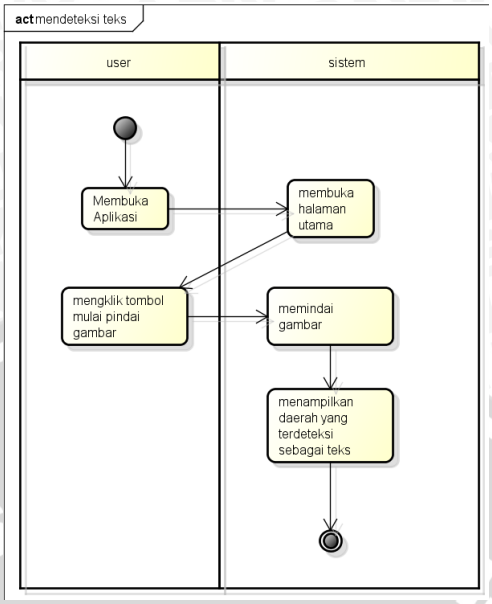
Pada Gambar 4.3 digambarkan rancangan arsitektural sistem yang dimulai dengan kamera menangkap objek gambar yang berisi teks yang secara *realtime*. Kemudian gambar-gambar yang telah ditangkap (*image captured*) dipindai dengan menggunakan modul *tracker text* pada *Vuvoria SDK*. Hasil pemindaian modul *tracker text* tesebut dimasukkan dalam suatu kondisi. Jika *tracker text* mampu mendekteksi adanya teks maka hasil *text* yang terdeteksi akan dikoreksi kedalam modul koreksi kata. Modul koreksi kata bertujuan untuk mengoreksi kata yang terdeteksi dan mengembalikan kata yang benar. Hasil teks keluaran modul koreksi kemudian diterjemahkan kedalam bahasa inggris dengan menggunakan modul translate. Jika modul tracker text tidak menemukan teks maka proses tersebut akan berhenti disitu.

4.2.2 Activity Diagram

Activity diagram merupakan representasi grafis (*diagram*) dari seluruh tahapan alur kerja. diagram ini dapat digunakan untuk menjelaskan proses bisnis dan alur kerja operasional secara langkah demi langkah dari komponen suatu sistem.

4.2.2.1 *Activity diagram* mendekteksi teks

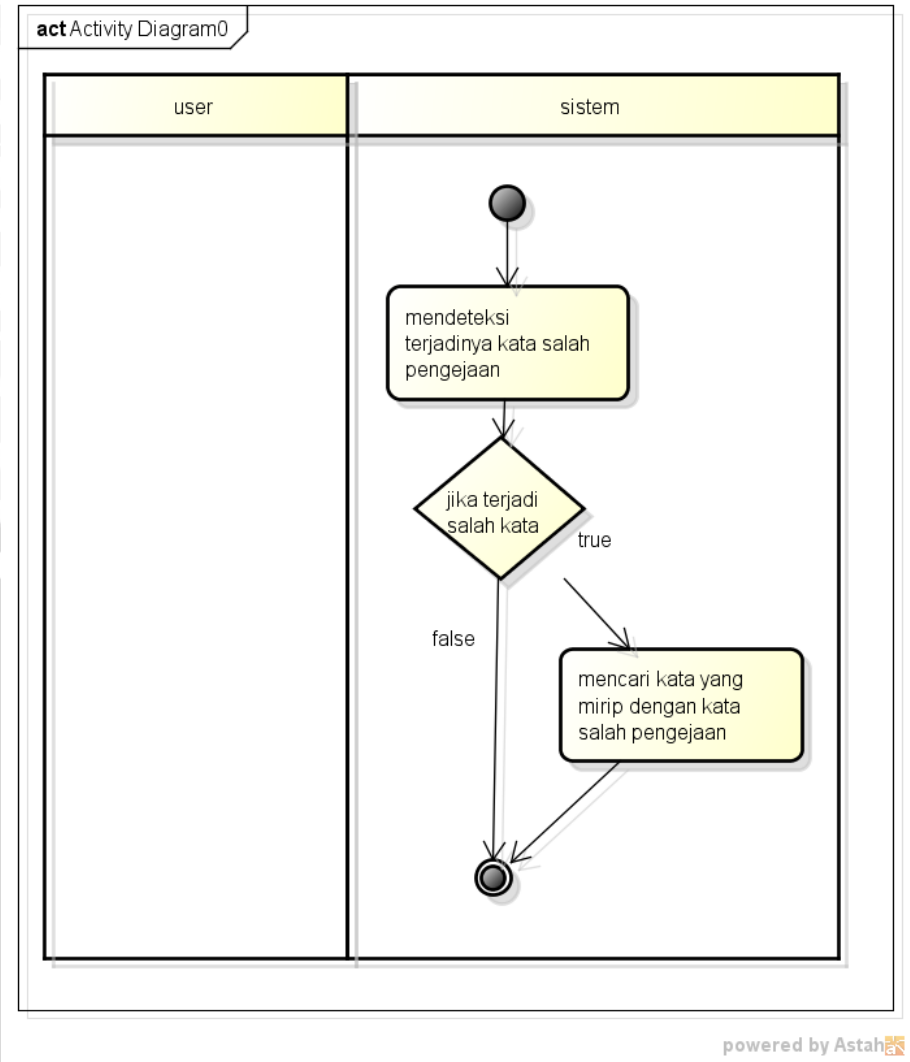
Pada Gambar 4.4, *activity diagram* mendekteksi teks dimulai dengan *user* membuka aplikasi. Sistem menampilkan halaman utama dari aplikasi yang berisi *menu about*, pindai gambar, dan keluar. User menekan tombol deteksi gambar, sistem akan memulai memindai gambar melalui kamera belakang *smartphone* secara *realtime*. Jika pada saat memindai gambar tersebut aplikasi menemukan kata-kata di dalamnya maka aplikasi akan menggambar objek kotak yang mengelilingi kata tersebut dari luar. Hasil dari aktivitas ini adalah daerah dari kata yang terdeteksi dan kata yang terdeteksi tersebut.



Gambar 4. 4 *Activity diagram* mendekteksi teks

4.2.2.2 *Activity Diagram* koreksi kata salah

Activity diagram yang sesuai dengan Gambar 4.5 merupakan proses untuk mengenali kata yang salah atau disebut dengan *typo* dan mengembalikan kata lebih tepat. *Activity diagram* ini merupakan lanjutan dari proses mendeteksi teks pada proses sebelumnya. Proses pada *activity diagram* mengoreksi kata salah dimulai dengan mendeteksi terjadinya salah pengejaan kata dan proses mendeteksi kata salah selesai. Jika ditemukan kata yang salah pengejaan maka aplikasi akan mencari kata yang mirip dengan kata salah tersebut. Jika tidak terjadi salah pengejaan kata maka proses akan dinyatakan selesai.

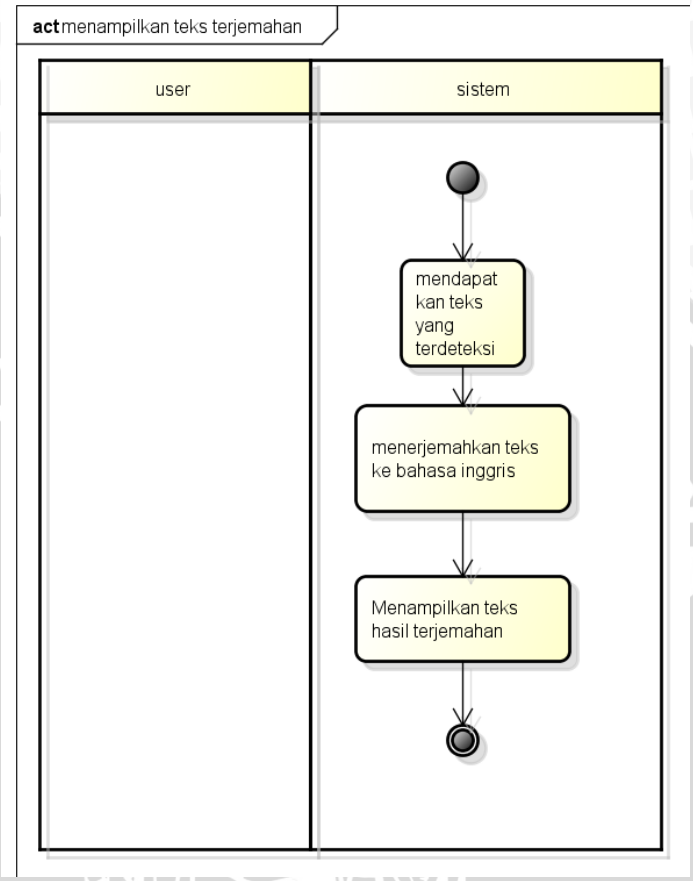


Gambar 4. 5 *Activity Diagram* koreksi kata salah

4.2.2.3 *Activity diagram* menerjemahkan teks

Dalam Gambar 4.5, *activity diagram* menerjemahkan teks merupakan lanjutan proses setelah mendekteksi teks yang dilakukan sebelumnya. Pada tahap mendekteksi teks hasil yang didapatkan adalah daerah dari kata yang terdeteksi dan kata yang terdeteksi tersebut. *Activity diagram* menerjemahkan teks tidak akan dieksekusi sebelum *activity* mendekteksi teks dieksekusi terlebih dahulu. Pada *activity diagram* menerjemahkan teks dimulai dengan mengambil kata yang terdeteksi. Kemudian kata yang yang terdeteksi tersebut diterjemahkan dengan modul penerjemah yang dibuat tersendiri. Jika modul penerjemah mampu menerjemahkan teks yang terdeteksi dari bahasa indonesia

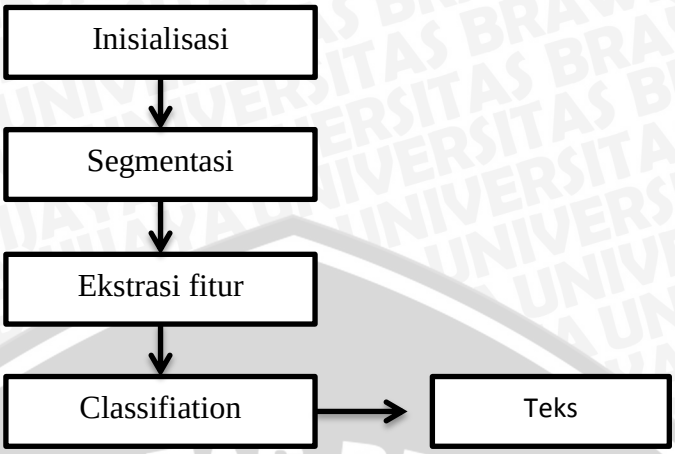
kedalam bahasa inggris maka aplikasi akan menampilkan hasil terjemahan tersebut.



Gambar 4. 6 *Activity diagram* menerjemahkan teks

4.2.3 Algoritma Mendeteksi teks

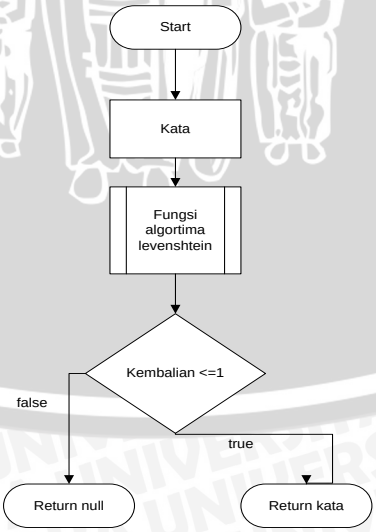
Pada pengenalan teks dikenal 4 proses umum yaitu inisialisasi, segmentasi, ektrasi fitur, dan classification. Pada Gambar 4.7 dijelaskan proses pengenalan teks diawali dengan inisialisasi. Proses inisialisasi yang dimaksud adalah mempersiapkan citra digital yang akan dideteksi. Pada proses segmentasi dilakukan proses pemecahan gambar teks menjadi gambar tiap karakter. Selanjutnya pada proses Ekstrasi fitur dilakukan penyimpanan hasil segmentasi yang mana hasil segmentasi tiap karakter disimbolkan dengan simbol yang unik. Sedangkan pada proses Classification dilakukan proses mencocokan semua fitur ektrasi yang disimpan dengan segmen uji yang ada pada program. Kemudian setelah itu mengeluarkan hasil teks[20].



Gambar 4. 7 Algoritma Mendeteksi Kata

4.2.4 Algoritma Pengoreksian kata salah

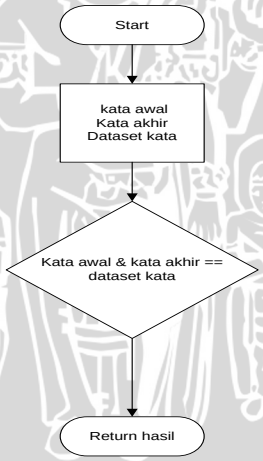
Algortima pengoreksian kata salah berfungsi untuk mengoreksi kata yang terdeteksi pada proses pendeteksian teks. Kata salah merupakan kata yang kehilangan satu huruf pada suatu kata. Kata yang terdapat kesalahan penulisan berketentuan maksimal satu huruf yang hilang. Algortima yang digunakan untuk mengkoreksi kata salah adalah algoritma Levenshtein. Pada Gambar 4.8 terdapat masukkan kata yang kemudian dideteksi oleh algoritma levenshtein jika menghasilkan kembalian kurang dari 1 maka kata tersebut dianggap benar, jika sebaliknya lebih dari 1 maka akan menghasilkan kembalian kosong.



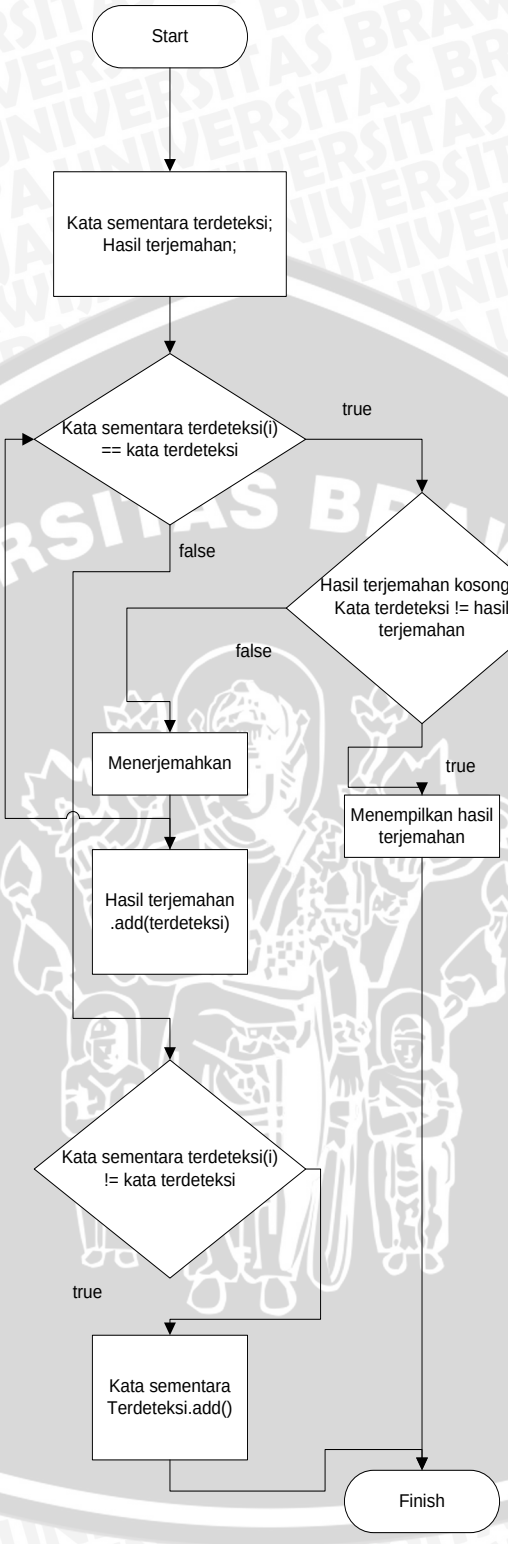
Gambar 4. 8 Algoritma Koreksi Kata

4.2.5 Algoritma Menerjemahkan Teks

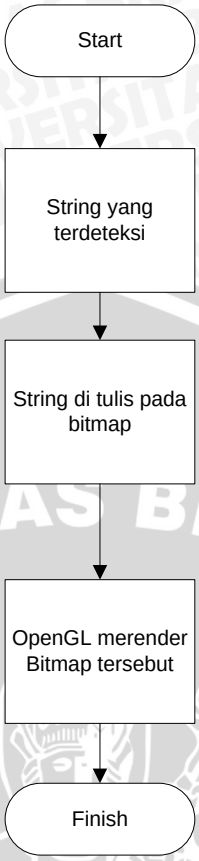
Gambar 4.10 merupakan diagram alir untuk menyimpan hasil terjemahan. Agoritma yang digunakan dalam penerjemahan teks adalah algoritma brute force dengan mencocokan string kata yang terdeteksi dengan dataset terjemahan. Semua kata yang terdeteksi akan dimasukkan pada data temporary kemudian data temporary tersebut akan diterjemahkan sesuai dengan kata yang terdeteksi. Hasil terjemahan disimpan pada variable sementara agar kata yang sama terdeteksi kembali tidak perlu ke proses menerjemahkan cukup mencocokan data dengan variable sementara terjemahan. Gambar 4.9 merupakan diagram alir algoritma menerjemahkan dua kata. Setelah kata berhasil di terjemahkan hasil terjemahan tersebut disimpan pada variable sementara. Dari variable sementara tersebut hasil terjemahan ditampilkan sesuai algoritma pada Gambar 4.11. Algoritma menampilkan hasil terjemahan dimulai dengan teks hasil terjemahan yang kemudian ditulis kedalam format bitmap Sehingga OpenGL mampu merender teks melalui bitmap tersebut.



Gambar 4. 9 Algoritma Menerjemahkan teks



Gambar 4. 10 Algoritma Menambahkan Hasil Terjemahan



Gambar 4. 11 Algoritma Menampilkan Hasil Terjemahan

4.2.6 Penggunaan *Library Vuvoria* pendekteksi teks

Dalam merancang aplikasi modul penampil kamus penerjemah berbasis *augmented reality* dibutuhkan *library* Vuforia untuk menangani proses pada hasil analisis kebutuhan sistem. Seperti yang terdapat pada Tabel 4.5 yang menerangkan penggunaan library Vuforia pada pendeteksian teks. Dalam sistem ini *library* Vuforia berperan untuk mendeteksi teks secara *realtime*. *Library* Vuforia memiliki banyak kelas yang digunakan dibutuhkan dalam mengimplementasikan aplikasi ini.

Tabel 4. 7 Tabel penggunaan library Vuforia

No.	Nama Kelas	Fitur Umum	Fungsi yang dibutuhkan
1	*Renderer*	merupakan	➢ *Begin*()

		salah satu kelas dalam library yang bertanggung jawab dalam memenuhi tugas-tugas yang terkait AR seperti merender *background* dan objek 3D	- menandai awal render untuk *frame* dan mengembalikan *objek state*. ➢ *End*() -menandai akhir rendering untuk *frame* yang sedang berjalan. ➢ *drawVideoBackground*() -menggambar latar belakang berupa *video*. Fungsi ini hanya boleh dipanggil antara fungsi *begin*() dan *end*().
2.	*State*	kelas yang bertanggung jawab untuk mendapatkan jumlah *tracable objeck* yang terdeteksi	➢ *getNumTrackableResults*() - Mengembalikan jumlah dari objek *Trackable* yang sedang dilacak. ➢ *getTrackableResult*() - Menyediakan akses atas objek *TrackableResult*
3.	*TrackableResult*	kelas yang mewakili keadaan objek yang dilacak yang ditemukan dalam *frame* tertentu	➢ *isOfType*() - Mengecek apakah tipe dari instan *TrackableResult* telah sesuai. ➢ *getPose*() - Mengembalikan pose

			matrik saat ini sesuai dengan *row-major order*.
4.	*Vec2F*	merupakan kelas yang merepresentasikan *vector 2D* dari variabel *float*.	➢ *getData*() - mengembalikan nilai dari *vector* 2D.
5.	*WordResult*	kelas yang menyimpan kelas *word* yang diambil dari objek *TrackableResult*	➢ *getTrackable*() - Mengembalikan objek *Trackable* yang merepresentasikan hasil *tracking*. ➢ *getObb*() - Mengembalikan kotak yang berorientasi pada ruang gambar dari kata. ➢ *getClassType*() - Mengembalikan objek bertipe *TrackableResult*.
6.	*Word*	merepresentasikan satu elemen dari penulisan	➢ *getSize*() - Mengembalikan ukuran (panjang dan lebar) kotak dari kata. ➢ *getStringU*() - Mengembalikan nilai teks yang berhasil dilacak.

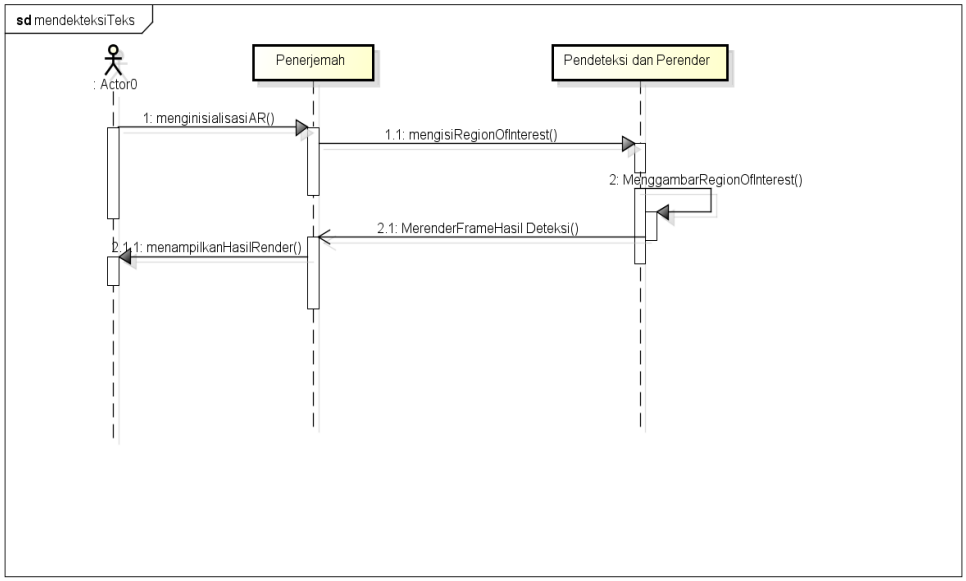
7.	*Obb2D*	merepresentasikan kotak *2D* yang terikat	➢ *getCenter*() - Mengembalikan nilai pusat dari kotak teks.
8.	*Matrix44F*	merupakan kelas yang merepresentasikan matrik dengan 4 baris dan 4 kolom yang berisikan variabel *float.*	➢ *getData*() - mengembalikan *array* dari *matrik* objek.

4.2.7 Squence diagram

Diagram skuensial menggambarkan perilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirim dan diterima antar objek. Oleh karena itu untuk menggambar diagram *skuensial* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

4.2.4.1 Diagram Sekuensial mendeteksi teks

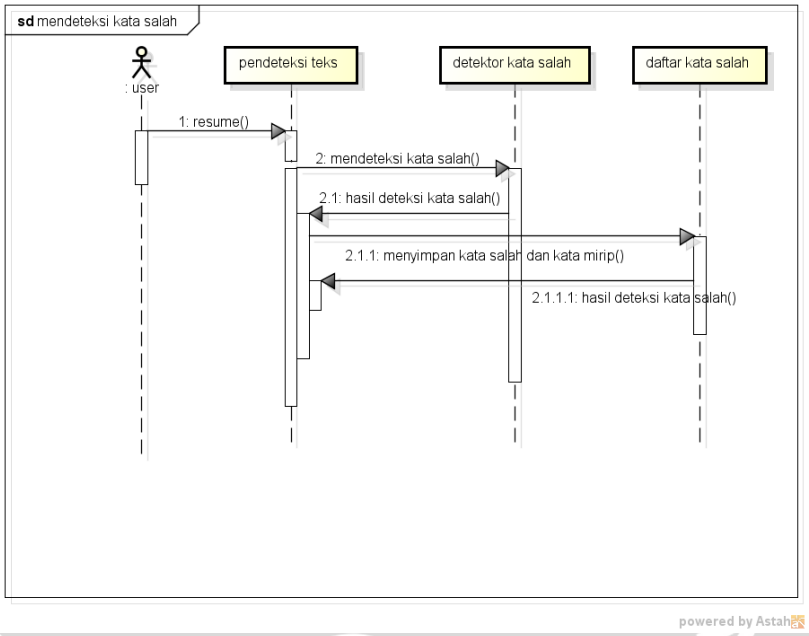
Pada Gambar 4.12, *diagram skuensial* mendeteksi teks digunakan untuk *user* mendeteksi teks. *User* menekan tombol mulai memindai pada aplikasi. Aplikasi akan menginialisasi *augmented reality* dan mengset *Region Of Interest*(*ROI*). *Region Of Interest* merupakan area atau daerah yang menangani pendektesian teks artinya pendeteksian teks hanya dapat dilakukan pada area *ROI* tersebut. Kemudian aplikasi merender area *ROI*. Kemudian *merenderFrameHasilDeteksi*() dipanggil untuk menggambar *frame* teks yang terdeteksi yang kemudian ditampilkan kepada *user*.



Gambar 4. 12 *Diagram* skuensial mendeteksi teks

4.2.4.2 Diagram sekuensial mengoreksi kata salah

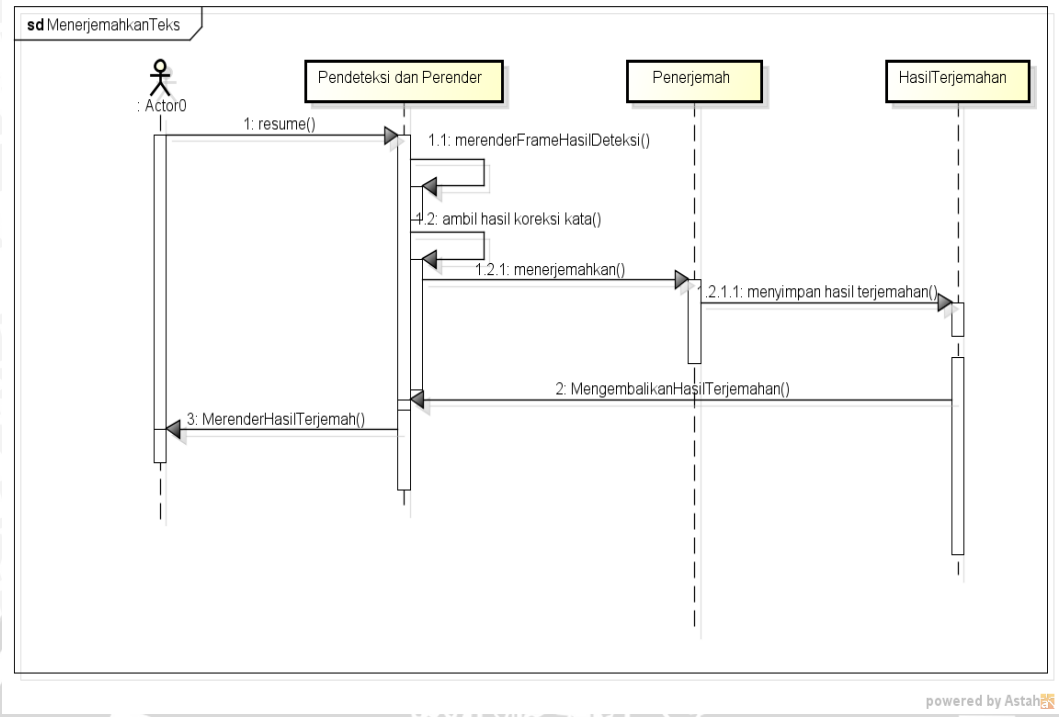
Diagram skuensial mengoreksi kata salah digambarkan pada Gambar 4.13. Diagram ini dimulai dengan mendeteksi kata salah() yang berfungsi mendeteksi kesalahan penulisan pada kata yang terdeteksi oleh pengenalan teks aplikasi. Hasil dari deteksi kata salah dikembalikan pada modul pendeteksi teks yang kemudian disimpan didalam modul daftar kata yang salah.



Gambar 4. 13 Diagram skuensial mengoreksi kata salah**.**

4.2.4.3 *Diagram* sekuensial menerjemahkan teks

Pada Gambar 4.14, *diagram* skuensial menerjemahkan teks yang digunakan user untuk menerjemahkan teks yang terdeteksi. *Diagram* ini melanjutkan *diagram* skuensial mendeteksi teks. *Diagram* ini dimulai dengan merender *frame* teks yang terdeteksi melaui fungsi *merenderFrameHasilDeteksi*() kemudian memencocokan teks yang terdeteksi dengan kata-kata yang diinginkan dengan fungsi *mengecekKataTerdeteksi*(). Jika teks yang terdeteksi merupakan teks yang diinginkan untuk diterjemahkan maka teks tersebut akan diterjemahkan melalui fungsi *menerjemahkan*(). Kemudian kata yang telah diterjemahkan ditampilkan kepada user secara *realtime*.



Gambar 4. 14 Diagram skuensial menerjemahkan teks

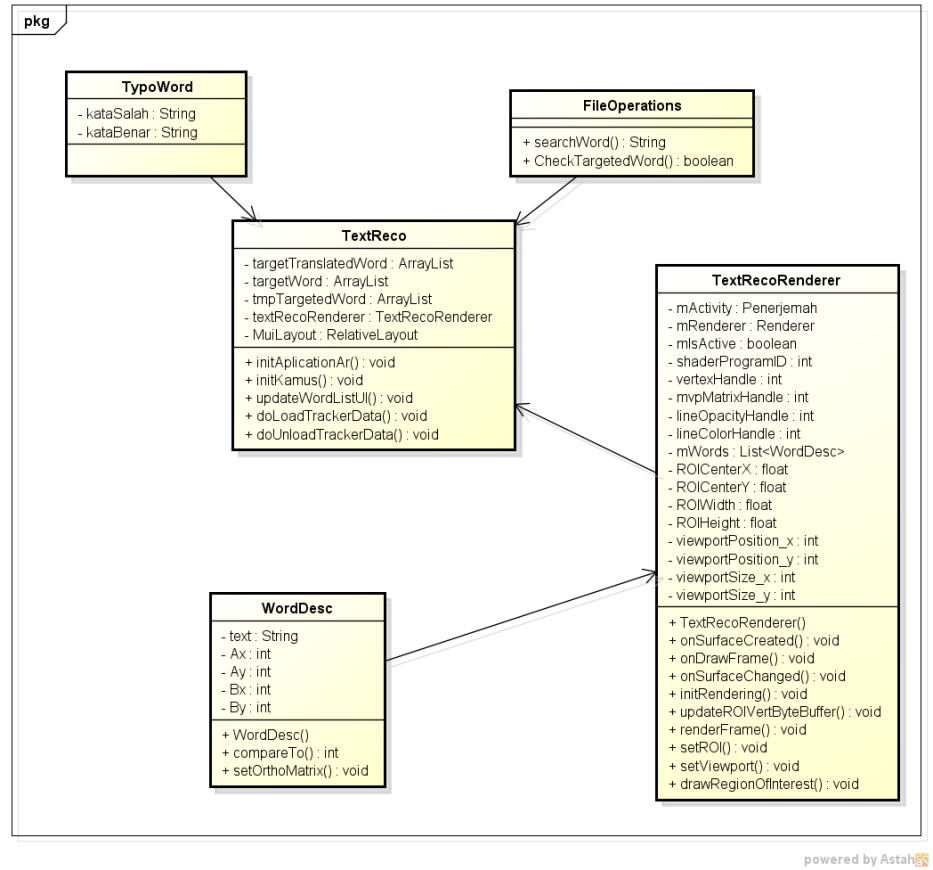
4.2.8 Class Diagram

Class diagram merupakan *diagram* yang menggambarkan pemodelan elemen-elemen *class* yang memberntuk sebuah sistem. *Class diagram* bisa didapat dengan menganalisis secara detail *usecase* yang dimodelkan.

4.2.5.1 *Class diagram* pengguna

Class diagram pengguna digunakan untuk memberikan gambaran pemodelan elemen perangkat lunak yang dibutuhkan oleh pengguna dalam menjalankan sistem. Pada Gambar 4.15 class diagram pengguna, *class textRecoRenderer* digunakan untuk merender *frame text* yang terdeteksi, merender *Region Of Interest*, mendeteksi teks,dan menyimpan teks yang dideteksi secara *temporary*. *Class WordDesc* digunakan untuk menyimpan kata yang akan dideteksi. *Class TranslatedWords* merupakan *class* yang digunakan untuk menyimpan kata yang telah diterjemahkan. *Class FileOperation* merupakan *class* yang digunakan untuk memvalidasi kata yang terdeteksi untuk diterjemahkan dan mencari terjemahan kata-kata yang terdeteksi. *Class TextReco* merupakan *class* utama dari semua kelas karena

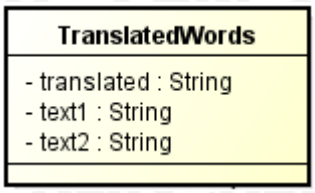
berawal dari *class TextReco* semua *class* dipanggil. *TextReco* memiliki *activity* dari aplikasi. Pada *class TextReco* terdapat inialisasi *AR*, inialisasi kamus, meload dataset *wordlist*, dan memanggil objek dari *TextRecoRenderer*.



Gambar 4. 15 Class Diagram Pengguna

- *Class TranslatedWords*

  Gambar 4.16 menunujukan *diagram class TranslatedWord* yang digunakan untuk menyimpan rangkaian kata yang telah diterjemahkan. Jika ada kata yang telah diterjemahkan sebelumnya maka aplikasi tidak perlu menerjemahkan lagi. Aplikasi akan mengambil hasil terjemahan pada objek dari *class TranslatedWords*.



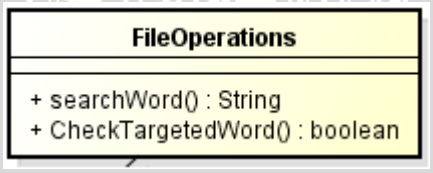
Gambar 4. 16 *Class diagram TranslatedWords*

Tabel 4. 8 Struktur *class TranslatedWords*

Nama Class : *TranslatedWords*
Deskripsi : *Class TranslatedWord* merupakan *class* yang digunakan untuk menyimpan rangkaian kata yang telah diterjemahkan.

- *Class FileOperation*

  Class FileOperation ditunjukkan dengan Gambar 4.17 merupakan *class* yang digunakan untuk memvalidasi kata yang terdeteksi untuk diterjemahkan dan mencari terjemahan dari kata-kata yang terdeteksi.



Gambar 4. 17 *Class diagram FileOperation*

Tabel 4. 9 Struktur *class TranslatedWord*

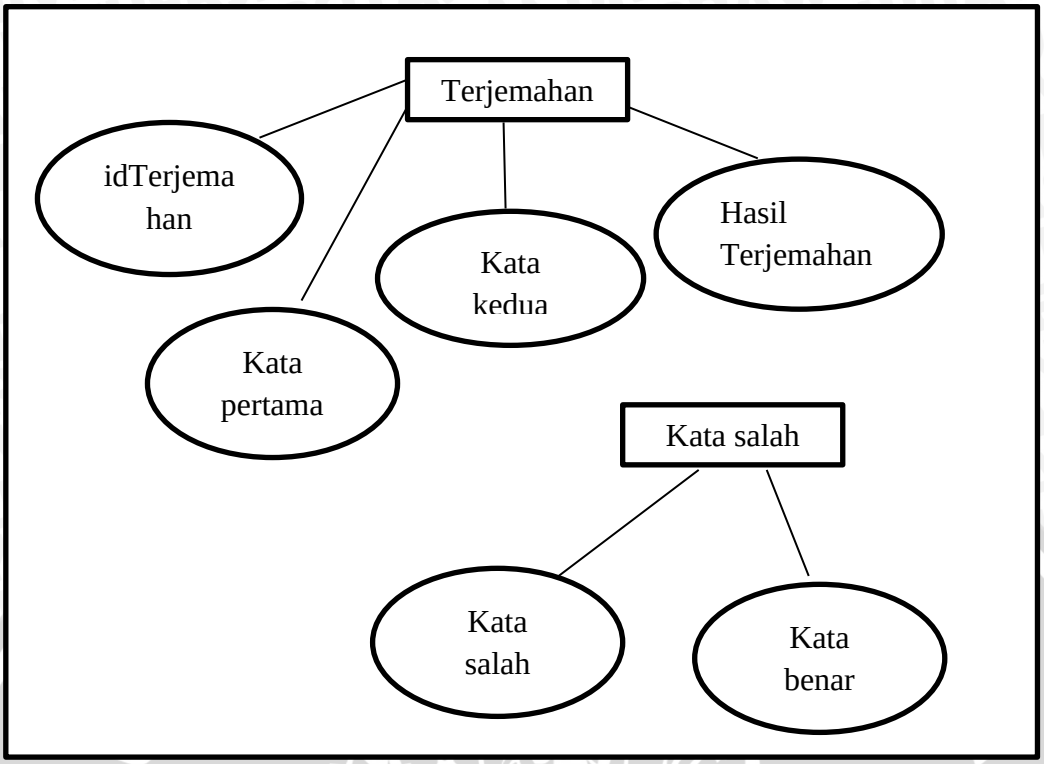
Nama *Class* : *FileOperation*
Deskripsi :

Class FileOperation merupakan *class* yang digunakan untuk mencari kata yang sesuai dengan tujuan dan mencari terjemahan dari kata-kata yang terdeteksi.
Nama *Method*: *SearchWord*() **Fungsi :** *Method* ini berfungsi untuk menerjemahkan kata-kata yang terdeteksi dengan mencocokan kata-kata yang terdeteksi dengan kata yang ada pada dataset.
Nama *Method*: *CheckTargetedWord*() **Fungsi :** Method ini berfungsi untuk memvalidasi kata yang terdeteksi untuk diterjemahkan.

4.2.9 Perancangan Basis Data

Perancangan basis data digunakan untuk merancang basis data yang akan dibuat agar masukan dan keluaran program sesuai dengan hasil analisis kebutuhan. Perancangan basis data menggunakan hasil dari proses analisis kebutuhan sebagai acuan dalam merancang basis data. Perancangan basis data dapat diamati pada Gambar 4.18.

Pada Gambar 4.18, terdapat hanya dua tabel yaitu, tabel terjemahan yang digunakan untuk mengelola daftar kata yang dapat dideteksi dan daftar kata yang akan diterjemahkan (kata pertama dan kata kedu). Sedangkan atribut hasilTerjemahan digunakan untuk meyimpan hasil hasil terjemahan rangkaian kata pertama dan kata kedua. Pada tabel kata salah digunakan untuk mengelola daftar kata yang salah penulisan. Attribut kata salah terdiri dari kata salah (yang merepresentasikan kata yang terjadi kesalahan penulisan) dan kata benar (yang merepresentasikan kata yang benar).

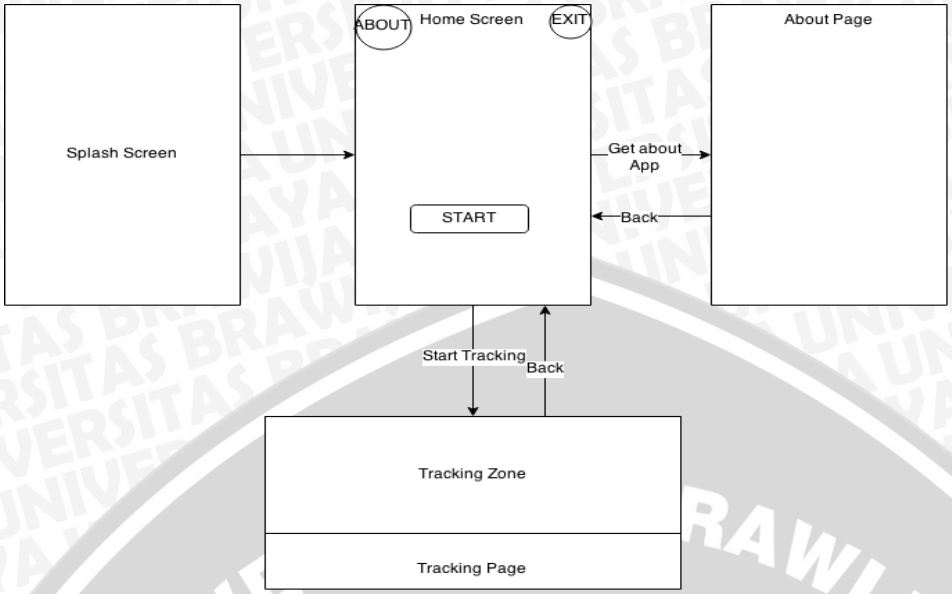


Gambar 4. 18 *Entity Diagram* sistem

4.2.10 Perancangan *Page Flow*

Pada perancangan *page flow* dijelaskan alur yang terdapat pada antarmuka aplikasi. Dalam perancangan *page flow* harus mengacu hasil proses analisis kebutuhan. Perancangan *page flow diagram* dapat diamati pada Gambar 4.19.

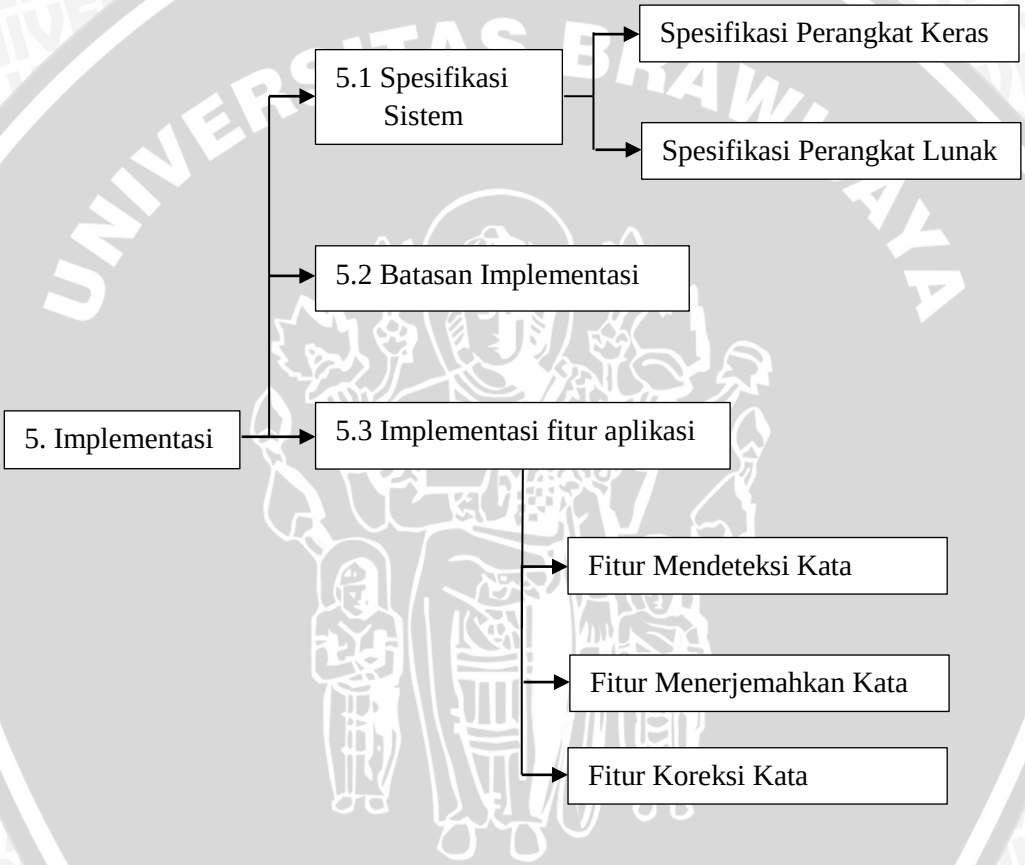
Pada Gambar 4.9, merupakan page flow diagram yang diawali dengan halaman *Splash Screen* yang menampilkan gambar pada awal aplikasi dijalankan. Pada halaman *home* memiliki fitur menutup aplikasi (*exit*), membuka halaman about (About), dan memulai melacak gambar secara *realtime*(START). Halaman *home* memanggil fungsi getAboutApp untuk membuka halaman about. Halaman about akan menampilkan deskripsi tentang aplikasi dan *developer* aplikasi ini. Halaman *tracking* dibuka melalui startTracking. Pada halaman *tracking* dilakukan pelacakan teks pada kondisi *realtime* melalui kamera *smartphone*. Pelacakan teks akan berfungsi jika gambar atau objek berada pada layar *Tracking Zone*.



Gambar 4. 19 *Page Flow Diagram* sistem

BAB V
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi aplikasi berdasarkan hasil yang telah didapatkan dari perancangan. Pembahasan terdiri dari penjelasan tentang spesifikasi sistem, batasan-batasan dalam implementasi, implementasi fitur aplikasi. Pohon implementasi ditunjukkan pada Gambar 5.1.



Gambar 5. 1 Pohon Implementasi

5.1 Spesifikasi Sistem

Implementasi rancang bangun penampil aplikasi kamus *augmented reality*, dikembangkan di dalam lingkungan yang terdiri dari lingkungan sistem perangkat lunak dan perangkat keras.

5.1.1 Spesifikasi Linkungan Sistem Perangkat Lunak

Spesifikasi lingkungan perangkat lunak yang digunakan untuk proses pengembangan ditunjukkan pada Tabel 5.1.

Tabel 5. 1 Spesifikasi Perangkat Lunak

ASUSA43SD	
Sistem Operasi	Windows 8.1 64-bit
Bahasa Pemrograman	*JAVA* dan *XML*
Library	Vuforia
IDE (*Integrated Development Environment*)	*Eclipse*
OS mobile	Android OS v4.0

5.1.2 Spesifikasi Lingkungan Sistem Perangkat Keras

Tabel 5.2 menjelaskan spesifikasi lingkungan perangkat keras yang digunakan pada proses pengembangan.

Tabel 5. 2 Spesifikasi Perangkat Keras

Lenovo K910L	
Prosessor	*Qualcomm Snapdragon 800 2,2GHz quad-core*
RAM(*Random Access Memory*)	1GB
Layar	IPS 5,5 inci *Full HD* (1920 x 1080 *pixel*), 400 *ppi*, *Corning Gorilla Glass* 3
Kamera	Belakang 13 *megapixel autofocus* dengan LED, depan 5 *megapixel*

5.2 Batasan – Batasan Implementasi

Beberapa batasan dalam mengimplementasikan rancang bangun penampil berbasis *augmented reality* pada kamus kamera *realtime* sebagai berikut :

1. Perangkat lunak ini merupakan aplikasi *android mobile native.*
2. Dataset teks yang digunakan untuk mengenali kata adalah *wordlist*
3. Aplikasi ini menggunakan dataset untuk menerjemahkan teks.

5.3 Implementasi *Use Case*

Pada penulisan skripsi ini hanya dicantumkan algortima dari proses inti yang diimplementasikan berdasarkan fitur yang dirancang yaitu mendeteksi teks, menerjemahkan teks,dan mengkoreksi kata. Implementasi algortima akan direpresentasikan dengan bahasa *java*.

5.3.1 Implementasi Fitur Mendeteksi Kata

Implementasi mendeteksi kata .Pada Source Code 5.1 menggambarkan proses mendeteksi kata pada gambar bergerak yang terdeteksi. Penjelasan algoritma proses mendeteksi teks :

1. Baris 1-26 menangkap gambar dengan video kamera.
2. Baris 27-68 digunakan untuk mendeteksi kata pada gambar dan mendapatkan atributnya.

Baris	Source Code

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
	State state = mRenderer.begin();
	mRenderer.drawVideoBackground();
	GLES20.glEnable(GLES20.GL_DEPTH_TEST);
	
	GLES20.glEnable(GLES20.GL_CULL_FACE);
	GLES20.glCullFace(GLES20.GL_BACK);
	if
	(Renderer.getInstance().getVideoBackgroundConfig().getReflection() ==
	VIDEO_BACKGROUND_REFLECTION.VIDEO_BACKGROUND_REFLECTION_ON)
	{
	GLES20.glFrontFace(GLES20.GL_CW); // Front camera
	} else
	{
	//GLES20.glFrontFace(GLES20.GL_CW); // Front camera
	GLES20.glFrontFace(GLES20.GL_CCW); // Back camera
	}
	//GLES20.glColorMask(true, false, true, true);;
	
	// enable blending to support transparency
	GLES20.glEnable(GLES20.GL_BLEND);
	//GLES20.glBlendFunc(GLES20.GL_SRC_ALPHA,
	// GLES20.GL_ONE_MINUS_CONSTANT_ALPHA);
	
	// clear words list
	mWords.clear();
	// did we find any trackables this frame?

```
28          for (int tIdx = 0; tIdx <
29  state.getNumTrackableResults(); tIdx++)
30          {
31
32              // get the trackable
33              TrackableResult result =
34  state.getTrackableResult(tIdx);
35
36              Vec2F wordBoxSize = null;
37
38              if (result.isOfType(WordResult.getClassType()))
39              {
40                  WordResult wordResult = (WordResult) result;
41                  Word word = (Word)
42  wordResult.getTrackable();
43                  Obb2D obb = wordResult.getObb();
44                  wordBoxSize = word.getSize();
45                  String wordU = word.getStringU();
46
47                  if (wordU != null)
48                  {
49                      float wordx = -
50  obb.getCenter().getData()[1];
51                      float wordy =
52  obb.getCenter().getData()[0];
53
54                      if (mWords.size() < MAX_NB_WORDS)
55                      {
56                          mWords.add(new WordDesc(wordU,
57                              (int) (wordx -
58  wordBoxSize.getData()[0] / 2),
59                              (int) (wordy -
60  wordBoxSize.getData()[1] / 2),
61                              (int) (wordx +
62  wordBoxSize.getData()[0] / 2),
63                              (int) (wordy +
64  wordBoxSize.getData()[1] / 2)));
65                      }
66
67                  }
68              } else
69              {
            Log.d(LOGTAG, "Unexpected Detection : " +
    result.getType());
            continue;
                }
```

Source Code 5. 1 Fitur Mendeteksi Kata

5.3.2 Implementasi Fitur Menerjemahkan teks

Dalam pengimplem.Pada Source Code 5.2 menggambarkan proses menerjemahkan kata yang telah dideteksi pada fitur mendeteksi kata.

Baris	Source Code

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
	`public static String searchWord(String text1,String text2,Context context,StringBuilder sb) throws IOException{`
	
	`if(!sb.toString().contains(text1+'?'+text2+"\t")&&!sb.toString().contains(text2+'?'+text1+"\t"))`
	`return null;`
	`String[] tmpResult = sb.toString().split(text1+"\\?"+text2+"\\t");`
	
	`try{`
	`//s = '?'+s;`
	`tmpResult = tmpResult[1].split("\\n");`
	`//tmpResult = tmpResult[0].split("\\t");`
	`return tmpResult[0];`
	
	`}catch (Exception e){//Catch exception if any`
	`Log.e("ERROR", "XXXXXXsearchword:---> " + e.toString());`
	`tmpResult = sb.toString().split(text2+"\\?"+text1+"\\t");`
	`tmpResult = tmpResult[1].split("\\n");`
	`return tmpResult[0];`
	`}`
	`}`

Source Code 5. 2 Fitur Menerjemahkan Teks

Gambar 5.2 bagian kiri merupakan kondisi aplikasi akan menangkap gambar untuk dideteksi sedangkan bagian kanan merepresentasikan hasil terjemahan dari teks pada gambar tersebut.

Gambar 5. 2 Tampilan dari Fitur Menerjemahkan teks

5.3.3 Implementasi Fitur Mengkoreksi Kata

Pada Source Code 5.3 menggambarkan proses mengkoreksi kata yang telah dideteksi pada fitur mendeteksi kata. Penjelasan algoritma proses menerjemahkan teks :

1. Baris 2-3 digunakan untuk mengecek isi daftar kata.
2. Baris 4-15 digunakan untuk mencocokkan kata yang salah dan benar. Jika terdapat 1 kesalahan huruf maka akan mengembalikan kembalian true sebaliknya jika terdapat kesalahan lebih dari 1 hu
3. Baris 24-40 merupakan pengimplementasian algoritma levenshein untuk mengetahui selisih perbedaan antara dua kata.

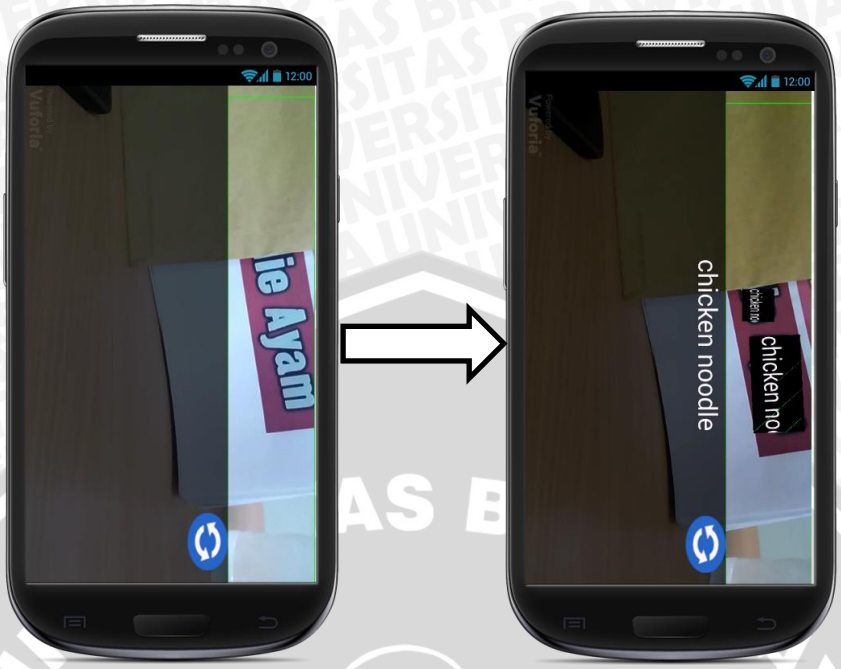
Baris	Source Code
1	`public static String checkTypoWord(String`
2	`targetWord,StringBuilder sb){`
3	`if(sb.toString().contains("?"+targetWord+"\t"))`
4	`return targetWord;`
5	`try {`
6	`if(sb!=null){`
7	`String tmpString[] =`
8	`sb.toString().split("\\t");`
	`String tmp[];`

```
                            for(int
i=0;i<tmpString.length;i++){
                                    tmp =
tmpString[i].split("\\?");
if(((tmp[1].length()-targetWord.length())==1)||
((targetWord.length()-tmp[1].length())==1)){
            if(distance(tmp[1], targetWord)<=1){
                                    return tmp[1];
                                    }
                        }
                    }
        }else{
                        return null;
                        }
                    } catch (Exception e) {
                        // TODO: handle exception
                    }
                    return null;
        }

        public static int distance(String parameter, String
input) {
            parameter = parameter.toLowerCase();
            input = input.toLowerCase();
            // i == 0
            int [] costs = new int [input.length() + 1];
            for (int j = 0; j < costs.length; j++)
                costs[j] = j;
            for (int i = 1; i <= parameter.length(); i++) {
                // j == 0; nw = lev(i - 1, j)
                costs[0] = i;
                int nw = i - 1;
                for (int j = 1; j <= input.length(); j++) {
                    int cj = Math.min(1 + Math.min(costs[j],
costs[j - 1]), parameter.charAt(i - 1) == input.charAt(j -
1) ? nw : nw + 1);
                    nw = costs[j];
                    costs[j] = cj;
                }
            }
            return costs[input.length()];
        }
```

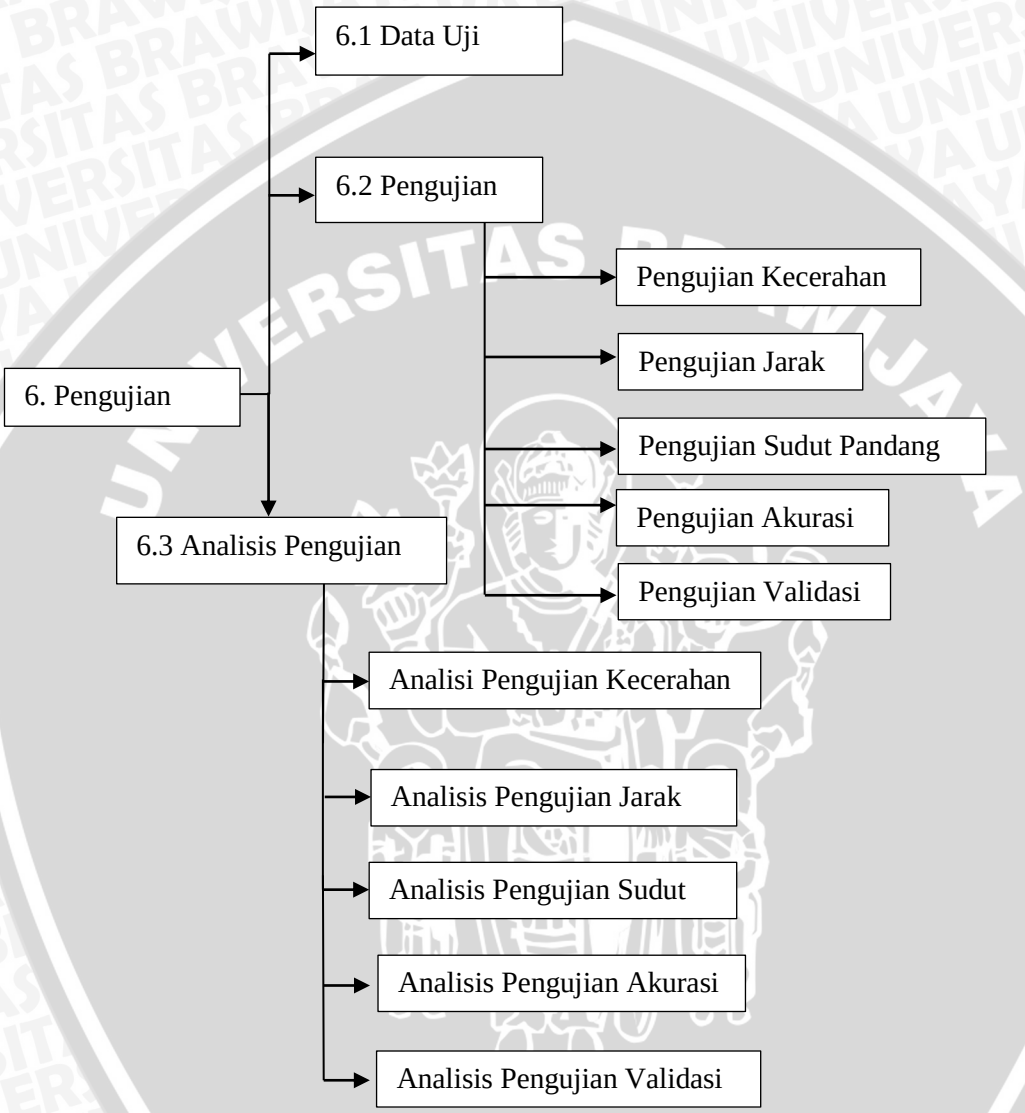
Source Code 5. 3 Fitur Mengkoreksi Kata

Gambar 5.3 bagian kiri merupakan kondisi aplikasi akan menangkap gambar untuk dideteksi dimana 1 kata hilang sedangkan bagian kanan merepresentasikan hasil terjemahan dari teks yang telah dikoreksi pada gambar teks tersebut.



Gambar 5. 3 Tampilan Fitur mengkoreksi kata salah

BAB VI
PENGUJIAN



6.1 Data Uji

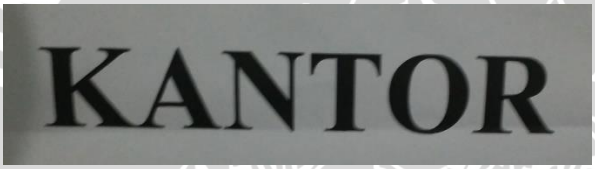
Data uji digunakan sebagai objek dari metode pengujian yang dilakukan untuk mengetahui tingkat keberhasilan masing-masing pengujian. Pada Tabel 6.1 tersedia kolom gambar, font, ukuran *font*,dan status. Kolom gambar berisi penampakan

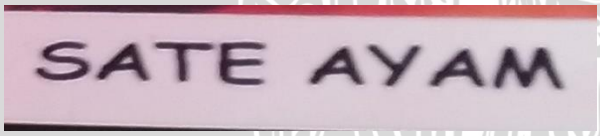
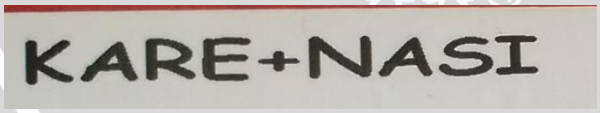
gambar data yang akan diuji. Berdasarkan gambar data uji akan dideteksi jenis *font* dan ukuran font dengan menggunakan perangkat ketiga yaitu website http://www.myfonts.com/WhatTheFont/. Kolom *font* berisi *font* yang berhasil dideteksi berdasarkan hasil deteksi *website myfonts* sedangkan ukuran *font* didapat dari rata-rata ukuran teks pada gambar yang diuji. Kolom status berisi status keberhasilan aplikasi mampu mendeteksi gambar teks.

Tabel 6. 1 Data Uji

Data-uji	gambar	*Font*	Ukuran *font*
1		*Linotype Helvetica*	100pt
2		*Monotype Impact*	130pt
3		*Positype Air Soft*	80
4		*Aah Yes Shelflife*	110

5		*Arial*	50pt
6		*Linotype Helvetica*	100pt
7		*Linotype Helvetica*	100pt
8		*Monotype Arial*	60pt
9		*Arial Black*	50pt
10		*Monotype Rockwell*	65pt
11		*Arial Black*	50pt

12		*Arial Black*	60pt
13		*Arial Black*	50pt
14		*Monotype CG Times*	100pt
15		*Arial Black*	100pt
16		*Arial Black*	50pt
17		-	60pt
18		*Aah Yes Shelflife*	60pt

19		*Arial Black*	90pt
20		*Arial Black*	50pt
21		*Arial Black*	100pt
22		-	50pt
23		-	50pt
24		*Linotype Arian*	40pt
25		*Arial Black*	90pt

26		-	-
27		*Arial Black*	-
28		-	-
29		*Arial Black*	-
30		-	-
31		-	-

32		-	-
33		-	-
34		-	-
35		-	-

6.2 Pengujian

Pengujian aplikasi penampil berbasis *augmented reality* pada aplikasi kamus kamera *realtime* menggunakan pengujian tingkat kecerahan, jarak optimal, sudut pandang ,dan akurasi. Masing-masing pengujian akan menggunakan data uji yang diatas. Untuk pengujian tingkat kecerahan, jarak optimal ,dan sudut pandang menggunakan data uji ke-1 sampai ke-20 dari data uji yang telah disebutkan pada subbab sebelumnya. Pengujian akurasi menggunakan seluruh data uji yang disebutkan sebelumnya.

6.2.1 Pengujian Tingkat Kecerahan

Pengujian tingkat kecerahan digunakan untuk menguji akurasi aplikasi dari proses mendeteksi sampai menghasilkan atau menampilkan hasil terjemahan pada

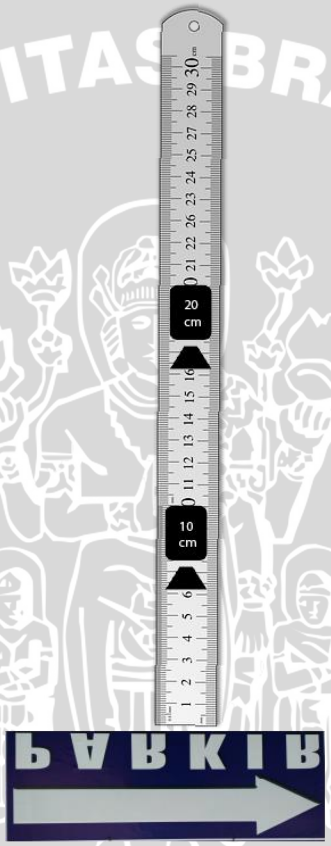
variasi gambar pada tingkat kecerahan tertentu. Gambar 6.1 merupakan contoh dari salah satu data uji yang diubah tingkat kecerahannya. Gambar dengan kecerahan 10% didapatkan dari gambar asli yang mana tingkat kecerahannya ditambah sebanyak 10% dari gambar asli Sedangkan gambar dengan tingkat kecerahan – 10% didapat dari mengurangi tingkat kecerahan gambar 10% dari gambar asli.

Gambar 6. 1 Variasi Tingkat kecerahan data uji ke-1

Pada Tabel 6.6 menjelaskan hasil pengujian tingkat kecerahan pada data uji ke-1 sampai ke-20. Kolom yang berisi nilai “TRUE” merepresentasikan data uji ke-n pada tingkat kecerahan ke-i mampu terdeteksi dan menampilkan terjemahan sedangkan kolom yang berisi nilai “FALSE” merepresentasikan aplikasi tidak mampu mendeteksi dan menampilkan terjemahan.

6.2.2 Pengujian Jarak Optimal

Pengujian jarak optimal digunakan untuk mencari jarak terdekat dan terjauh yang dibutuhkan aplikasi untuk mengenali gambar dan menampilkan hasil terjemahan. Pada pengujian ini dilakukan pada data uji ke-1 sampai ke-20. Pengujian hanya berfokus pada jarak maka dari itu pengujian ini mengabaikan sudut kamera dalam mendeteksi teks. Variasi data uji berupa perbedaan jarak antara 5cm-200cm pada setiap data uji.



Gambar 6. 2 Skenario Pengujian Jarak

Pada Tabel 6.7 menggambarkan hasil pengujian jarak. Kolom yang berisi nilai “TRUE” merepresentasikan data uji ke-n pada jarak ke-i mampu terdeteksi dan menampilkan terjemahan sedangkan kolom yang berisi nilai “FALSE” merepresentasikan aplikasi tidak mampu mendeteksi dan menampilkan terjemahan.

6.2.3 Pengujian Sudut Pandangan

Pengujian sudut pandangan merupakan pengujian yang digunakan untuk mencari sudut pendangan optimal untuk mendeteksi dan menampilkan hasil terjemahan. Pada pengujian ini dilakukan pada data uji ke-1 sampai ke-20. Pengujian hanya berfokus pada sudut horizontal objek terdahap kamera. Sudut variasi yang digunakan adalah 90˚ sampai 10˚ setiap data uji. Pada gambar 6.2 menjelaskan skenario pengujian gambar berisi teks parker merupakan contoh objek. Gambar kamera pada sudut 90˚ sampai 10˚ merupakan variasi setiap data uji.

Gambar 6. 3 Skenario Pengujian Sudut Pandangan

Pada Tabel 6.8 menggambarkan hasil pengujian sudut pandang. Kolom yang berisi nilai “TRUE” merepresentasikan data uji ke-n pada sudut ke-i mampu terdeteksi dan menampilkan terjemahan sedangkan kolom yang berisi nilai “FALSE” merepresentasikan aplikasi tidak mampu mendeteksi dan menampilkan terjemahan.

6.2.4 Pengujian Akurasi

Pengujian akurasi digunakan untuk mencari tingkat akurasi aplikasi untuk mengenali teks pada gambar dan menampilkan terjemahannya. Pengujian ini menggunakan seluruh data uji. Kemudian dari seluruh data uji disimpulkan presentase tingkat keberhasilan aplikasi mengenali dan menampilkan terjemahan masing-masing data-uji. Tabel 6.9 merupakan hasil dari pengujian akurasi

menggunakan data uji. Status "terdeteksi" mengartikan bahwa gambar pada data uji ke-n mampu terdeteksi dan menampilkan terjemahan dalam kondisi gambar normal. Gambar normal berarti bahwa tingkat kecerahan gambar sesuai dengan gambar tersebut, jarak gambar sesuai dengan jarak optimal, dan sudut sesuai dengan sudut pandangan optimal.

6.2.5 Pengujian Validasi

Pada penelitian ini dilakukan uji kasus validasi terhadap aplikasi penampil berbasis *augmented reality* pada aplikasi kamus kamera *realtime*. Untuk mengetahui kesesuaian antara daftar daftar kebutuhan dan kinerja sistem, pada setiap kebutuhan sistem dilakukan proses pengujian kasus masing-masing akan ditunjukkan pada Tabel 6.2 sampai dengan Tabel 6.4.

Tabel 6. 2 Uji Validasi mendeteksi kata

Nomor Kasus Uji	VAL_02
Nama Kasus Uji	Kasus uji operasi mendeteksi kata
Objek Uji	Kebutuhan Fungsional (F01)
Tujuan Pengujian	Memastikan aplikasi mampu mendeteksi kata.
Prosedur Uji	Tekan start pada home aplikasi kemudian arahkan kamera menuju objek uji.
Hasil yang diharapkan	Aplikasi dapat menampilkan daerah kata yang terdeteksi.

Tabel 6. 3 Uji Validasi menerjemahkan teks

Nomor Kasus Uji	VAL_03
Nama Kasus Uji	Kasus uji Operasi menerjemahkan dan menampilkan teks
Objek Uji	Kebutuhan Fungsional (F02)
Tujuan Pengujian	Memastikan operasi menerjemahkan teks mengeluarkan terjemahan dengan benar.
Prosedur Uji	Aplikasi mendapatkan nilai dari kata yang terdeteksi

	kemudian menerjemahkan dan menampilkan hasil terjemahan.
Hasil yang diharapkan	Aplikasi mempu mengeluarkan hasil terjemahan dengan benar pada daerah hasil deteksi.

Tabel 6. 4 Uji Validasi Mengkoreksi kata salah

Nomor Kasus Uji	VAL_04
Nama Kasus Uji	Kasus uji mengkoreksi kata salah
Objek Uji	Kebutuhan Fungsional (F03)
Tujuan Pengujian	Untuk memastikan gambar dengan kata salah mampu terdeteksi dan menampilkan hasil terjemahan yang benar.
Prosedur Uji	Aplikasi mendapatkan nilai dari kata salah yang terdeteksi kemudian menerjemahkan dan menampilkan hasil terjemahan.
Hasil yang diharapkan	Aplikasi mempu menampilkan hasil terjemahan secara *realtime* pada daerah hasil deteksi.

Berdasarkan pada kasus uji yang dilakukan akan diperoleh hasil pengujian. Hasil pengujian validasi akan ditampilkan pada Tabel 6.5.

Tabel 6. 5 Hasil Pengujian Validasi

No.	Nomor Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan	Status Validasi
1.	VAL_01	Aplikasi dapat terbuka dan menampilkan halaman *home*.	Aplikasi dapat terbuka dan menampilkan halaman *home*.	VALID
2.	VAL_02	Aplikasi dapat menampilkan daerah kata yang terdeteksi.	Aplikasi dapat menampilkan daerah kata yang terdeteksi.	VALID
3.	VAL_03	Aplikasi mempu mengeluarkan hasil terjemahan dengan benar pada daerah	Aplikasi mempu mengeluarkan hasil terjemahan dengan benar pada daerah hasil	VALID

		hasil deteksi.	deteksi.	
4.	VAL_04	Aplikasi dapat melakukan pengambilan *file* citra dari *Gallery*.	*File* citra dapat diambil dari *Gallery*.	VALID
5.	VAL_05	Aplikasi mempu menampilkan hasil terjemahan secara *realtime* pada daerah hasil deteksi.	Aplikasi mempu menampilkan hasil terjemahan secara *realtime* pada daerah hasil deteksi.	VALID

Tabel 6. 6 Hasil Pengujian Tingkat Kecerahan

Pengujian gelap terang gambar(%)										
Data Ke-	-80	-60	-40	-20	-10	10	20	40	60	80
1	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
2	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
3	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
4	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
5	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
6	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
7	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
8	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
9	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
10	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
11	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
12	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
13	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
14	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
15	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
16	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
17	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
18	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
19	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
20	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE

Tabel 6. 7 Hasil Pengujian jarak

Pengujian dengan Jarak(cm)													
data-ke	15	20	25	30	35	45	50	55	100	150	200	250	300
1	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
2	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
3	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
4	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
5	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
6	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
7	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
8	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE
9	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
10	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE
11	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
12	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
13	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
14	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
15	FALSE	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
16	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
17	FALSE	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
18	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
19	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
20	FALSE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE

Tabel 6. 8 Hasil Pengujian Sudut Pandangan

Pengujian sudut pandangan									
Data Ke-	90	80	70	60	50	40	30	20	10
1	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
2	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE
3	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
4	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
5	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
6	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
7	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
8	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
9	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
10	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
11	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
12	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
13	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
14	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
15	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
16	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
17	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
18	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE
19	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
20	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE

Tabel 6. 9 Hasil Pengujian Akurasi

Data-uji	Status	Data-uji	Status	Data-uji	Status
1	terdeteksi	16	Terdeteksi	31	Tidak Tedeteksi
2	terdeteksi	17	Terdeteksi	32	Terdeteksi
3	Terdeteksi	18	terdeteksi	33	Terdeteksi
4	terdeteksi	19	terdeteksi	34	Tidak Terdeteksi
5	terdeteksi	20	Terdeteksi	35	Tidak Terdeteksi
6	terdeteksi	21	Tidak Terdeteksi		
7	terdeteksi	22	Terdeteksi		
8	Terdeteksi	23	Terdeteksi		
9	Terdeteksi	24	Terdeteksi		
10	Terdeteksi	25	Terdeteksi		
11	Terdeteksi	26	Tidak Terdeteksi		
12	Terdeteksi	27	Tidak Terdeteksi		
13	Terdeteksi	28	Tidak Terdeteksi		
14	Terdeteksi	29	Terdeteksi		
15	Terdeteksi	30	Tidak Terdeteksi		

Tabel 6. 10 Hasil Pengujian Validasi Hasil Terjemahan

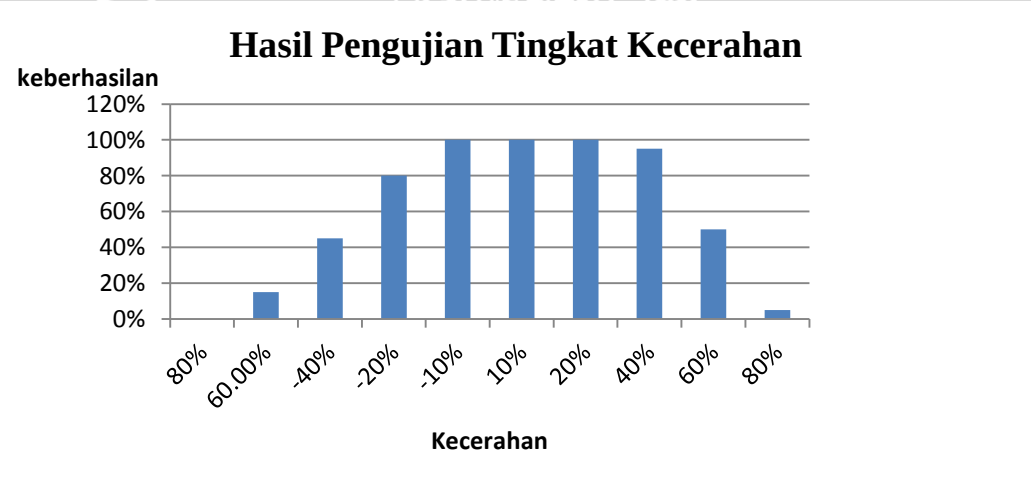
Data-uji	Status	Data-uji	Status
1	True	16	True
2	True	17	True
3	True	18	True
4	True	19	True
5	True	20	True
6	True	21	True
7	True	22	True
8	True	23	True
9	True	24	True
10	True	25	True
11	True	26	True
12	True	27	True
13	True	28	True
14	True	29	True
15	True	30	True

6.3 Analisis Pengujian

Analisis Pengujian bertujuan untuk mendapatkan kesimpulan dari hasil pengujian aplikasi penampil berbasis *augmented reality* pada kamus kamera *realtime* ini. Analisis dilakukan berdasarkan langkah-langkah pengujian yang dilakukan. Pada proses analisis dilakukan sebanyak empat tahapan, yaitu analisis tingkat kecerahan, jarak optimal, sudut pandang, dan akurasi.

6.3.1 Analisis Hasil Pengujian Tingkat Kecerahan

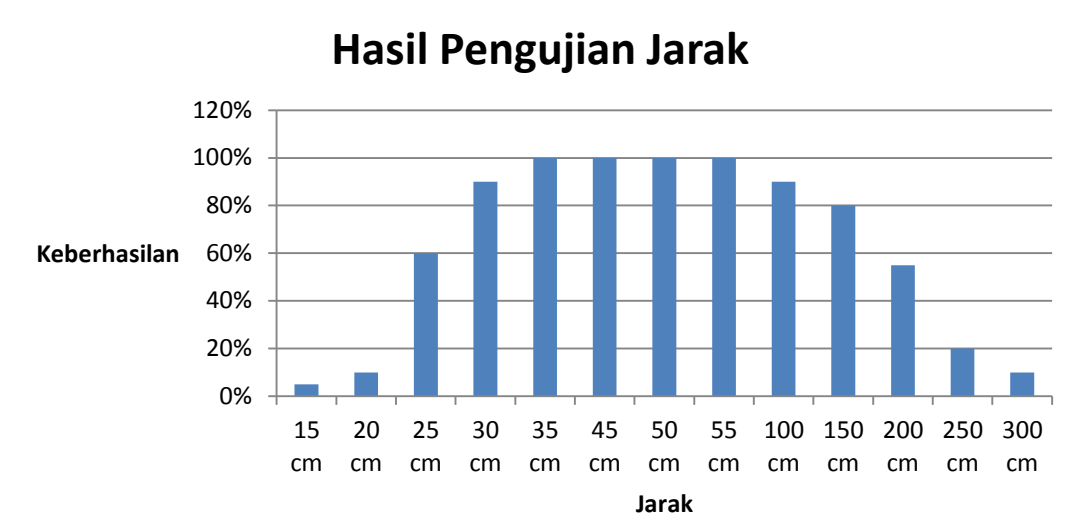
Dari Tabel 6.6 dapat dibuat analisis hasil pengujian tingkat keceranan seperti pada Gambar 6.2 yang menjelaskan hasil pengujian tingkat kecerahan. Dari Hasil tersebut didapatkan tingkat kecerahan optimal -40% sampai 20% agar gambar mampu dikenali dan aplikasi menampilkan hasil terjemahan. Diluar derajat optimal tersebut teks pada gambar akan relatif sulit untuk dideteksi.



Gambar 6. 4 Hasil Pengujian Kecerahan

6.3.2 Analisis Hasil Pengujian Jarak Optimal

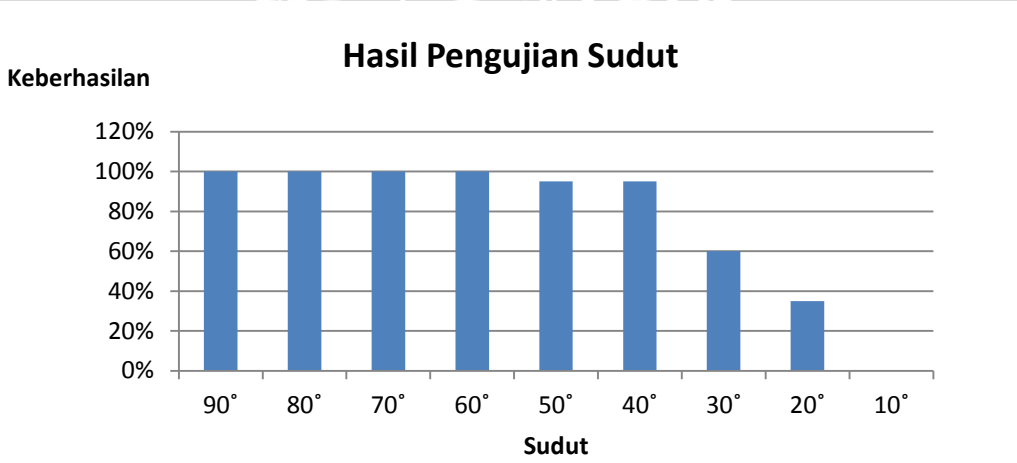
Dari tabel 6.7 dapat ditarik kesimpulan melalui Gambar 6.3 yang menjelaskan hasil pengujian jarak optimal untuk mendeteksi teks dan menerjemahkannya. Dari Hasil tersebut didapatkan jarak optimal 30-150cm agar gambar mampu dikenali dan aplikasi menampilkan hasil terjemahan. Dari hasil tersebut diketahui jarak minimum yang dibutuhkan adalah 30cm dan jarak maksimum adalah 150cm.



Gambar 6. 5 hasil pengujian Jarak

6.3.3 Analisis Hasil Pengujian Sudut Pandang

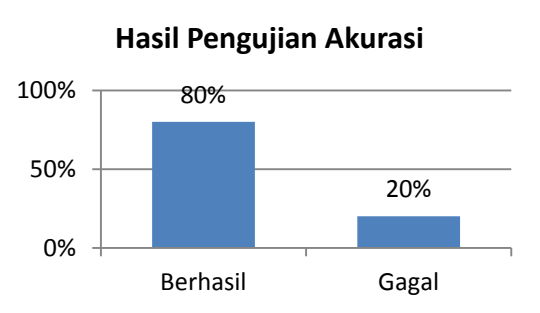
Dari Tabel 6.8 pengujian sudut pandang dapat disimpulan melalui Gambar 6.4 yang menjelaskan hasil pengujian tingkat kecerahan. Dari Hasil tersebut didapatkan tingkat sudut pandang optimal 90˚ sampai 40˚ agar gambar mampu dikenali dan aplikasi menampilkan hasil terjemahan.



Gambar 6. 6 Hasil Pengujian Sudut Pandang.

6.3.4 Analisis Hasil Pengujian Akurasi

Berdasarkan Tabel 6.9 dapat disimpulkan hasil pengujian akurasi dalam Gambar 6.5 ditunjukkan bahwa tingkat keberhasilan aplikasi dalam mendeteksi dan menampilkan terjemahan mencapai 80% dari 35 data gambar teks yang memiliki perbedaan jenis huruf.



Gambar 6. 7 Hasil Pengujian Akurasi

6.3.6 Analisis Hasil Pengujian Validasi

Proses analisis terhadap hasil pengujian validasi dilakukan dengan melihat kesesuaian antara hasil kinerja sistem dengan daftar kebutuhan fungsional. Berdasarkan hasil pengujian validasi dari Tabel 6.2 sampai Tabel 6.5 dapat disimpulkan bahwa Aplikasi penampil berbasis *augmented reality* pada aplikasi kamus kamera *realtime* telah memenuhi kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan. Analisis kebutuhan yang dimaksud adalah dapat menjalankan semua fitur aplikasi, yaitu mendeteksi teks, menerjemahkan teks , dan mengkoreksi kata salah.

BAB VII
PENUTUP

7.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian dari aplikasi yang dibuat, maka diambil kesimpulan sebagai berikut:

1. Aplikasi penampil berbasis *augmented reality* berhasil diimplemantasikan pada aplikasi kamus kamera *realtime*.
2. Berdasarkan hasil implementasi dan pengujian validasi, aplikasi berhasil diimplementasikan sesuai dengan spesifikasi kebutuhan dan perancangan.
3. Berdasarkan hasil pengujian kecerahan, aplikasi mampu mengenali dan menampilkan hasil terjemahan secara optimal pada tingkat kecerahan -40% sampai 20%.
4. Berdasarkan hasil pengujian jarak, aplikasi mampu mengenali dan menampilkan hasil terjemahan secara optimal pada jarak optimal 30cm sampai 150 cm.

5. Berdasarkan hasil pengujian sudut pandangan, aplikasi mampu mengenali dan menampilkan hasil terjemahan secara optimal pada sudut 90˚ sampai 40˚.
6. Berdasarkan pengujian tingkat akurasi keberhasilan aplikasi mampu mendeteksi dan menampilkan terjemahannya adalah 80%.

7.2 Saran

Aplikasi penampil ini masih mempunyai banyak kekurangan, sehingga perlu dikembangkan lagi agar menjadi lebih sempurna. Untuk meningkatkan kualitas dan fungsionalitas dari aplikasi, saran yang diberikan untuk pengembangan penelitian selanjutnya antara lain:

1. Menambahkan fitur melihat keterangan teks yang terdeteksi.
2. Menambahkan algoritma yang effisien untuk menerjemahkan kalimat yang terdiri dari lebih dari 2 kata.
3. Menambahkan fitur untuk menampilkan hasil terjemahan satu kali dalam area yang terdeteksi.

DAFTAR PUSTAKA

[1] A. Holzer and J. Ondrus, "Mobile Application Market: AMobile Network Operators' Perspective," in Exploring the Grand Challenges for Next Generation E-Business, ser. Lecture Notes in Business Information Processing, W. Aalst et al., Eds. Springer Berlin Heidelberg, 2011, vol. 52, pp. 186–191

[2] Mauricio Hincapié. " An Introduction to *Augmented Reality* with Applications in Aeronautical Maintenance",Mexico: Instituto Tecnològico y de Estudios Superiores de Monterrey. 2011

[3] Anonymouse.(____). Levenshtein Algorithm(online). Available: http://www.levenshtein.net/

[4] ANDI.Step by step menjadi programmer android.Penerbit ANDI.2013

[5]Madden Lester. Professional *Augmented Reality* Browsers for *Smartphones*, United Kingdom:Wiley Publishing. 2011

[6]Vuforia.(2014). Vuforia (online documentation). Available: https://developer.Vuforia.com/resources/dev-guide/word-targets.

[7]Rizky Yuduantika, Aditya. Implemantasi *Augmented Reality* di Museum: studi awal pearancangan aplikasi untuk pengunjung museum,Yogyakarta: Universitas Gajah Mada. 2013

[8] Rizky, Yoze. Markerless *Augmented Reality* pada perangkat Android.Surabaya: Institut Teknologi Sepuluh November

[9] Siltanen ,Sanni. Theory and applications of marker-based *augmented reality*,Finland:VTT. 2012

[10]Google.(____) .Android Anatomy and Physiology(e-book). Available: http://androidteam.googlecode.com/files/Anatomy-Physiology-of-an-Android.pdf.

[11] Yovcheva ,Zornitza. Overview of *Smartphone Augmented Reality* Applications for Tourism. Bournemouth University, United Kingdom.

[12] McReynolds ,Tom. Programming with OpenGL: An Introduction. Silicon Graphics Computer Systems.1996

[13] Author. OpenGL Programing Guide(online). Available : http://www.sgi.com/Technology/openGL

[14] Fragoso ,Victor. TranslatAR: A Mobile *Augmented Reality* Translator on the Nokia N900.University of California

[15] Mithe ,Ravina. *Optical Character Recognition*, IJRTE. 2013

[16] Luckin ,Rosemary. Using Mobile Technology to Create Flexible Learning Contexts, Journal of Interactive Media in Education. 2005

[17] American Public Transportation Assotiation.(2007).Public Transportation: Benefits for the 21st Century. Available: http://www.apta.com/resources/reportsandpublications/Documents/twenty_first_century.pdf

[18] Patkar S. , Raviraj. Marker Based *Augmented Reality* Using Android OS, International Journal of Advanced Research in Computer Science and Software Engineering.2013

[19] Fragoso ,Victor. TranslatAR: A Mobile *Augmented Reality* Translator,IEEE.2010

[20] G.Vamvakas. A Complete *Optical Character Recognition* Methodology for Historical

[21] Two-Dimensional Programming in OpenGL.

[22]Fowler Martin.UML distilled.Yogyakarta: ANDI, 2005.

[23]Song ,Il-Yeol. A Comparative Analysis of Entity-Relationship Diagrams. Journal of Computer and Software Engineering.1995

[24] Winda Destiana Putri. (2014, 16 Oktober). Mari Elka: Perkembangan Perhotelan Indonesia Pesat [online]. Available: http://www.republika.co.id/berita/gaya-hidup/travelling/14/10/16/ndiabg-mari-elka-perkembangan-perhotelan-indonesia-pesat

[25] Personal, Social and Humanities Education Section Education Bureau. 2013