TUGAS AKHIR – KI 141502

APLIKASI PENERJEMAH GAMBAR TEKS BERBAHASA INGGRIS MENGGUNAKAN TEKNOLOGI REALITAS TERTAMBAH PADA PERANGKAT BERBASIS ANDROID

ANTONIUS KEVIN WIGUNA
NRP 05111440000132

Dosen Pembimbing
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Wijayanti Nurul Khotimah, S.Kom., M.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KI 141502

APLIKASI PENERJEMAH GAMBAR TEKS BERBAHASA INGGRIS MENGGUNAKAN TEKNOLOGI REALITAS TERTAMBAH PADA PERANGKAT BERBASIS ANDROID

ANTONIUS KEVIN WIGUNA
NRP 05111440000132

Dosen Pembimbing
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Wijayanti Nurul Khotimah, S.Kom., M.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

(Halaman ini sengaja dikosongkan)

FINAL PROJECT– KI 141502

ENGLISH TEXT IMAGE TRANSLATOR APPLICATION USING AUGMENTED REALITY TECHNOLOGY ON ANDROID-BASED DEVICE

ANTONIUS KEVIN WIGUNA
NRP 05111440000132

Advisors
Dr.Eng. Nanik Suciati, S.Kom., M.Kom.
Wijayanti Nurul Khotimah, S.Kom., M.Sc.

DEPARTMENT OF INFORMATICS
Faculty Of Information and Communication Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2018

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

APLIKASI PENERJEMAH GAMBAR TEKS BERBAHASA INGGRIS MENGGUNAKAN TEKNOLOGI REALITAS TERTAMBAH PADA PERANGKAT BERBASIS ANDROID

Tugas Akhir

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Interaksi, Grafika, dan Seni
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:
ANTONIUS KEVIN WIGUNA
NRP. 05111440000132

Disetujui oleh Dosen Pembimbing Tugas Akhir

Dr.Eng. Nanik Suciati S.Kom., M.Kom. .....
NIP: 19710428 199412 2 001 (pembimbing 1)

Wijayanti Nurul Khotimah S.Kom., M.Sc. .....
NIP: 19860312 201212 2 004 (pembimbing 2)

SURABAYA
JUNI, 2018

(Halaman ini sengaja dikosongkan)

APLIKASI PENERJEMAH GAMBAR TEKS BERBAHASA INGGRIS MENGGUNAKAN TEKNOLOGI REALITAS TERTAMBAH PADA PERANGKAT BERBASIS ANDROID

Nama Mahasiswa	: Antonius Kevin Wiguna
NRP	: 05111440000132
Departemen	: Informatika FTIK-ITS
Dosen Pembimbing I	: Dr.Eng. Nanik Suciati S.Kom., M.Kom.
Dosen Pembimbing II	: Wijayanti Nurul Khotimah S.Kom., M.Sc.

ABSTRAK

Di dalam lingkungan dunia nyata, banyak informasi berupa teks yang disajikan menggunakan bahasa yang tidak kita mengerti. Informasi tersebut bisa kita temui di tempat-tempat umum. Biasanya, diperlukan alat bantu untuk memahami tulisan dalam bahasa asing tersebut. Kemudahan dan kecepatan dalam penggunaan alat penerjemah menjadi penting terutama bila kita sedang berada di tempat umum.

Untuk itu, diajukan pengembangan sebuah aplikasi yang mampu menerjemahkan teks yang mudah digunakan. Pengguna memilih gambar teks yang ingin diterjemahkan dari tampilan kamera aplikasi dengan menyentuh gambar teks yang diinginkan. Area gambar teks diambil menggunakan algoritma berbasis deteksi tepi pada gambar. Teks dikenali menggunakan Optical Character Recognition (OCR) dengan bantuan pustaka Tesseract dan diterjemahkan menggunakan bantuan Yandex. Translate API. Hasil penerjemahan akan ditampilkan menggunakan konsep realitas tertambah, di mana hasil akan ditampilkan langsung pada tampilan kamera di aplikasi. Aplikasi ini dibuat untuk berjalan pada perangkat yang menggunakan sistem operasi Android. Penerjemahan yang dapat dilakukan adalah penerjemahan dari bahasa Inggris ke bahasa Indonesia.

Pengujian aplikasi dilakukan menggunakan metode kotak hitam untuk menguji setiap komponen aplikasi dan dengan melakukan pengujian ke sejumlah orang untuk mengetahui kenyamanan dalam menggunakan aplikasi. Pengujian menunjukkan hasil yang cukup baik untuk pengambilan area region of interest (ROI) dan proses mendapatkan hasil terjemahan, Hasil pengujian pada pengguna menjukkan bahwa pengguna puas dengan kenyamanan, performa, dan kemudahan penggunaan aplikasi.
Kata kunci**: **Penerjemah, Gambar Teks, Bahasa Inggris, Bahasa Indonesia, Realitas Tertambah, Android

ENGLISH TEXT IMAGE TRANSLATOR APPLICATION USING AUGMENTED REALITY TECHNOLOGY ON ANDROID-BASED DEVICE

Student Name	: Antonius Kevin Wiguna
NRP	: 05111440000132
Major	: Informatika FTIK-ITS
Advisor I	: Dr.Eng. Nanik Suciati S.Kom., M.Kom.
Advisor II	: Wijayanti Nurul Khotimah S.Kom., M.Sc.

ABSTRACT

In the real life environment, there are so many text information that is written in languages that we don't understand. We can find those information in public places. Usually, we need a tool to understand information written in foreign languages.. Ease and speed in using translation tools become important especially when being used in public places.

For that problem, a development for an aplication that can translate text through camera view Users choose text image that they want to translate from the camera view by touching the desired text image. Text image area obtained by using an edge detection based algorithm. Text image is recognized using Optcal Character Recognition (OCR) with Tesseract library and then will be translated using Yandex. Translate API. Translation results will be displayed using augmented reality concept, where the result is showed directly on the application's camera view. The application supports English to Bahasa Indonesia translation.

Application testing is done by using blacbox method to test every application components and by doing testing to some people to know the comfortness when the application is being used by its users. Testing result shows quite good results for ROI are obtaining method and translation result creation method.

User testing results shows that users are satisfied with application comfort, performance, and ease of use.
Keywords**: **Translator, Text Image, English, Bahasa Indonesia, Augmented Reality, Android

KATA PENGANTAR

Pertama-tama, puji dan syukur penulis sampaikan kepada Tuhan Yang Maha Esa karena berkat bantuan dan bimbingan-Nya, penulis mampu menyelesaikan tugas akhir yang berjudul ”Aplikasi Penerjemah Gambar Teks Berbahasa Inggris Menggunakan Teknologi Realitas Tertambah pada Perangkat Berbasis Android” ini.

Pengerjaan tugas akhir ini dibuat dalam rangka pemenuhan syarat untuk memperoleh gelar Sarjana Komputer untuk program studi Informatika Fakultas Teknologi Informasi dan Komunikasi Institut Teknologi Sepuluh Nopember Surabaya. Tugas akhir ini juga menjadi kesempatan dan pembuktian bagi penulis untuk menggunakan ilmu yang didapatkan selama kuliah dalam menyelesaikan salah satu permasalahan yang ada di dunia nyata.

Dalam pelaksanaan dan pembuatan tugas akhir ini penulis mendapatkan sangat banyak bantuan dari berbagai pihak. Melalui lembar ini, penulis ingin secara khusus menyampaikan ucapan terima kasih kepada:

1. Tuhan Yesus Kristus, yang telah membimbing dan menyertai, serta memberi kesempatan untuk penulis dalam mengerjakan tugas akhir ini.
2. Ayah penulis, Argasianus Wiguna, ibu penulis, Grace Tjio May Eng, serta adik-adik penulis, Natalie Eileen Wiguna dan Fransiscus Jason Wiguna, yang selalu memberi semangat dan doa bagi penulis dalam menjalani kuliah dan hidup hingga titik ini.
3. Ibu Dr.Eng. Nanik Suciati S.Kom., M.Kom. dan Wijayanti Nurul Khotimah S.Kom., M.Sc., selaku dosen pembimbing yang telah menyediakan waktunya untuk membimbing penulis dalam menyelesaikan semua proses tugas akhir.
4. Bapak Dr. Eng. Darlis Herumurti, S.Kom., M.Kom. selaku ketua departemen Informatika ITS, Bapak Radityo Anggoro, S.Kom, M.Sc. selaku koordinator Tugas Akhir

dan koordinator Kerja Praktik dan segenap Bapak/Ibu dosen dari departemen Informatika maupun dari pihak UPMB yang telah memberikan ilmunya kepada penulis selama kuliah ini.

5. Seluruh jajaran staff departemen Informatika ITS yang telah memberikan bantuan kepada penulis semasa kuliah hingga pengerjaan tugas akhir ini.
6. Teman-teman penulis dari angkatan 2014, yang telah memberikan bimbingan, pertolongan dan semangat bagi penulis dalam mengerjakan tugas akhir ini.
7. Teman-teman penuls dari angkatan 2013 yang telah bersedia memberikan pengalamannya untuk membimbing penulis dalam mengerjakan tugas akhir ini.
8. Masyarakat sekitar kampus ITS Sukolilo yang telah memberikan dukungan bagi penulis selama kuliah di kampus ITS Sukolilo
9. Pihak-pihak lain yang belum disebutkan satu per satu yang juga turut membantu penulis menyelesaikan tugas akhir ini.

Penulis telah berusaha sebaik mungkin dalam mengerjakan tugas akhir ini, namun penulis mohon maaf apabila terdapat kesalahan dan kekurangan dalam pengerjaan tugas akhir ini. Penulis sadar bahwa pengalaman dan ilmu yang dimiliki oleh masih jauh dari kata sempurna. Kritik dan saran sangat penulis harapkan untuk perbaikan kedepannya. Akhir kata, penulis berharap tugas akhir ini dapat menginspirasi dan memberikan ide bagi orang lain yang membacanya

Surabaya, Juni 2018
Penulis

Antonius Kevein Wiguna

DAFTAR ISI

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR KODE SUMBER

(Halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

1.1 Latar Belakang

Berada di tempat dengan bahasa yang tidak kita kuasai seringkali menimbulkan masalah. Kita tidak dapat menggunakan pengetahuan bahasa yang kita miliki untuk menyerap informasi dari lingkungan sekitar. Salah satunya adalah informasi yang berupa tulisan. Informasi berupa tulisan banyak sekali terdapat baik dalam lingkungan nyata seperti jalan raya dan fasilitas umum, maupun lingkungan maya seperti di Internet. Kesulitan akan muncul ketika informasi yang kita butuhkan adalah informasi yang penting, seperti informasi cara menggunakan transportasi umum atau informasi posisi dan arah untuk menuju suatu tempat yang biasanya terdapat pada rambu-rambu lalu lintas di jalan.

Salah satu solusi untuk mengatasi masalah tersebut adalah dengan menggunakan alat penerjemah bahasa. Alat penerjemah bahasa sudah tersedia dalam berbagai macam bentuk dan jenis. Dilihat dari segi metode masukannya, alat penerjemah bahasa menggunakan berbagai jenis metode, seperti mengetikkan masukan berupa teks [1], mengucapkan kata yang ingin diterjemahkan [2], ataupun dengan menggunakan deteksi tulisan tangan, khususnya untuk bahasa yang menggunakan huruf non-Latin [3]. Metode-metode ini memiliki kekurangan dimana metode-metode tersebut tidak terlalu optimal jika kita melihat suatu teks dalam bahasa asing dan ingin menerjemahkannya ke dalam bahasa yang kita ketahui. Teks perlu dimasukkan ulang secara manual ke dalam alat penerjemah sehingga agak memakan waktu jika kita melihat sebuah teks dan ingin hasil terjemahan yang lebih cepat, misalnya ketika sedang berada dalam tempat umum, akan lebih baik tentunya jika kita tidak terlalu berlama-lama diam dalam satu tempat hanya untuk mengoperasikan alat penerjemah.

Untuk mengatasi masalah tersebut tugas akhir ini akan mengembangkan sebuah aplikasi penerjemah teks dengan metode *input* yang mudah digunakan, yaitu dengan menggunakan kamera pada perangkat bergerak dengan sistem operasi Android, dimana teks yang ingin diterjemahkan diinputkan melalui kamera dan hasil translasi akan ditampilkan menggunakan teknologi realitas tertambah (*augmented reality*) langsung pada gambar teks yang didapatkan melalui kamera. Pengguna memilih tulisan yang ingin diterjemahkan dengan cara menyentuh gambar teks yang ingin diterjemahkan pada layar perangkat bergerak. Gambar tulisan akan diambil dari gambar kamera dengan membentuk batas-batas dari gambar teks yang dipilih berdasarkan masukan pengguna. Nantinya akan dilakukan ekstraksi data teks dari gambar tersebut menggunakan metode Optical Character Recognition (OCR). Setelah itu teks akan diterjemahkan dan hasil terjemahan tersebut akan diubah menjadi sebuah gambar yang diletakan di layar dalam bentuk realitas tertambah. Penerjemahan yang dapat dilakukan oleh aplikasi ini adalah penerjemahan dari bahasa Inggris ke bahasa Indonesia.

Dari pengerjaan tugas akhir ini, diharapkan aplikasi teks berbasis realitas tertambah yang akan dibangun ini dapat menjadi sebuah aplikasi penerjemah yang mudah digunakan dan dapat memberikan hasil terjemahan yang cepat serta memiliki keakuratan translasi yang baik, sehingga nantinya dapat digunakan untuk masalah-masalah dalam kehidupan nyata.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang dikemukakan, beberapa permasalahan yang akan diselesaikan dalam tugas akhir adalah sebagai berikut:

1. Bagaimana cara mengambil area gambar teks yang ingin diterjemahkan oleh pengguna dari gambar yang diperoleh melalui kamera perangkat bergerak berbasis Android?

2. Bagaimana cara mengenali teks dari gambar teks dan melakukan penerjemahan dari hasil pengenalan teks tersebut?
3. Bagaimana cara melakukan augmentasi pada masukan gambar dari kamera perangkat bergerak berbasis Android dengan hasil terjemahan secara langsung (*real time*)?
4. Bagaimana cara melakukan pengujian terhadap aplikasi penerjemah gambar teks berbahasa Inggris menggunakan teknologi realitas tertambah pada perangkat berbasis Android yang dibahas dalam tugas akhir ini?

1.3 Batasan Masalah

Beberapa batasan masalah yang menjadi batas pada tugas akhir ini adalah sebagai berikut:

1. Aplikasi ini berjalan pada platform Android dan dibagun dengan bahasa pemrograman Java dengan pustaka OpenCV4Android *Software Development Kit* (SDK), Tesseract versi tess-two dan Yandex Translate API.
2. Perangkat bergerak berbasis Android yang akan digunakan dalam tugas akhir ini dalam tahap implementasi dan pengujian adalah Asus Zenfone 2 Laser ZE500KL. Tabel 1.1 menjelaskan spesifikasi perangkat keras dari perangkat Asus Zenfone 2 Laser ZE500KL yang digunakan.

Tabel 1.1 Spesifikasi Perangkat Keras dari Perangkat Asus Zenfone 2 Laser ZE500KL

Central Processing Unit (CPU)	Quad-core 1.2 GHz Cortex-A53
Graphics Processing Unit (GPU)	Adreno 306 450 MHz
Random Access Memory (RAM)	2 GB
Penyimpanan Internal	16 GB
Resolusi Layar	1280 x 720 piksel

3. Penerjemahan yang dapat dilakukan oleh aplikasi adalah penerjemahan dari bahasa Inggris ke dalam bahasa Indonesia.
4. Pengguna hanya dapat menerjemahkan satu kata dalam satu kali proses penerjemahan.
5. Dalam layar perangkat bergerak hanya satu hasil penerjemahan yang muncul dalam satu waktu.
6. Gambar teks yang dapat dikenali berupa gambar teks cetak (bukan tulisan tangan).
7. Warna kata dan latar belakangnya masing-masing bersifat seragam (hanya memiliki satu warna)
8. Gerakan perangkat yang dapat ditangani merupakan gerakan pelan dan tidak tiba-tiba.

1.4 Tujuan

Tujuan dari pembuatan tugas akhir ini adalah membuat sebuah aplikasi realitas tertambah untuk melakukan penerjemahan bahasa Inggris ke bahasa Indonesia yang berupa gambar teks.

1.5 Manfaat

Manfaat dari pembuatan tugas akhir ini adalah memudahkan penggunanya untuk melakukan penerjemahan kata dalam bahasa Inggris ke bahasa Indonesia berupa gambar teks dengan memberikan hasil penerjemahan yang cepat.

1.6 Metodologi

Pembuatan tugas akhir dilakukan menggunakan metodologi yang terdiri dari tahapan-tahapan berikut:

A. Studi Literatur

Tahap ini merupakan tahap dimana semua teknologi yang akan digunakan untuk mendukung pengembangan aplikasi dipelajari. Literatur yang dipelajari bersumber dari media Internet. Teknologi yang akan dipelajari untuk mendukung

pengembangan aplikasi yang akan dibagun adalah:
1. Realitas tertambah ;
2. Sistem operasi Android;
3. OpenCV;
4. OCR;
5. Tesseract;
6. Yandex Translate API;
7. Histogram HSV

Literatur utama yang digunakan sebagai acuan utama tugas akhir ini adalah [4].

B. Perancangan Perangkat Lunak

Tahap ini merupakan tahap di mana aplikasi yang dibuat dalam tugas akhir ini dirancang. Perancangan dilakukan dengan memperhatikan kebutuhan fungsional dan non-fungsional serta teknologi yang akan digunakan berdasarkan hasil studi literatur. Tahap ini akan menghasilkan algoritma yang akan diimplementasikan pada tahap implementasi rancangan. Perancangan aplikasi akan memperhatikan tiga bagian inti berikut:

1. Pengambilan Teks yang akan Diterjemahkan
   Pada bagian ini dibuat rancangan tentang bagaimana aplikasi akan mengambil gambar teks yang akan diterjemahkan dari kamera perangkat bergerak berbasis sitem operasi Android milik pengguna.

2. Penerjemahan Kata
   Pada bagian ini dibuat rancangan tentang bagaimana aplikasi akan mengenali kata berdsarkan gambar dari tahap sebelumnya dan menerjemahkannya.

3. Penambahan Tampilan Kamera Perangkat Bergerak dengan Hasil Terjemahan

Pada bagian ini dibuat rancangan tentang bagaimana aplikasi akan menampilkan hasil terjemahan yang didapatkan dari tahap sebelumnya ke dalam tampilan kamera (*camera view*) pada perangka bergereak milik pengguna. Hasil terjemahan akan ditampilkan dengan cara melakukan penambahan ke dalam tampilan kamera pengguna sehingga membentuk tampilan realitas tertambah.

C. Implementasi Rancangan Sistem

Pada tahap ini, dilakukan implementasi dari hasil analisa dan perancangan perangkat lunak pada tahap sebelumnya untuk menghasilkan perangkat lunak yang menjadi tujuan dalam tugas akhir ini. Implementasi rancangan akan dilakukan mengguanakan bahasa pemrograman Jawa dan C++ dengan kakas bantu IDE Android Studio versi 3.0.1. Pustaka tambahan yang akan digunakan adalah OpenCV4Android SDK versi 3.4.3, Tesseract versi tess-two, dan Yandex. Translate API.

D. Uji Coba dan Evaluasi

Pengujian yang akan digunakan dalam tugas akhir ini adalah sebagai berikut:

a. Pengujian kotak hitam (*black box*)

Pada tahap ini dilakukan uji coba dengan menggunakan beberapa macam kondisi untuk mencoba aplikasi dapat berjalan atau tidak. Pengujian yang dilakukan adalah pengujian kotak hitam (*black box*). Dalam pengujian kotak hitam, aplikasi akan diuji apakah setiap fungsi aplikasi sudah berjalan dengan baik atau belum tanpa memperhatikan kode sumber apliksi. Salah satu metode yang dilakukan dalam pengujian kotak hitam adalah dengan memeriksa keluaran aplikasi apabila diberikan masukan tertentu dengan kondisi pengunaan aplikasi

tertentu [5]. Dalam pengujian tugas akhir ini, setiap komponen aplikasi pada bagian-bagian yang ditentukan pada tahap perancangan (tahap D) akan diuji performanya dalam kondisi-kondisi pengujian tertentu sesuai dengan tantangan yang akan dihadapi setiap fungsi ketika digunakan untuk kebutuhan nyata.

b. Pengujian usabilitas
Pengujian usabilitas adalah pengujian yang dilakukan oleh calon pengguna untuk menemukan permasalahan kegunaan dan kepuasan pengguna terhadap objek yang diujikan.

E. Penyusunan Laporan Tugas Akhir
Pada tahap ini dilakukan penyusunan laporan yang berisi dasar teori, dokumentasi dari permainan yang dibuat, dan hasil-hasil yang diperoleh selama pengerjaan tugas akhir.

1.7 Sistematika Penulisan

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

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

BAB II TINJAUAN PUSTAKA
Bab ini membahas teknologi dan teori yang menjadi dasar dalam pembuatan tugas akhir ini.

BAB III ANALISIS DAN PERANCANGAN
Bab ini membahas analisis dari sistem yang dibuat meliputi analisis permasalahan, deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifikasi pengguna. Kemudian

membahas rancangan dari sistem yang dibuat meliputi rancangan skenario kasus penggunaan, arsitektur, data, dan antarmuka.

BAB IV IMPLEMENTASI
Bab ini membahas implementasi dari rancangan sistem yang dilakukan pada tahap perancangan.

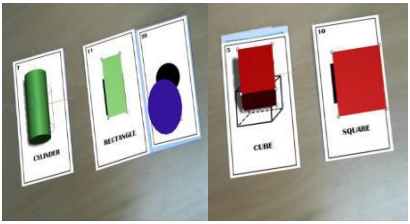
BAB V PENGUJIAN DAN EVALUASI
Bab ini membahas pengujian dari aplikasi yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan melakukan evaluasi terhadap hasil pengujian untuk melihat performa dari aplikasi yang telah dibuat.

BAB VI PENUTUP
Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan serta saran untuk pengembangan aplikasi sejenis di masa depan.

BAB II
TINJAUAN PUSTAKA

2.1 Realitas Tertambah (*Augmented Reality*)

Realitas tertambah (*Augmented reality*) adalah proses proyeksi dunia nyata, baik secara langsung maupun secara tidak langsung, yang diaugmentasi dengan elemen-elemen yang dihasilkan oleh komputer, seperti suara, video, dan grafis, ataupun hasil input dari dunia nyata itu sendiri [6]. Realitas tertambah sudah banyak diaplikasikan dalam kehidupan sehari-hari, contohnya dalam bidang pendidikan sebagai alat bantu pendidikan yang menarik, contohnya ditunjukan oleh Gambar 2.1 dan dalam bidang hiburan sebagai salah satu cara bermain yang baru untuk meningkatkan imersifitas media hiburan tersebut, contohnya ditunjukan oleh Gambar 2.2.



Gambar 2.1 Contoh Penerapan Teknologi Realitas Tertambah pada Bidang Pendidikan [7]

Gambar 2.2 Contoh Penerapan Teknologi Realitas Tertambah pada Bidang Hiburan [8]

Untuk melakukan penambahan pada proyeksi dunia nyata, khususnya untuk penambahan grafis, umumnya akan terlebih dahulu dilakukan proses pelacakan (*tracking*) dan pengenalan (*recognizing).* Proses *tracking* akan terlebih dahulu mengenali kondisi dunia nyata melalui metode-metode visi komputer. *Pelacakan* biasanya dilakukan pada gambar tertentu yang sudah ditentukan sebagai penanda (*marker*) atau melalui deteksi *interest point* pada gambar yang diambil melalui kamera. Metode yang biasanya dilakukan untuk *tracking* adalah deteksi fitur pada gambar dunia nyata, contohnya deteksi tepi. Setelah itu, proses pengenalan akan membangun sistem koordinat berdasarkan hasil deteksi pada tahap pelacakan dan memproyeksikan grafis yang akan ditampilkan ke dalam sistem koordinat tersebut. Pose kamera terhadap objek yang diproyeksikan tersebut akan ditentukan berdasarkan posisi kamera terhadap *interest point* yang sudah dideteksi pada tahap pelacakan, sehingga objek augmentasi akan terlihat melalui kamera sesuai dengan posisi kamera dan memberikan kesan bahwa objek seolah-oleh ada di dunia nyata [9].

2.2 Sistem Operasi Android

Android adalah sistem operasi perangkat bergerak berbasis Linux Kernel yang dikembangkan oleh Google dan Open Handset Alliance. Android merupakan sistem operasi yang bersifat *open source*, sehingga aplikasi-aplikasi yang berjalan pada sistem Android dapat memanfaatkan kemampuan perangkat keras dari perangkat bergerak yang berjalan pada sistem operasi Android secara penuh dan agar penggunanya dapat menggunakan aplikasi-aplikasi tersebut untuk mengkustomisasi perangkat Android yang dimiliki agar sesuai dengan keinginan dan kebutuhan pengguna [10].

Hingga kini, pengembangan Android telah mencapai versi 8.0 dan sudah mencakup berbagai jenis perangkat bergerak dari yang awalnya hanya mendukung perangkat bergerak berjenis *smartphone*. Jenis-jenis perangkat bergerak yang didukung oleh

sistem operasi Android diluar *smartphone* adalah *tablet*, *smart TV*, *smartwatch*, dan perangkat navigasi dan hiburan *on board* pada mobil [11]. Aplikasi Android dapat dibangun menggunakan bahasa Java, C++, dan Kotlin dan memiliki sebuah *Integrated Development Environment* (IDE) yang dikhusukan untuk pembangunan aplikasi Android, yaitu Android Studio. Kode sumber aplikasi Android dicompile menjadi sebuah file *installer* dengan ekstensi .apk [12].

2.3 OpenCV

OpenCV adalah sebuah pustaka yang digunakan untuk melakukan kegiatan *machine learning* dan visi komputer. OpenCV tersedia dalam berbagai jenis bahasa pemrograman, yaitu C++, Python dan Java, serta mendukung beberapa jenis platform, yaitu Windows, Linux, Android dan MacOS. OpenCV bersifat *open source* dan tersedia secara gratis. Untuk mendukung fungsinya sebagai *library* visi komputer, OpenCV memiliki berbagai fitur unggulan, diantaranya yang akan digunakan dalam pembuatan tugas akhir ini adalah fungsi identifikasi objek dan menjadikannya sebagai *marker* untuk *augmanted reality* [13].

Dalam pembuatan tugas akhir ini, *library* OpenCV yang akan digunakan adalah *Software Development Kit* (SDK) OpenCV4Android yang menyediakan API OpenCV untuk pengembangan aplikasi Android. API yang disediakan oleh OpenCV4Android ditulis dalam bahasa pemrograman Java dan C++ [14].

2.4 Optical Character Recognition

Optical Character Recognition (OCR) adalah sebuah metode konversi dari sebuah gambar atau gambar dari sebuah dokumen yang mengandung teks menjadi sebuah data teks yang dapat diubah isinya, misalnya menjadi sebuah *file* Microsoft Word (.doc). Tahapan pengenalan teks menggunakan metode OCR dimulai dengan melakukan *preprocessing* pada gambar atau

gambar dokumen yang mengandung teks. Kemudian, ekstraksi teks akan dimulai dari gambar huruf-huruf secara individual. Pengenalan huruf membutuhkan data latih berupa pola-pola karakter. Setelah karakter berhasil dikenali, karakter-karakter tersebut akan dibangun menjadi kata-kata, untuk kemudian dibangun menjadi kalimat-kalimat. Setelah itu kalimat-kalimat akan disimpan menjadi sebuah dokumen teks dengan format yang diperlukan [15]. Selain disimpan menjadi sebuah dokumen, beberapa jenis perangkat lunak atau layanan yang melayani OCR melayani konversi dengan menggunakan API. OCR yang menggunakan sistem API dapat digunakan untuk membangun sistem *back end* sebuah aplikasi

2.5 Tesseract

Tesseract adalah sebuah *engine* yang menyediakan fasilitas OCR. Tesseract bersifat *open source* dan dapat digunakan secara gratis. Tesseract terdiri dari dua bagian, yaitu sebuah program berbasis baris perintah dan sebuah pustaka yang menyediakan layanan OCR melalui API untuk keperluan pembangunan *back end* perangkat lunak yang membutuhkan OCR. Tesseract mampu menerjemahkan lebih dari seratus bahasa dengan pengaturan dasarnya dan dapat menyediakan hasil ekstraksi dalam format plain text, HTML, PDF dan TSV [16]. Tesseract ditulis dalam bahasa pemrograman C dan C++ dan tersedia pada platform Windows, Linux , macOS, MSYS2,dan Cygwin. Library Tesseract juga tersedia untuk keperluan pembangunan aplikasi berbasis sistem operasi perangkat bergerak seperti Android dan iOS melalui pengembangan yang dilakukan oleh pihak ketiga [17].

Menurut dokumentasi teknis Tesseract [18], cara kerja Tesseract dibagi ke dalam tahap-tahap berikut:

1. Pencarian baris dan kata

   Pada tahap ini, dilakukan pencarian baris-baris teks dan kata dalam sebuah gambar yang mengandung teks yang ingin

dikenali. Pertama, dilakukan pencarian ukuran rata-rata teks pada gambar. Dalam proses ini, dilakukan penyaringan terhadap huruf-huruf yang dapat mengganggu penghitungan rata-rata ukuran teks, seperti huruf yang menyentuh satu sama lain secara vertikal dan huruf yang dibesarkan di awal paragraf (*drop-cap*). Setelah itu, hasil dari penyaringan tersebut dibagi ke dalam baris-baris kata. Baris-baris ini merupakan baris yang paralel dan tidak menimpa satu dengan yang lain, namun melandai untuk mendeteksi peluang gambar teks miring saat dilakukan pengenalan teks. Dari baris-baris tersebut kemudian dicari garis alas teks/*baseline* dari baris-baris teks tadi menggunakan *least median of squares fit*, yang kemudian dilakukan penyesuan terhadap baris-baris kata yang sudah ditemukan sebelumnya agar sesuai dengan *baseline* tadi. Proses penyaringan dan pembagian gambar teks ke dalam baris-baris tadi dibuat sedemikian rupa agar pengenalan teks dapat berjalan dengan baik meskipun gambar teks tersebut miring.

Sesudah *baseline* ditemukan, dilakukan penyesuaian lebih jauh terhadap *baseline* untuk mengantisipasi gambar teks yang bersifat melengkung akibat penjilidan buku atau penyebab-penyebab lainnya. Penyesuaian ini dilakukan dengan menggunakan *quadratic spline*.

Setelah *baseline* disesuaikan dengan kelengkungan gambar teks, Tesseract mencari tahu, apakah huruf- huruf pada barisan kata yang ditemukan memiliki luas yang sama setiap hurufnya (*fixed pitch*). Jika ya, maka proses ini langsung berlanjut ke tahap pengenalan huruf. Jika tidak (luas setiap huruf pada barisan kata bersifat proporsional sesuai dengan jenis *font* huruf yang digunakan pada gambar teks), maka akan dilakukan tahap pengenalan kata.

2. Pengenalan kata

Tahap ini dilakukan khusus untuk kata dengan huruf-huruf yang memiliki luas yang tidak sama (*non-fixed pitch*).

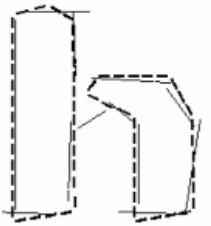
Tahap ini mengelompokan huruf-huruf ke dalam kelompok-kelompok kata. Tahap ini terbagi lagi menjadi dua tahap yaitu tahap pemotongan dan asosiasi. Pada tahap pemotongan, Tesseract mengambil data teks pada setiap baris lalu melakukan pemotongan pada titik-titik tertentu untuk memisahkan huruf-huruf pada sebuah kata. Titik-titik pemotongan yang dipilih merupakan *vertex-vertex* cekung dari garis tepi bentuk poligonal kasar huruf. Gambar 2.3 menunjukkan hasil dari tahap pemotongan yang berupa penggalan-penggalan bentuk poligonal kasar huruf.

Gambar 2.3 Hasil Tahap Pemotongan [18]

Pada tahap asosiasi, penggalan-penggalan huruf dikelompokan untuk mendapatkan kandidat-kandidat huruf. Kandidat huruf adalah kelompok penggalan-penggalan huruf yang diperkirakan menyusun satu huruf yang sama. Proses pengelompokan dilakukan dengan menggunakan metode pencarian A*.

3. Klasifikasi Karakter

Pada tahap ini, dilakukan pengenalan karakter berdasarkan hasil pada tahap 1 untuk kata dengan jenis huruf *fixed pitch* atau tahap 2 untuk kata dengan jenis huruf *non-fixed pitch*. Hasil dari salah satu tahap tersebut diubah menjadi data uji dengan menemukan garis-garis kecil yang memiliki panjang tetap yang diambil dari garis tepi pada hasil tersebut. Data latih yang digunakan dalam klasifikasi ini berupa segmen-segmen garis pada bentuk poligonal huruf yang dipilih menjadi data latih. Gambar 2.4 menunjukkan data latih dan uji pada sistem klasifikasi karakter milik Tesseract.



Gambar 2.4 Visualisasi Data Uji dan Data Latih Dari Karakter "h" pada Sistem Klasifikasi Tesseract [18]

Proses klasifikasi dilakukan setelah data uji dan data latih didapatkan. Proses ini terdiri dari dua tahap. Pertama dilakukan klasifikasi untuk menemukan daftar perkiraan karakter yang mungkin cocok dengan karakter yang sedang diklasifikasikan. Setelah itu, dari daftar tersebut dilakukan pemilihan karakter yang paling cocok dengan karakter yang sedang diklasifikasikan. Pemilihan dilakukan dengan mencari bit vektor dari data latih pada setiap kelas dan mencari kesesuaiannya dengan data fitur.

Data latih yang digunakan pada sistem klasifikasi Tesseract berupa 20 sampel yang berisi 90 karakter kata dengan 8 jenis *font* yang berbeda namun memiliki ukuran yang sama dan memiliki 4 atribut (normal, italic, bold, bold italic)

Pada tugas akhir ini akan digunakan Tesseract untuk Android yang dinamakan tess-two yang dikembangkan oleh pengguna github rmtheis. Versi Tesseract yang digunakan oleh tess-two pada saat tugas akhir ini dibuat adalah versi 3.0.5 [19].

2.6 Yandex. Translate API

Yandex. Translate API adalah sebuah layanan untuk melakukan penerjemahan kata yang berbentuk data teks melalui sistem API yang disediakan oleh sebuah perusahaan teknologi

bernama Yandex. Yandex. Translate API mendukung lebih dari 90 bahasa [20].

Yandex. Translate API menggunakan JSON/JSONP dan XML sebagai antarmukanya. Pengguna dapat memilih salah satu di antara dua antarmuka tersebut untuk menggunakan Yandex. Translate API. Yandex. Translate API digunakan dengan cara melakukan permintaan HTTP ke alamat https://translate.yandex.net/api/v1.5/tr.json/translate dengan menyertakan sejumlah parameter. Tabel 2.1 menunjukkan parameter-parameter yang disertakan dalam permintaan HTTP Yandex. Translate API.

Tabel 2.1 Parameter Permintaan HTTP Yandex. Translate API

Nama Parameter	Keterangan
key	Kunci API yang tersedia secara gratis dan bisa didapatkan di https://tech.yandex.com/translate/
text	Kata yang ingin diterjemahkan
lang	Kode dua huruf dari bahasa yang digunakan oleh kata yang ingin diterjemahkan dan bahasa yang menjadi sasaran penerjemahan.
format	Format dari kata yang ingin diterjemahkan. Nilainya dapat berisi “plain” untuk teks biasa atau "html” untuk teks dengan format HTML yang memungkinkan pengguna untuk menerjemahkan halaman situs web. Bersifat opsional.
options	Berisi angka satu apabila pengguna ingin menyertakan bahasa hasil deteksi bahasa

	otomatis apabila pengguna hanya menyertakan bahasa tujuan pada parameter “lang”. Bersifat opsional.
callback	Nama fungsi callback yang digunakan khusus untuk penggunaan dengan antarmuka JSONP. Bersifat opsional.

Yandex Translate API tersedia secara gratis untuk keperluan penerjemahan sepuluh juta karakter per bulan dan satu juta karakter perhari [21] . Untuk keperluan penerjemahan di atas angka tersebut, Yandex. Translate memungut biaya mulai dari 15 Dolar Amerika per satu juta karakter untuk kapasitas kurang dari lima puluh juta karakter per bulan hingga 6 Dolar Amerika per satu juta karakter untuk kapasitas di atas lima ratus juta karakter hingga satu miliar karakter per bulan [22].

2.7 Filter Sobel

Filter Sobel adalah sebuah filter gambar yang digunakan untuk mendeteksi tepi di dalam sebuah gambar. Filter Sobel mencari tepi dengan cara menemukan perkiraan derajat gradien absolut dari setiap titik pada sebuah gambar *grayscale* yang menjadi masukan dari filter tersebut [23]. Filter Sobel memiliki dua jenis *kernel*, yaitu *kernel* untuk menemukan derajat komponen gradien dalam sumbu x (Gx) dan *kernel* untuk menemukan derajat komponen gradien dalam sumbu x (Gy). Gambar 2.5 menunjukkan filter Gx dan Gy yang digunakan oleh filter Sobel.

-1	0	+1
-2	0	+2
-1	0	+1

(a)

+1	+2	+1
0	0	0
-1	-2	-1

(b)

Gambar 2.5 Kernel Gx (a) dan Gy (b) yang Digunakan dalam Filter Sobel

Jika diketahui sebuah gambar *grayscale* seperti gambar 2.6 berikut:

a	b	c
d	e	f
g	h	i

Gambar 2.6 Matriks Gambar Contoh

Maka derajat komponen gradien sumbu x pada titik e (koordinat 2,2) pada gambar 2.x adalah:

$$Gx[2,2] = -a + c - 2d + 2f - g + i \qquad (2.1)$$

Dan derajat komponen gradien sumbu y pada titik e adalah:

$$Gy[2,2] = a + 2b + c + -g - 2h - i \qquad (2.2)$$

Filter Gx digunakan untuk menemukan tepi-tepi vertikal, sementara filter Gy digunakan untuk menemukan tepi-tepi horizontal [23]. Kedua komponen gradien Gx dan Gy dapat digunakan untuk menemukan perkiraan derajat gradien G dari sebuah titik, dengan rumus:

$$|G| = \sqrt{G_x^2 + G_y^2} \quad (2.3)$$

2.8 Ruang Warna HSV

HSV adalah salah satu model ruang warna untuk merepresentasikan suatu warna tertentu. Ruang warna HSV dibuat untuk membuat representasi warna yang sesuai dengan pola pikir manusia untuk melakukan persepsi terhadap warna, yaitu dengan memisahkan antara warna dengan tingkat kecerahan dari warna tersebut [24]. Menurut Youseff [25], ruang warna HSV bagus dipakai dalam pengolahan citra karena ruang warna HSV mampu memisahkan antara komponen kromatis dan akromatis.

Nilai H (*Hue*) merepresentasikan warna asli dari suatu warna. Nilai Hue berada pada jangkauan nilai 0^o sampai 360^o. Pada pustaka OpenCV, nilai Hue berada pada jangkauan nilai 0 sampai 179 [26]. Nilai S (*Saturation*) merepresentasikan tingkat kejenuhan dari suatu warna. Nilai V (Value) merepresntasikan tingkat kecerahan dari suatu warna. Nilai Saturation dan Value berada pada jangkauan nilai 0 sampai 1. Pada pustaka OpenCV, nilai Saturation dan Value berada pada jangkauan nilai 0 sampai 255 [26].

2.9 Histogram Intensitas

Histogram intensitas adalah histogram yang menggambarkan jumlah piksel dalam sebuah gambar yang memiliki instensitas dengan nilai tertentu [27]. Histogram intensitas biasanya dibuat dari gambar *grayscale* sebuah gambar, yang menunjukkan intensitas warna dari setiap piksel. Contohnya dalam sebuah gambar *grayscale*, nilai intensitas sebuah piksel dapat bernilai 0 hingga 256. Histogram intensitas *grayscale* akan menunjukkan jumlah piksel pada sebuah gambar yang memiliki intensitas 0 hingga 256. Selain gambar *grayscale*, histogram intensitas juga dapat dicari dari gambar dengan *channel* lebih dari

satu, contohnya gambar RGB. Pada histogram gambar RGB, histogram dapat dibuat dari masing-masing *channel* Red, Green, atau Blue. Histogram juga dapat dibuat menggunakan tiga *channel* warana tersebut sekaligus, membentuk sebuah histogram tiga dimensi [27].

2.10 Histogram Correlation

Histogram Correlation merupakan salah satu rumus yag digunakan pada fungsi compHist di dalam pustaka OpenCV. Pada pustaka OpenCV, fungsi compHist digunakan untuk mencari derajat kecocokan dari dua histogram yang berbeda [28]. Nilai yang dihasilkan oleh fungsi compHist yang menggunakan rumus Histogram Correlation akan berupa nilai desimal dengan derajat kecocokan maksimal 1 untuk dua histogram yang sama persis.

Rumus Histogram Correlation memiliki pendefinisian sebagai berikut [28]:

$$d(H_1, H_2) = \frac{\sum_I (H_1(I) - \overline{H_1})(H_2(I) - \overline{H_2})}{\sqrt{\sum_I (H_1(I) - \overline{H_1})^2 \sum_I (H_2(I) - \overline{H_2})^2}} \quad (2.4)$$

$$\overline{H_k} = \frac{1}{N} \sum_J H_k(J) \quad (2.5)$$

Dimana:
H_1 = Histogram pertama
H_2 = Histogram kedua
k = Nomor urut histogram
I, J = Nomor urut bin histogram
N = Jumlah bin histogram

2.11 Morfologi *Opening*

Morfologi *opening* adalah salah satu morofologi matematika yang merupakan turunan dari morfologi erosi dan dilasi [29]. Morfologi *opening* merupakan morfologi erosi yang

diikuti dengan morfologi dilasi dengan menggunakan ukuran kernel yang sama untuk masing-masing operasi. Morfologi ini biasanya digunakan untuk gambar biner atau gambar grayscale [29]. Morfologi ini biasanya digunakan untuk menghilangkan titik-titik *noise* pada gambar biner. Gambar 2.7 menunjukkan contoh gambar sebelum dan sesudah diberi morfologi *opening*

(a) (b)

Gambar 2.7 Contoh Gambar Sebelum (a) dan Sesudah (b) Dilakukan Morfologi Opening [30]

(Halaman ini sengaja dikosongkan)

BAB III
ANALISIS DAN PERANCANGAN

Pada bab ini, akan dibahas analisis dan perancangan aplikasi penerjemah gambar teks berbahasa Inggris menggunakan teknologi realitas tertambah pada perangkat berbasis Android. Pembahasan meliputi analisis fitur yang dibutuhkan dan perancangan perangkat lunak.

3.1 Analisis Sistem

Pada Bab I, telah disebutkan bahwa aplikasi dibuat menggunakan tekonlogi realitas tertambah. Teknologi realitas tertambah dipilih agar masukan gambar yang mengandung kata yang ingin diterjemahkan dapat diambil secara langsung (*real time*) melalui kamera perangkat bergerak pengguna. Pengambilan gambar masukan melalui kamera bertujuan untuk mengurangi waktu pengguna dalam memasukan kata yang ingin diterjemahkan dibandingkan dengan cara input lain seperti mengetikan kata. Pengguna juga dapat dengan cepat memilih kata lain yang ingin diterjemahkan dengan langsung menggerakan kamera perangkat ke kata yang diinginkan dan memilih kata tersebut.

Cara pengguna untuk berinteraksi dengan aplikasi adalah dengan memilih kata yang diinginkan pada gambar dari kamera perangkat yang ditampilkan di layar perangkat pengguna. Pemilihan kata dilakukan dengan cara menyentuh pada kata yang diinginkan oleh pengguna.

Untuk dapat mengolah gambar kata yang didapatkan dari kamera perangkat berdasarkan pilihan pengguna, perlu dilakukan pengolahan gambar hingga didapatkan data string dari gambar kata yang dipilih pengguna. Data string ini diperlukan agar dapat diketahui kata apa yang terdapat pada gambar yang dipilih pengguna. Untuk mendapatkan gambar kata berdasarkan pilihan pengguna, akan digunakan OpenCV sebagai pustaka pengolah gambar. Untuk dapat mengenali kata dari gambar kata yang sudah

diambil berdasarkan pilihan pengguna, akan digunakan Tesseract untuk melaukan pengenalan kata. Tesseract merupakan pustaka yang dapat melakukan pengenalan karakter optis (Optical Character Recognition/OCR), sehingga dapat mengenali kata yang terdapat pada sebuah gambar. Tesseract yang digunakan dalam pembangunan aplikasi ini adalah Tesseract versi tess-two. Tesseract versi tess-two merupakan pustaka Tesseract yang dibangun khusus untuk digunakan dalam pembuatan aplikasi Android dan tersedia dalam bahasa pemrogaman Java.

Untuk melakukan penerjemahan kata, digunakan Yandex. Translate API. API ini memungkinkan penggunannya untuk mengirimkan kata ke Yandex Translate melalui jaringan Internet untuk diterjemahkan ke bahasa yang diinginkan dan Tesseract akan mengembalikan hasilnya dalam bentuk data string.

Untuk melakukan augmentasi pada gambar dari kamera diperlukan pelacak untuk dapat mengenali gambar kata yang dipilih pengguna dan mengetahui posisi dari gambar tersebut gambar pada kamera mengenali perubahan, misalnya ketika kamera digerakan oleh pengguna. Pelacak tersebut dibagun menggunakan pustaka OpenCV sebagai pustaka pengolahan gambar. Augmentasi/penggambaran hasil terjemahan juga akan dilakukan menggunakan pustaka OpenCV. Pustaka OpenCV dipilih untuk penggambaran karena objek yang akan digambar hanya berupa bentuk dua dimensi sederhana. Pustaka OpenCV yang digunakan dalam pembangunan aplikasi ini adalah OpenCV4Android. OpenCV4Android dalah pustaka yang dikhususkan untuk membangun aplikasi Android dan tersedia dalam bahasa pemrograman Java dan C/C++

Perangkat lunak dalam tugas akhir ini dibangun menggunakan aplikasi Android Studio. Aplikasi ini merupakan *integrated development environment* (IDE) yang dikhususkan untuk membangun aplikasi berbasis sistem operasi Android. Selain bahasa pemrograman Java yang menjadi bahasa pemrograman utama dalam pemrograman aplikasi Android, Android Studio juga mendukung bahasa pemrograman C/C++,

sehingga pustaka pendukung yang tersedia dalam bahasa C/C++ seperti OpenCV versi bahasa pemrograman C dapat digunakan untuk membangun aplikasi Android. Kelebihan dari penggunaan bahasa pemrograman C/C++ dalam pemrograman aplikasi Android adalah dapat memanfaatkan performa dari perangkat Android secara lebih, yang dibutuhkan dalam aplikasi dengan tingkat kegiatan komputasi yang lebih tinggi [31]. Kemampuan tersebut dibutuhkan dalam pembangunan aplikasi ini karena aplikasi ini membutuhkan komputasi grafis yang dapat dilakukan dengan cepat agar aplikasi tidak lambat saat digunakan.

3.2 Perancangan Perangkat Lunak

Pada subbab ini, akan dibahas deskripsi umum perangkat lunak, spesifikasi kebutuhan fungsional dan kebutuhan non-fungsional serta bagaimana karakteristik pengguna permainan.

3.2.1 Deskripsi Umum Perangkat Lunak

Tugas akhir yang dibuat adalah aplikasi penerjemah gambar kata berbahasa Inggris menggunakan teknologi realitas tertambah pada perangkat berbasis Android. Permainan ini dibuat dengan menggunakan pustaka OpenCV sebagai pustaka pengolahan gambar, Tesseract versi tess-two sebagai pustaka pengenal karakter optis (OCR), dan Yandex. Translate API sebagai kakas bantu penerjemahan kata.

Sebagai antarmuka aplikasi dengan pengguna, aplikasi menampilkan gambar yang diambil secara langsung dari kamera pengguna. Pengguna memilih kata yang ingin diterjemahkan dengan menyentuh gambar dari kata yang ditampilkan pada tampilan kamera. Gambar kata yang dipilih pengguna akan diterjemahkan dan ditampilkan pada tampilan kamera dengan melakukan penambahan grafis pada setiap *frame* gambar yang diambil oleh kamera dengan gambar hasil terjemahan. Setelah hasil terjemahan di tampilkan dan pengguna membaca hasil terjemahan tersebut, pengguna dapat memilih kata lainnya untuk diterjemahakn dengan cara yang sama (menyentuh gambar kata

yang ingin diterjemahkan). Penerjemahan yang dapat dilakukan oleh aplikasi ini adalah penerjemahan kata bahasa Inggris menjadi bahasa Indonesia dan hanya satu kata dalam satu kali penerjemahan. Ketika pengguna memilih kata yang lain untuk diterjemahkan, hasil terjemahan kata sebelumnya akan hilang.

Setelah pengguna menyentuh gambar kata yang ingin diterjemahakan pada tampilan kamera aplikasi, aplikasi akan mengambil gambar kata tersebut dan menjadikannya *region of interest* (ROI). ROI terebut akan menjadi masukan pada dua proses dalam aplikasi yang dimulai dalam waktu yang bersamaan. Proses pertama adalah pelacakan, yaitu proses pendeteksian posisi ROI pada setiap *frame* kamera yang masuk ke dalam perangkat. Proses pelacakan dilakukan untuk mencari posisi kata untuk selanjutnya menjadi posisi untuk melakukan penggambaran hasil. Proses kedua adalah proses penerjemahan. Proses ini akan mengenali kata yang terdapat dalam ROI dan mengirim kata tersebut ke kakas bantu penerjemah. Hasil terjemahan kemudian akan diubah menjadi objek Mat OpenCV untuk kemudian digambarkan pada setiap *frame* tampilan kamera. Hasil yang digambarkan akan berupa persegi panjang yang berisi kata hasil terjemahan, yang diletakan di atas sebuah persegi panjang yang mengelilingi kata yang dipilih oleh pengguna.

3.2.2 Spesifikasi Kebutuhan Fungsional

Berdasarkan deskripsi umum sistem yang disampaikan sebelumnya, maka kebutuhan fungsional dalam permainan ini adalah pengguna dapat memilih kata yang ingin diterjemahkan dan melihat hasil terjemahan dari kata tersebut pada tampilan kamera.

3.2.3 Spesifikasi Kebutuhan Non-Fungsional

Pada aplikasi ini, terdapat beberapa kebutuhan non-fungsional yang bertujuan meningkatkan kualitas dari aplikasi

tersebut. Berikut daftar kebutuhan non-fungsional dari aplikasi yang dibuat dalam tugas akhir ini:

1. Grafis

Aplikasi harus menampilkan hasil terjemahan dengan jelas dan mudah dibaca oleh pengguna.

2. Performa

Aplikasi yang dibuat harus memperhatikan performa agar pada saat aplikasi digunakan oleh pengguna, aplikasi dapat memberikan hasil terjemahan yang baik dan nyaman ketika digunakan oleh pengguna. Aplikasi secara keseluruhan harus dapat berjalan pada perangkat bergerak berbasis Android dengan kecepatan yang tidak mengganggu kenyamanan pengguna dalam menggunakan aplikasi, namun tidak mengorbankan kemampuan dari pengolahan grafis yang dilakukan oleh aplikasi.

3.2.4 Karakteristik Pengguna

Berdasarkan deskripsi umum diatas, maka diketahui bahwa pengguna yang akan menggunakan permainan ini hanya satu orang, yaitu pengguna yang menjalankan permainan. Karakteristik pengguna tercantum dalam Tabel 3.1

Tabel 3.1 Karakteristik Pengguna

Nama Aktor	Tugas	Hak Akses Aplikasi	Kemampuan yang harus dimiliki
Pengguna	Pihak luar yang menggunakan aplikasi	Menjalankan dan menggunakan aplikasi	Pernah menggunakan perangkat berbasis Android.

3.2.5 Perancangan Alur Interaksi Pengguna dengan Aplikasi

Subbab ini akan membahas bagaimana aplikasi berinteraksi dengan penggunanya. Pertama-tama, pengguna membuka aplikai pada perangkat bergerak berbasis sistem operasi Android. Aplikasi akan menampilkan tampilan kamera perangkat bergerak tersebut. Setelah itu, pengguna mengarahkan perangkat ke gambar kata dalam bahasa Inggris yang ingin diterjemahkan ke dalam bahasa Indonesia. Setelah gambar yang dimaksud tampil pada tampilan kamera aplikasi, pengguna menyuntuh gambar kata tersebut. Sentuhan dilakukan pada salah satu huruf pada bagian tengah (bukan pada huruf bagian tepi) dari gambar kata tersebut. Setelah itu, aplikasi akan menampilkan persegi yang menandai gambar kata yang dipilih. Aplikasi kemudian menampilkan hasil penerjemahan pada layar perangkat yang dapat dibaca oleh pengguna. Selanjutya pengguna dapat melakukan penerjemahan pada gambar kata lainnya dan proses pun diulangi hingga pengguna keluar dari aplikasi. Gambar 3.1 menampilkan alur dari proses interaksi pengguna dengan aplikasi.

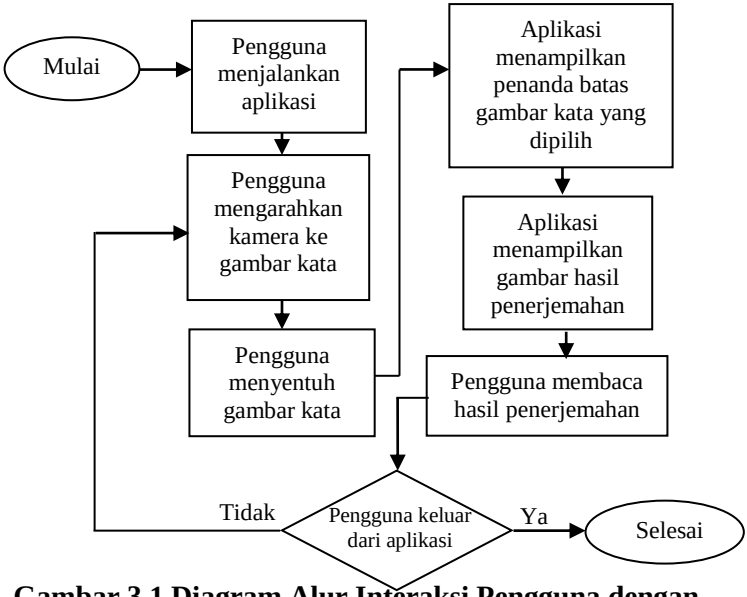
3.2.6 Perancangan Arsitektur Aplikasi

Subbab ini akan membahas arsitektur dari aplikasi yang dibuat dalam tugas akhir ini. Arsitektur yang dirancang berupa diagram alur yang dapat dibagi menjadi tiga bagian, yaitu pengambilan gambar kata yang akan diterjemahkan sebagai *region of interest* (ROI), penerjemahan kata, dan penambahan tampilan kamera perangkat dengan hasil terjemahan.

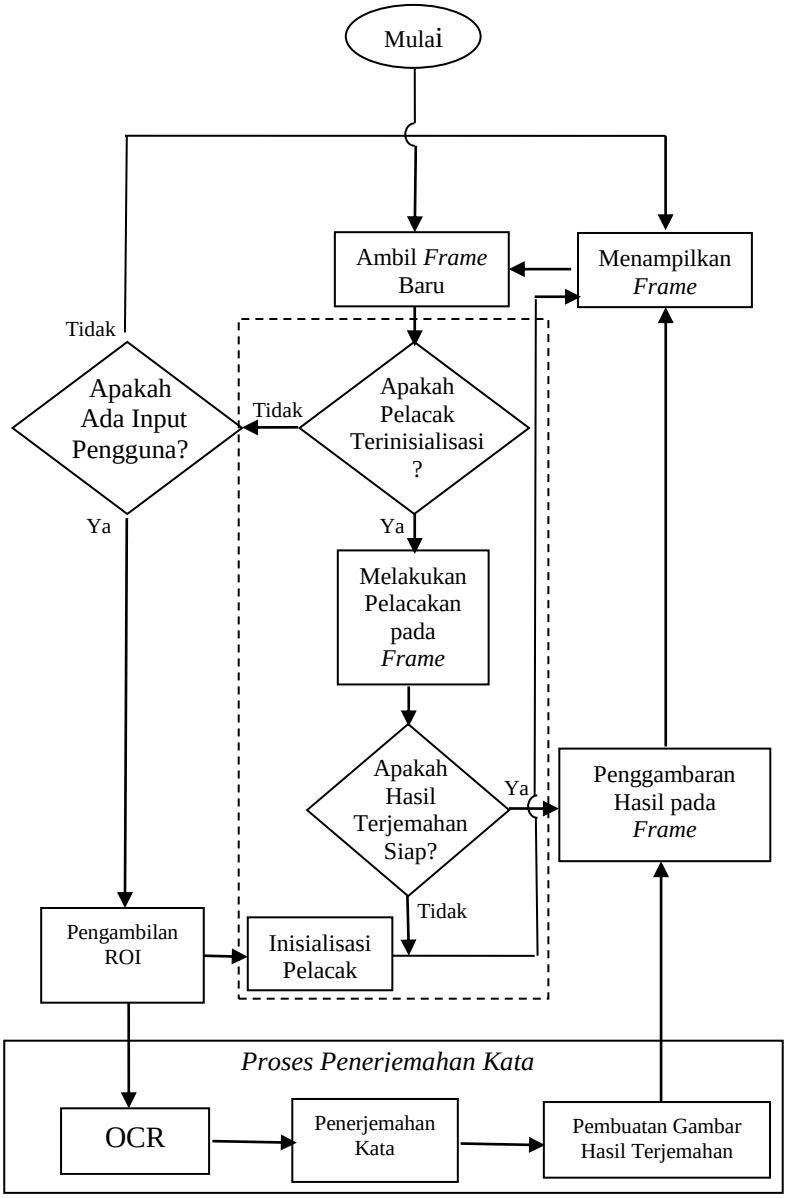
Ketika pengguna membuka aplikasi, aplikasi akan mengambil *frame* baru dari kamera perangkat. Aplikasi akan menunggu masukan pengguna berupa sentuhan. Selama pengguna tidak memberikan masukan, aplikasi akan terus mengambil *frame* baru dari kamera perangkat dan menampilkannya pada layar perangkat. Apabila pengguna memberikan masukan berupa sentuhan pada kata yang akan diterjemahkan, Aplikasi akan

mengambil area gambar kata yang dipilih sebagai ROI. ROI tersebut akan menjadi masukan untuk proses penerjemahan kata. Pada bagian tersebut, kata dikenali dari ROI menggunakan OCR untuk kemudian diterjemahkan. Setelah hasil penerjemahan didapatkan, aplikasi akan membuat gambar yang berisikan hasil penerjemahan yang akan ditampilkan pada layar aplikasi. Setelah itu, aplikasi akan menggambarkan hasil terjemahan di atas posisi ROI hasil pelacakan.

Pelacak ROI diaktifkan setelah area ROI didapatkan. Pelacak akan mendeteksi posisi ROI pada *frame* masukan dari kamera perangkat. Pada setiap *frame*-nya, pelacak akan menunggu gambar hasil penerjemahan dari bagaian penerjemahan kata. Apabila gambar hasil penerjemahan sudah siap, aplikasi akan menggambarkan hasil penerjemahan pada *frame*. Gambar 3.2 menampilkan diagram alur arsitektur aplikasi secara keseluruhan.



Gambar 3.1 Diagram Alur Interaksi Pengguna dengan Aplikasi



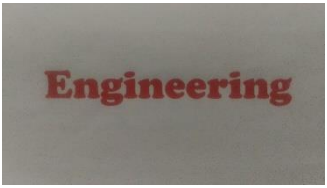
Gambar 3.2 Diagram Alur Arsitektur Aplikasi

3.2.6.1 Rancangan Proses Pengambilan Gambar Kata yang Akan Diterjemahkan Sebagai *Region of Interest* (ROI)

Pada bagian ini dibuat rancangan tentang bagaimana aplikasi akan mengambil gambar kata dari *frame* gambar yang didapatkan dari kamera perangkat pengguna untuk digunakan sebagai *region of interest* (ROI). Masukan dari proses ini adalah *frame* gambar yang didapatkan dari kamera perangkat pengguna dan posisi sentuhan pengguna pada layar perangkat pengguna.

Pertama-tama, *frame* gambar yang menjadi input pada proses ini diubah menjadi *grayscale* dan diberikan filter Sobel *Gx* dan *Gy* seara terpisah yang lalu dimasukkan ke dalam dua objek Mat OpenCV yang berbeda (kedua objek Mat ini selanjutnya disebut *Ix* dan *Iy*) . Mat *Ix* akan digunakan untuk mencari tinggi huruf. Hal ini dikarenakan *Ix* mendeteksi gradien horizontal pada gambar, yang juga menunjukkan tepian vertikal pada gambar. Sementara objek Mat *Iy* akan digunakan untuk mencari panjang kata karena *Iy* menangkap gradien vertikal pada gambar, yang juga menunjukkan tepian horizontal pada gambar.

Setelah *Ix* dan *Iy* didapatkan, diberikan *threshold* pada *Ix* dan *Iy* yang bertujuan untuk menghilangkan *noise* yang dapat mengganggu proses pengambilan gambar kata. Karena *Ix* dan *Iy* masih berupa gambar *grayscale*, *thresholding* yang dipilih merupakan nilai pada jangkauan nilai warna grayscale (0-256). Pada tugas akhir ini dipilih *threshold* 15. Nilai ini dipilih berdasarkan eksperimen, di mana dicari *threshold* yang mampu menghilangkan *noise*, namun tidak memperburuk hasil pendeteksian tepi gambar kata . Piksel yang memiliki nilai lebih dari sama dengan 15 akan diubah menjadi warna putih (256) dan sisanya menjadi warna hitam (0). Setelah itu, dilakukan morfologi *opening* pada *Ix* dan *Iy* untuk menghilangkan *noise* lebih lanjut. Gambar 3.3 menunjukkan *frame* gambar yang didapatkan dari kamera perangkat milik pengguna dan hasilnya setelah diberi filter Sobel *Gx* dan *Gy* secara terpisah serta thresholding dan morfologi opening.



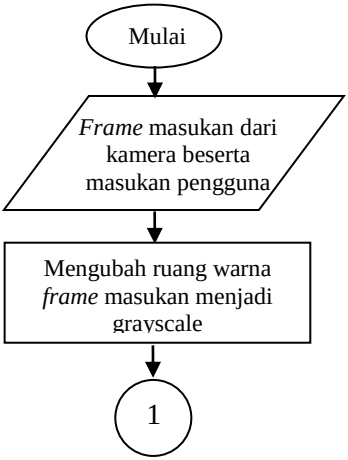
(a)

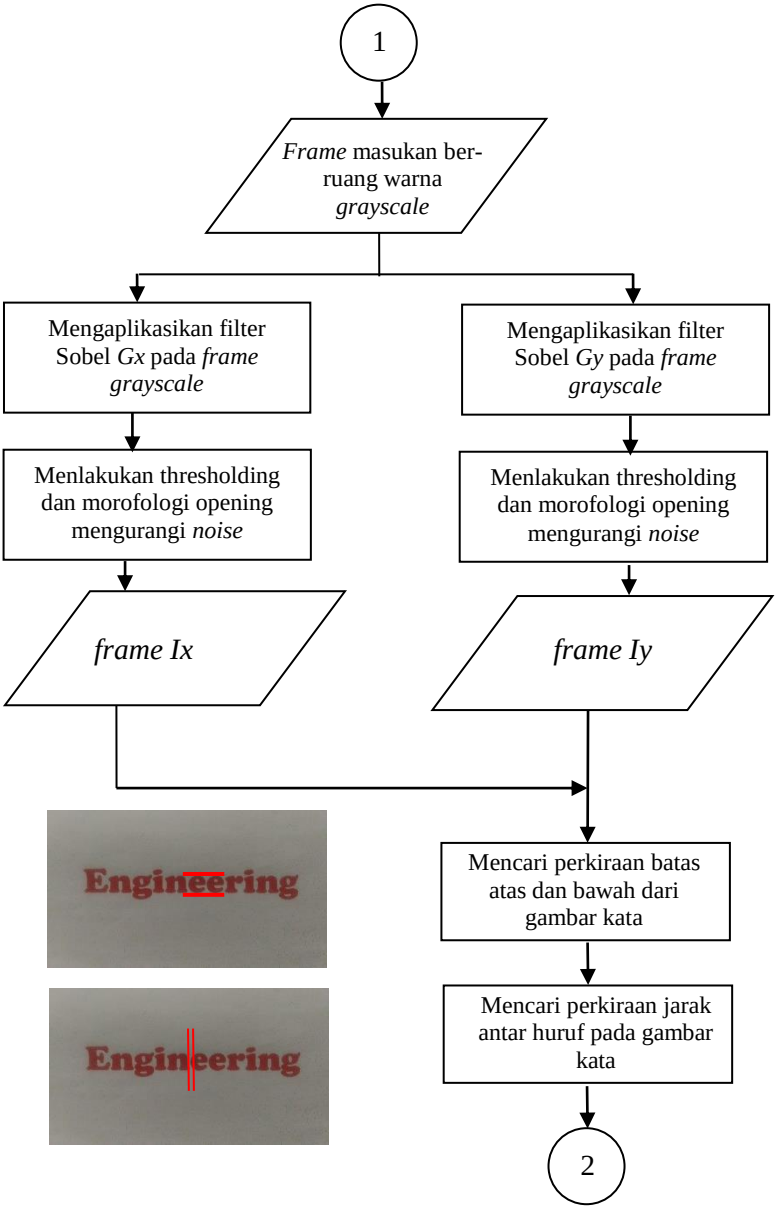
(b)

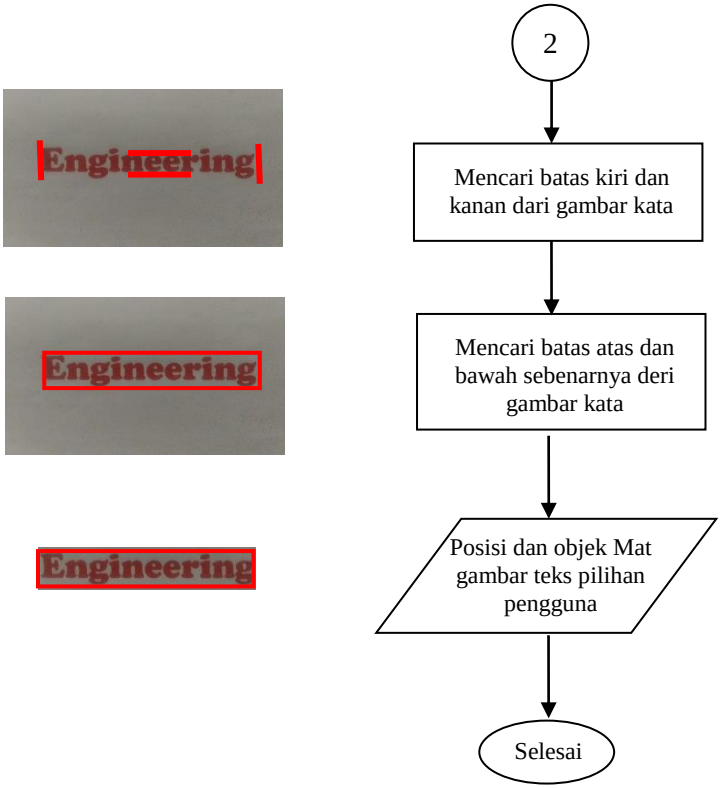
(c)

Gambar 3.3 Contoh Frame Gambar dari Kamera Perangkat Sebelum (a) dan Sesudah Diberi Sobel *Gx* (b) dan *Gy* (c)

Selanjutnya, akan dilakukan pendeteksian gambar kata dari *frame* gambar kamera. Gambar 3.4 menjelaskan tahap-tahap pendeteksian area gambar teks.




1
Frame masukan ber-ruang warna *grayscale*
Mengaplikasikan filter Sobel *Gx* pada *frame grayscale*
Mengaplikasikan filter Sobel *Gy* pada *frame grayscale*
Menlakukan thresholding dan morofologi opening mengurangi *noise*
Menlakukan thresholding dan morofologi opening mengurangi *noise*
frame Ix
frame Iy
Engineering
Engineering
Mencari perkiraan batas atas dan bawah dari gambar kata
Mencari perkiraan jarak antar huruf pada gambar kata
2



Gambar 3.4 Diagram Alur Proses Pengambilan Gambar Kata

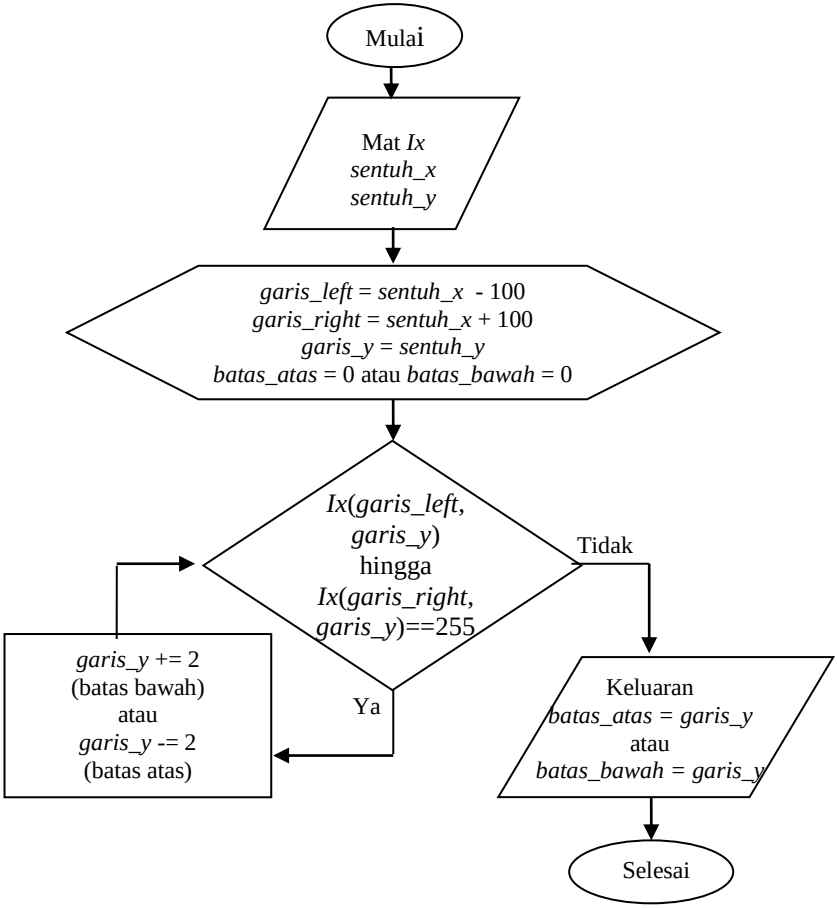
Setelah *frame Ix* dan *Iy* didapatkan, pertama-tam dilakukan pencarian perkiraan tinggi kata. Untuk tahap ini, akan digunakan objek Mat *Ix*. Pertama, akan dibuat sebuah garis horizontal imajiner dengan panjang 200 piksel yang berpusat pada posisi sentuhan pengguna. Ukuran panjang garis dipilih berdasarkan eksperimen dan fakta bahwa pengukuran pada tahap ini baru bersifat perkiraan. Pada garis tersebut, akan dilakukan iterasi pergeseran sebanyak 2 piksel ke arah sumbu Y negatif pada objek Mat *Ix*. Jarak pergeseran pada setiap iterasinya dipilih

berdasarkan eksperimen dan memperhatikan sifat-siat huruf tertentu seperti tanda titik pada karakter “i” dan “j”. Pada setiap iterasi, program akan memeriksa apakah disepanjang garis imajiner tersebut terdapat piksel yang merupakan tepian vertikal (yang ditandai dengan piksel bernilai 255 berdasakan hasil thresholding pada tahap sebelumnya). Iterasi akan berhenti setelah tidak lagi ditemukan piksel tepian vertikal. Hasil dari iterasi ini adalah batas atas dari gambar kata masukan. Tahap ini akan diulangi ke arah sumbu Y positif untuk mendapatkan batas bawah dari gambar kata masukan. . Tinggi kata didapatkan dari jarak antara batas atas ke batas bawah. Gambar 3.5 menunjukkan alur dari proses mendapatkan batas-batas ketinggian gambar kata.

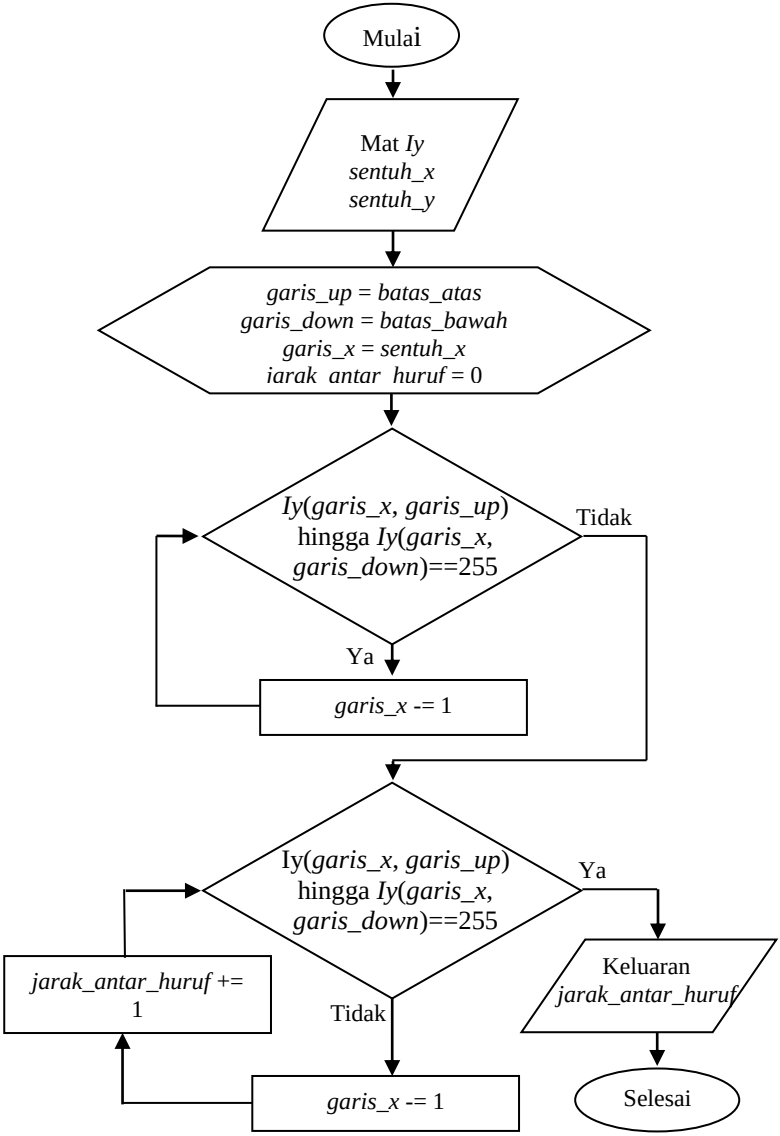
Selanjutnya, dilakukan pencarian jarak antar huruf pada gambar kata masukan. Pencarian dilakukan dengan menggunakan iterasi pergeseran garis yang serupa dengan tahap sebelumnya. Namun, masukan pada tahap ini menggunakan objek Mat *Iy* untuk mendeteksi tepian horizontal, dan garis imajiner yang digunakan berupa garis vertikal dengan panjang sesuai dengan perkiraan tinggi kata yang didapatkan pada tahap sebelumnya. Pergeseran garis pada setiap iterasinya dilakukan sebanyak 1 piksel ke arah sumbu X negatif. Iterasi dimulai dari posisi sentuhan pengguna. Jika pada saat iterasi tidak ditemui tepian horizontal lagi, iterasi akan dilanjutkan dengan beberapa perbedaan. Iterasi akan berjalan hingga kembali menemui tepian horizontal Pada iterasi bagian ini, jarak antar tepian horizontal tersebut akan diukur dan didapatkan jarak antar huruf. Gambar 3.6 menunjukkan alur dari proses tersebut.

Selanjutnya, dilakukan pencarian lebar gambar kata. Sama seperti pada tahap pencarian jarak antar huruf, tahap ini menggunakan objek Mat *Iy* untuk mendeteksi tepian horizontal, dan dan garis imajiner yang digunakan berupa garis vertikal dengan panjang sesuai dengan perkiraan tinggi kata. Pergesaran garis akan dilakukan sebanyak 1 piksel ke arah sumbu X negatif dan berhenti ketika iterasi menemui ruang kosong. Saat iterasi berhenti, garis imajiner akan digeser sebanyak jarak antar huruf

yang didapatkan pada tahap sebelumnya. Hal ini dilakukan untuk menangani ruang kosong antar huruf pada gambar kata. Bila setelah pergeseran ditemukan tepian, iterasi akan dilanjutkan kembali. Karena pada suatu kata jarak antar huruf mungkin tidak



Gambar 3.5 Diagram Alur Proses Pencarian Batas-Batas Vertikal Gambar Kata



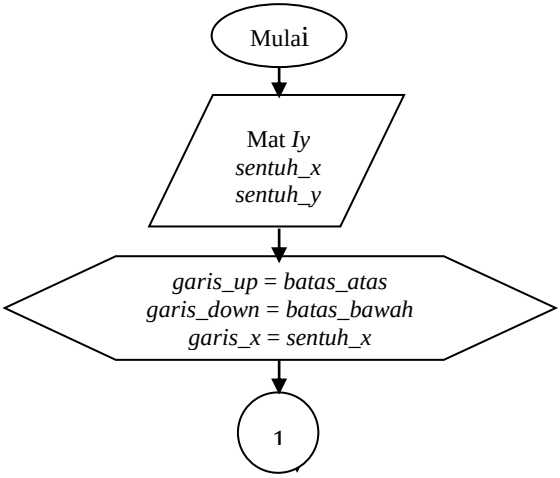
Gambar 3.6 Diagram Alur Proses Pencarian Jarak Antar Huruf Pada Gambar Kata Masukan

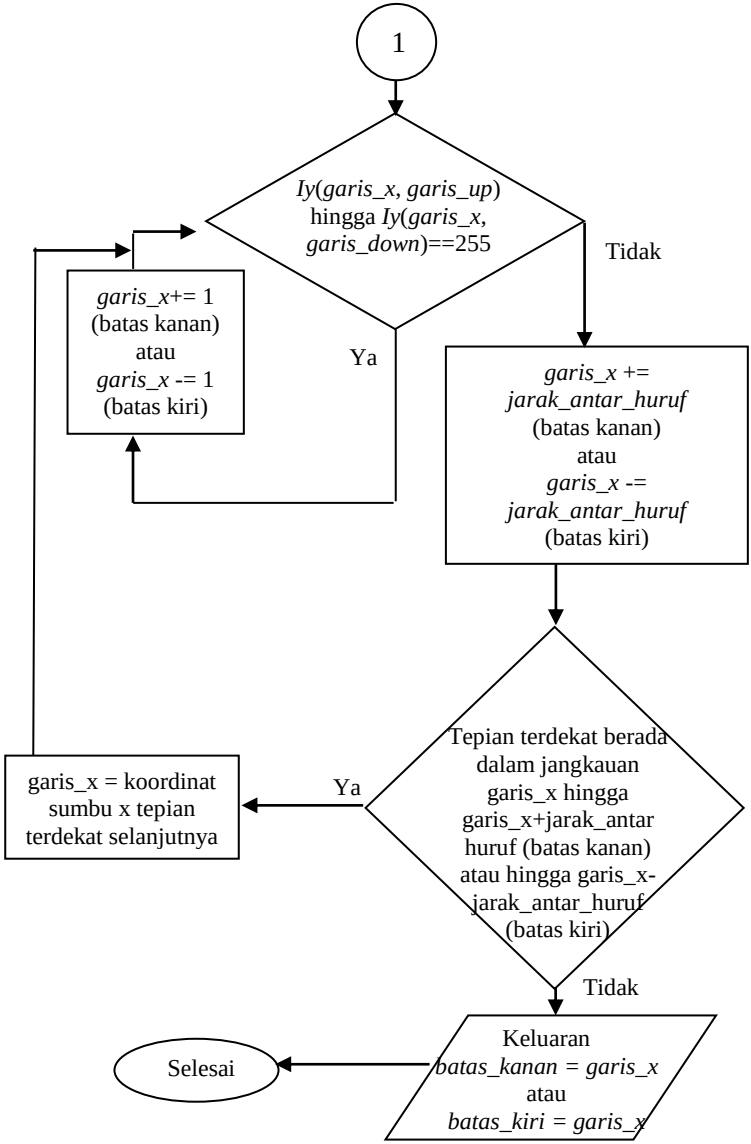
sama, tepian akan tetap diterima jika berada dalam jarak tambahan maksimal sebanyak jarak antar huruf.

Tahap ini akan diulangi ke arah sumbu X positif untuk mendapatkan batas kanan dari gambar kata masukan. Lebar kata didapatkan dari jarak antara batas atas ke batas bawah. Gambar 3.7 menunjukkan alur dari proses tersebut.

Selanjutnya, akan dilakukan pencarian ulang terhadap batas atas dan bawah kata. Proses ini akan dilakukuan sama seperti tahap pencarian perkiraan tinggi kata, namun dengan panjang garis imajiner sesuai dengan panjang kata yang didapatkan dari tahap sebelumnya. Hal ini dilakukan untuk menangani karakter yang tidak terdeteksi pada tahap pencarian tinggi kata perkiraan, seperti huruf kapital atau karakter tertentu seperti karakter "j" dan "g" yang terlewat. Tahap ini akan menghasilkan tinggi kata yang sesungguhnya.

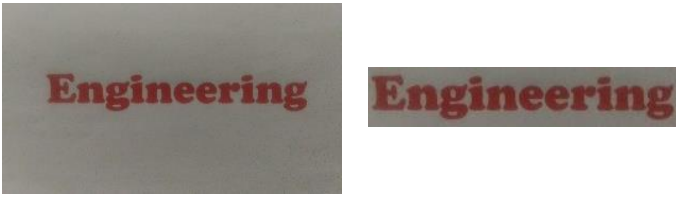
Setelah batas atas atas,bawah,kiri dan kanan dari gambar kata didapatkan, gambar kata diambil dari *frame* gambar masukan dari kamera perangkat dengan cara membentuk persegi berdasarkan nilai batas-batas gambar kata dan posisi sentuhan pengguna. Persegi ini digunakan untuk memotong *frame* gambar masukan dari kamera pada daerah tempat persegi tersebut berada.





Gambar 3.7 Diagram Alur Proses Pencarian Batas-Batas Horizontal Gambar Kata

Hasil akhir dari proses ini adalah gambar ROI beserta persegi *bounding box* tempat ROI berada. Gambar 3.8 menunjukkan hasil akhir dari proses pengambilan gambar kata yang akan diterjemahkan.



(a) (b)

Gambar 3.8 Contoh Frame Masukan (a) dan ROI yang Diambil dari Frame Masukan Tersebut (b)

3.2.6.2 Rancangan Proses Penerjemahan Gambar Kata

Pada bagian ini dibuat rancangan tentang bagaimana aplikasi akan mengenali kata berdsarkan gambar dari tahap sebelumnya dan menerjemahkannya. Masukan dari proses ini adalah gambar ROI yang didapatkan pada proses pengambilan gambar kata. Proses ini akan dijalankan pada *thread background* aplikasi agar tampilan kamera aplikasi dapat tetap berjalan tanpa harus menunggu penerjemahan selesai. Proses ini dibagi menjadi tiga tahap.

Pada tahap pertama, gambar ROI akan dimasukan ke OCR Tesseract untuk proses pengenalan huruf yang terdapat pada gambar tersebut. Untuk melakukan proses OCR, perlu dilakukan inisialisasi Tesseract terlebih dahulu. Inisialisasi Tesseract dilakukan dengan melakukan pemuatan data latih untuk pengenalan kata dalam bahasa Inggris. Keluaran dari proses ini adalah data String hasil pengenalan kata yang terdapat pada gambar kata masukan.

Tahap kedua, data String hasil penerjemahan akan dikirmkan ke Yandex Translate melalui Yandex. Translate API. Kata akan diterjemahkan dari bahasa Inggris ke bahasa Indonesia.

Pengiriman dilakukan dengan membuat sebuah request HTML ke alamat "https://translate.yandex.net/api/v1.5/tr.json/translate" dengan menyertakan parameter-parameter yang dijelaskan pada Bab II. Parameter yang akan digunakan dalam melakukan request HTML ke Yandex. Translate pada aplikasi ini adalah "key", "text", "lang", dan "format". Nilai untuk parameter "key" didapatkan dengan membuat kunci API pada situs "https://tech.yandex.com/translate/". Nilai parameter "text" menggunakan data String hasil tahap pengenalan kata. Nilai parameter "lang" menggunakan String "en-id" untuk melakukan penerjemahan dari bahasa Inggris ke bahasa Indonesia. Nilai parameter "format" akan menggunakan String "plain" karena bentuk kata yang akan diterjemahkan bersifat kata saja (*plain text*). Hasil terjemahan yang dikirimkan balik oleh Yandex. Translate akan berupa data berformat JSON. Untuk mendapatkan data String hasil terjemahan perlu dilakukan penguraian pada data JSON tersebut. Keluaran akhir pada tahap ini adalah data String dari hasil penerjemahan kata.

Tahap ketiga, objek Mat yang menampilkan gambar hasil penerjemahan yang didapatkan dari tahap sebelumnya dibuat. Objek Mat yang dibuat berbentuk persegi panjang dengan warana latar belakang putih dan warna kata hitam. Gambar 3.9 menunjukkan contoh objek Mat yang dihasilkan pada tahap ini.

Gambar 3.9 Contoh Objek Mat Hasil Terjemahan

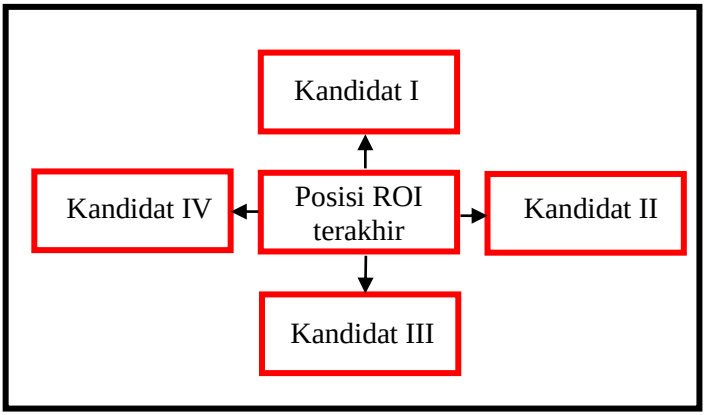
3.2.6.3 Rancangan Proses Penambahan Tampilan Kamera Perangkat dengan Hasil Terjemahan

Pada bagian ini dibuat rancangan tentang bagaimana aplikasi akan menampilkan hasil terjemahan yang didapatkan dari tahap sebelumnya ke dalam tampilan kamera pada perangkat bergerak milik pengguna. Hasil terjemahan akan ditampilkan

dengan cara melakukan penambahan ke dalam tampilan kamera pengguna sehingga membentuk tampilan realitas tertambah. Untuk itu, dibutuhkan informasi posisi gambar ROI pada setiap *frame* yang masuk dari kamera agar dapat aplikasi dapat mengetahui di mana hasil terjemahan harus digambarkan, serta gambar hasil terjemahan itu sendiri. Pelacak yang dibuat merupakan pelacak berbasis histogram HSV. Rancangan pelacak dibuat dengan memperhatikan kompleksitas agar mendukung spesifikasi perangkat uji. Pelacak juga memperhatikan kebutuhan aplikasi, di mana gerakan yang dilakukan saat melakukan penerjemahan relatif sedikit.

Pertama, setelah posisi ROI didapatkan berdasarkan informasi batas atas,bawah, kiri dan kanan dari gambar kata pada ROI tersebut, pelacak akan dinisialisasi. Inisialisasi dilakukan dengan pertama-tama membuat histogram HSV dari gambar teks hasil deteksi aplikasi. Histogram ini akan menjadi fitur gambar teks yang akan digunakan pada saat pelacakan dilakukan. Histogram yang dibuat merupakan histogram tiga dimensi yang memiliki bin sebanyak 8 untuk *channel* Hue, dan masing-masing 3 untuk *channel* Saturation dan Value.

Setelah aplikasi mendapatkan histogram HSV dari gambar kata, pelacak akan dijalankan pada setiap *frame* yang masuk dari kamera perangkat. Masukan dari pelacak ini adalah histogram HSV dari gambar kata dan persegi panjang yang menjadi batas dari gambar kata tersebut. Pertama, *frame* gambar yang masuk dari kamera akan diubah menjadi *frame* HSV. Kemudian pelacakan akan dilakukan dengan menggeser persegi panjang ROI ke arah atas, bawah, kiri, dan kanan. Setiap pergeseran akan dilakukan sejauh 2 piksel dan akan dihentikan setelah berjarak 50 piksel dari posisi asal. Kedua angka tersebut didapat dari hasil eksperimen dengan memperhatikan perpindahan gambar teks saat kamera digerakan serta performa dari perangkat pengguna. Gambar 3.10 menunjukkan posisi ROI terakhir serta daerah pencarian pelacak untuk menemukan posisi ROI selanjtunya



Gambar 3.10 Pergerakan Persegi ROI saat Proses Pelacakan

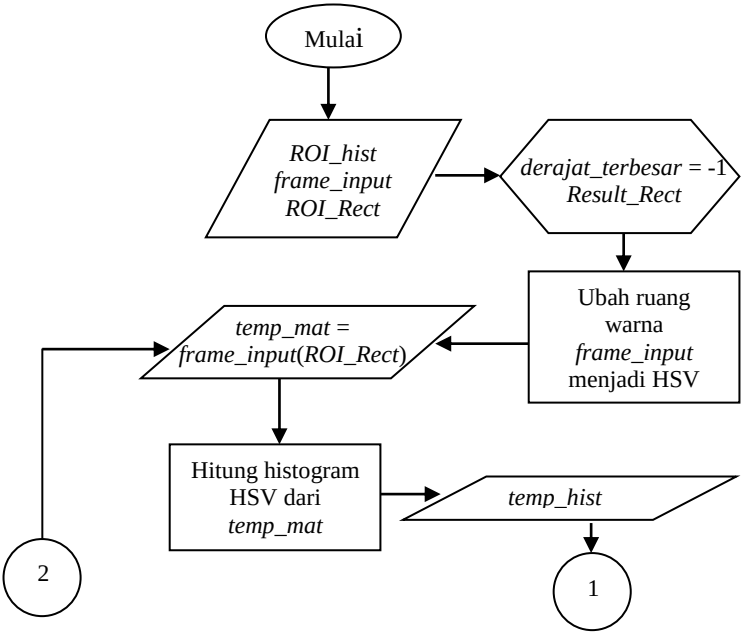
Pada setiap pegeseran, akan dibentuk histogram HSV dari potongan gambar yang terdapat di dalam persegi panjang tersebut. Histogram HSV yang dibuat memiliki jumlah bin yang sama dengan histogram ROI. Histogram tersebut akan di cari derajat kecocokannya dengan histogram ROI menggunakan fungsi Histogram Correlation. Jika hasil dari fungsi ini memiliki nilai yang lebih besar dari penghitungan derajat kecocokan yang didapatkan pada pergeseran sebelumnya, maka aplikasi akan menyimpan nilai derajat kecocokan tersebut dan menyimpan posisi persegi pada saat pergeseran tersebut dilakukan. Selisih kedua nilai tersebut harus lebih dari 0.005. Hal ini dilakukan agar pelacakan lebih stabil dan mengabaikan perubahan nilai kecocokan yang sangat kecil.

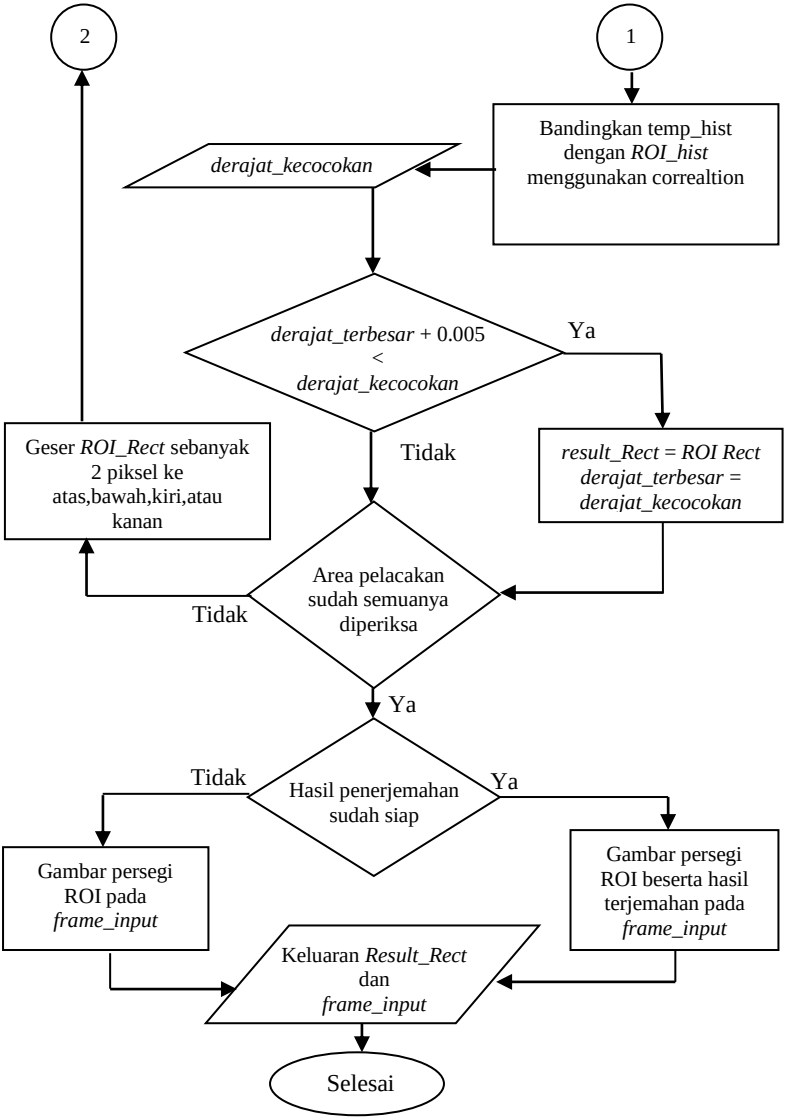
Setelah daerah dengan derajat kecocokan yang terbesar didapatkan, aplikasi akan menggambar persegi pada daerah tersebut pada *frame* masukan dan mengembalikan *frame* tersebut untuk kemudian ditampilkan pada aplikasi.

Jika hasil terjemahan sudah tersedia, hasil terjemahan akan digambar di atas persegi tersebut. Gambar 3.11 menunjukkan tataletak tampilan gambar hasil terjemahan beserta contohnya pada gambar teks. Gambar 3.12 menunjukkan alur

proses pelacakan secara keseluruhan beserta penggambaran hasil penerjemahan.

Gambar 3.11 Tataletak Tampilan Gambar Hasil Terjemahan (a) Beserta Contohnya pada Aplikasi (b)





Gambar 3.12 Diagram Alur Proses Pelacakan Area Gambar Teks beserta Penggambaran Hasil Penerjemahan

(Halaman ini sengaja dikosongkan)

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas tentang implementasi dari aplikasi penerjemah gambar teks berbahasa Inggris menggunakan teknologi realitas tertambah pada perangkat berbasis Android. Di dalamnya mencakup proses penerapan dan pengimplementasian sistem dan antarmuka yang mengacu pada rancangan yang telah dibahas sebelumnya.

4.1 Lingkungan Implementasi

Lingkungan implementasi dari tugas akhir ditunjukkan pada Tabel 4.1 sebagai lingkungan pembangungan aplikasi dan Tabel 4.2 sebagai lingkungan percobaan aplikasi.

Tabel 4.1 Lingkungan Implementasi Perangkat Lunak (1)

Perangkat Keras	Prosesor: - Intel(R) Core(TM) i5-4200U CPU @1.6GHz Memori: - 8.0 GB RAM
Perangkat Lunak	Sistem Operasi : - Microsoft Windows 10 64-bit Perangkat Lunak Pengembang : - Android Studio

Tabel 4.2 Lingkungan Implementasi Perangkat Lunak (2)

Perangkat Keras	Prosesor: - Quad-core 1.2 GHz Cortex-A53 Graphics Processing Unit (GPU): - Adreno 306 450 MHz Memori: - 2048 MB RAM

	Resolusi Layar: 1280 x 720 piksel
Perangkat Lunak	Sistem Operasi : - Android Versi 6.0 (Marshmallow)

4.2 Implementasi Antarmuka

Pada tahap ini akan dijelaskan antarmuka yang diimplementasikan pada aplikasi tugas akhir ini. Antarmuka aplikasi akan berupa tampilan kamera yang akan menampilkan gambar yang didapatkan dari kamera perangkat pengguna. Antarmuka ini juga memungkinkan aplikasi untuk menggunakan kamera untuk mengambil gambar yang akan dijadikan sebagai masukan aplikasi.

Pertama, terlebih dahulu dibuat activity yang akan menyediakan tampilan serta menangani logika yang akan dijalankan pada tampilan utama tersebut. Ketika membuat sebuah proyek Android baru, Android akan menawarkan pengguna untuk membuat sebuah activity baru. Activity yang akan digunakan adalah Blank Activity yang dinamai MainActivity. Setelah proyek dan activity terbuat, akan muncul dua file yang akan digunakan untuk membangun activity tersebut. Kedua file tersebut adalah file activity_main.xml untuk membangun layout activity dan file MainActivity.java untuk membangun proses logika dari activity tersebut.

Untuk melakukan implementasi antarmuka pada aplikasi Android,dilakukan pendefinisian layout pada file activity_main.xml. Antarmuka akan menggunakan ConstraintLayout sebagai layout dasarnya dan menggunakan objek JavaCamera2View yang disediakan oleh pustaka OpenCV4Android untuk menampilkan tampilan kamera pada aplikasi. Kode yang digunakan untuk mendefinisikan antarmuka aplikasi dapat dilihat pada Kode Sumber 4.1.

```
1.  <android.support.constraint.ConstraintLayout xmlns:android=
    "http://schemas.android.com/apk/res/android"
2.      xmlns:app="http://schemas.android.com/apk/res-auto"
3.      xmlns:tools="http://schemas.android.com/tools"
```

```
4.      xmlns:opencv="http://schemas.android.com/apk/res-auto"
5.      android:layout_width="match_parent"
6.      android:layout_height="match_parent"
7.      tools:context=".MainActivity">
8.
9.      <org.opencv.android.JavaCamera2View
10.       android:layout_width="match_parent"
11.       android:layout_height="match_parent"
12.       android:visibility="gone"
13.       android:id="@+id/cvMainCam"
14.       opencv:camera_id="any"
15.       />
16.
17. </android.support.constraint.ConstraintLayout>
```

Kode Sumber 4.1 Definisi Antarmuka Aplikasi

4.3 Implementasi Arsitektur Aplikasi

Pada tahap ini, dilakukan implementasi dari arsitektur aplikasi yang sudah dirancang dan dijelaskan pada Bab III. Implementasi akan dibagi menjadi dua bagian, yaitu pembuatan kelas MainActivity dan pembuatan fungsi-fungsi C/C++/fungsi native pada file native-lib.cpp.

4.3.1 Implementasi Kelas MainActivity

Pada bagian ini, akan dijelaskan implementasi kelas MainActivity yang dilakukan pada file MainActivity.java. Kelas ini akan menginplementasikan kelas CameraBridgeViewBase.CvCameraViewListener2 untuk mengendalikan tampilan kamera serta mendapatkan masukan dari kamera perangkat, dan kelas View.OnTouchListener untuk menangani masukan pengguna berupa sentuhan. Selain itu, kelas MainActivity juga memiliki beberapa variabel privat untuk mendukung jalannya aplikasi. Kode sumber 4.2 menunjukkan pendefinisian kelas MainActivity serta variabel-variabel privat yang akan digunakan beserta penjelasannya.

```
1.  public class MainActivity extends AppCompatActivity implement
    s CameraBridgeViewBase.CvCameraViewListener2, View.OnTouch
    Listener {
2.      private static String TAG = "TA_Apps";
3.      private CameraBridgeViewBase cameraBridgeViewBase;
4.      private Mat input_frame;
5.      private Mat input_capture;
6.      private Mat roi_hsv;
7.      private Mat roi_hist;
8.      private Mat input_track_clone;
9.      private Mat mat_hasil;
10.     private int click_x;
11.     private int click_y;
12.     private int tracking=0;
13.     private int hasil_ready = 0;
14.     private int touch =0;
15.     private double result_val[];
```

Kode Sumber 4.2 Pendefinisian Kelas MainActivity beserta Variabel-Variabel Privat yang Digunakan

Di dalam kelas MainActivity, dilakukan implementasi proses inisialisasi aplikasi, inisialisasi pustaka OpenCV, inisialisasi tampilan kamera, pengambilan *frame* masukan dari kamera perangkat, penanganan masukan pengguna berupa sentuhan pada layar perangkat, pengenalan kata menggunakan Tessract OCR, dan penerjemahan kata.

4.3.1.1 Implementasi Proses Inisialisasi Aplikasi

Pada kelas MainActivity, inisialisasi aplikasi dilakukan dengan cara melaukan Override pada fungsi onCreate. Di dalam fungsi onCreate akan dilakukan inisialisasi layout, pengaturan layar penuh dan layar selalu aktif, dan inisialisasi objek tampilan kamera. Kode sumber 4.3 menunjukkan kode sumber dari fungsi onCreate pada kelas MainActivity.

```
1.  protected void onCreate(Bundle savedInstanceState) {
2.      super.onCreate(savedInstanceState);
3.      setContentView(R.layout.activity_main);
```

```
4.      getWindow().setFlags(WindowManager.LayoutParams.FLAG_FU
        LLSCREEN, WindowManager.LayoutParams.FLAG_FULLSCREEN);
5.      getWindow().setFlags(WindowManager.LayoutParams.FLAG_KE
        EP_SCREEN_ON, WindowManager.LayoutParams.FLAG_KEEP_SCR
        EEN_ON);
6.      cameraBridgeViewBase =findViewById(R.id.cvMainCam);
7.      cameraBridgeViewBase.setVisibility(SurfaceView.VISIBLE);
8.      cameraBridgeViewBase.setCvCameraViewListener(this);
9.      cameraBridgeViewBase.setOnTouchListener(this);
10.  }
```

Kode Sumber 4.3 Fungsi onCreate pada kelas MainActivity

4.3.1.2 Implementasi Proses Inisialisasi Pustaka OpenCV

Pertama-tama, terlebih dahulu dibuat variabel privat baseLoaderCallback yang akan menyimpan sebuah objek BaseLoaderCallback. BaseLoaderCallback berfungsi untuk menerima status inisialisasi OpenCV setelah inisialisasi berhasil atau gagal dilakukan. Di dalam baseLoaderCallback, dilakukan Override pada fungsi onManagerConnected untuk menerima status apakah inisialisasi berhasil atau gagal dilakukan. Apabila inisialisasi berhasil dilakukan, aplikasi akan melakukan pemuatan fungsi-fungsi C/C++ yang ada di dalam file native-lib.cpp dan mengaktifkan tampilan kamera sehingga dapa dilihat pengguna. Kode sumber 4.4 menunjukkan inisialisasi variabel baseLoaderCallback

```
1.  private BaseLoaderCallback baseLoaderCallback = new BaseLoade
    rCallback(this) {
2.      @Override
3.      public void onManagerConnected(int status) {
4.          switch (status) {
5.              case LoaderCallbackInterface.SUCCESS:
6.                  System.loadLibrary("native-lib");
7.                  cameraBridgeViewBase.enableView();
8.                  break;
9.              default:
10.                 super.onManagerConnected(status);
11.                 break;
```

```
        }
    }
};
```

Kode Sumber 4.4 Inisialisasi Variabel Privat baseLoaderCallback

Pada kelas MainActivity, inisialisasi pustaka OpenCV dilakukan dengan cara melakukan Override pada fungsi onResume. Dalam siklus hidup aplikasi Android, onResume akan dijalankan setelah onCreate. Pertama aplikasi akan melakukan pemuatan pustaka OpenCV melalui fungsi OpenCVLoader.initDebug. Pemuatan ini akan mencari pustaka OpenCV di dalam package aplikasi. Apabila pemuatan suskes, maka onResume akan memanggil fungsi baseLoaderCallback.onManagerConnected untuk menyelesaikan proses inisialisasi. Apabila tidak, aplikasi akan melakukan pemuatan pustaka OpenCV melalui aplikasi OpenCVManager pada perangkat pengguna, dan memanggil baseLoaderCallback untuk menyelesaikan inisialisasi. Kode sumber 4.5 menunjukkan fungsi onResume pada kelas MainActivity.

```
@Override
protected void onResume() {
    super.onResume();
    if (!OpenCVLoader.initDebug()) {
        OpenCVLoader.initAsync(OpenCVLoader.OPENCV_VERSION, this, baseLoaderCallback);
    } else {
        baseLoaderCallback.onManagerConnected(LoaderCallbackInterface.SUCCESS);
    }
}
```

Kode Sumber 4.5 Fungsi onResume pada kelas MainActivity

4.3.1.3 Implementasi Proses Pengambilan Frame Masukan dari Kamera Perangkat

Pengambilan *frame* masukan dari kamera perangkat dilakukan dengan melakukan Override pada fungsi onCameraFrame. Fungsi ini dipanggil setiap kali aplikasi menerima *frame* masukan baru dari kamera perangkat. Pada fungsi ini dilakukan pemeriksaan apakah pelacak sedang dalam keadaan dan aktif atau tidak. Jika aktif, akan diperiksa lagi apakah hasil terjemahan sudah siap atau belum. Jika ya, pada saat pemanggilan fungsi trackFrame, alamat dari objek Mat hasil terjemahan akan diberikan. Jika tidak, pada saat pemanggilan fungsi trackFrame, alamat dari objek Mat hasil terjemahan tidak akan diberikan dan hanya akan diberikan nilai -1.

Jika pelacak tidak sedang aktif, akan diperiksa apakah proses pengendalian masukan pengguna berupa sentuhan sedang dilakukan. Apabila ya, aplikasi hanya akan menampilkan *frame* masukan. Jika tidak, aplikasi akan menyimpan *frame* masukan pada variabel input_capture untuk dapat digunakan apabila pengguna melakukan sentuhan. Pemeriksaan proses pengendalian masukan pengguna dilakukan agar pada isi dari variabel input_capture tidak terganggu saat dilakukan pengolahan. Kode sumber 4.6 menunjukkan fungsi onCameraFrame pada kelas MainActivity.

```
1.  @Override
2.  public Mat onCameraFrame(CameraBridgeViewBase.CvCameraVie
    wFrame inputFrame) {
3.     input_frame = inputFrame.rgba();
4.     if(tracking==2){
5.       input_capture=input_frame.clone();
6.       if(hasil_ready==1){
7.         result_val=trackFrame(input_frame.getNativeObjAddr(),roi_hi
    st.getNativeObjAddr(),result_val, mat_hasil.getNativeObjAddr());
8.         return input_frame;
9.       }else{
10.        result_val=trackFrame(input_frame.getNativeObjAddr(), roi_h
    ist.getNativeObjAddr(), result_val,-1);
11.        return input_frame;
```

```
12.         }
13.       }else if(touch==1){
14.         Imgproc.cvtColor(input_frame, input_frame, Imgproc.COLOR_R
      GBA2BGRA);
15.         return input_frame;
16.       }else {
17.         input_capture=input_frame.clone();
18.         Imgproc.cvtColor(input_frame, input_frame, Imgproc.COLOR_R
      GBA2BGRA);
19.         return input_frame;
20.       }
21.   }
```

Kode Sumber 4.6 Fungsi onCameraFrame pada kelas MainActivity

4.3.1.4 Implementasi Proses Pengambilan Frame Masukan dari Kamera Perangkat

Pengambilan *frame* masukan dari kamera perangkat dilakukan dengan melakukan Override pada fungsi onTouch. Fungsi ini dipanggil setiap kali pengguna melakukan sentuhan pada layar, baik saat jari diletakan pada layar, saat jari terletak pada permukaan layar, maupun saat jari meninggalkan permukaan layar. Untuk keperluan aplikasi ini, bentuk sentuhan yang ditangani oleh aplikasi hanya pada saat jari diletakan pada layar perangkat.

Pertama, fungsi onTouch akan mengambil *frame* masukan dari variabel input_capture dan mengubah mode warnanya menjadi grayscale dan BGR secara terpisah. Kemudian, posisi sentuhan pada layar akan disesuaikan menjadi posisi sentuhan pada *frame* masukan. Hal ini perlu dilakukan karena terdapat perbedaan antara luas layar perangkat dengan resolusi gambar *frame* masukan. Frame masukan dalam mode warna grayscale dan BGR serta posisi sentuhan pengguna yang sudah disesuaikan dengan resolusi *frame* masukan akan menjadi masukan bagi fungsi saveFrame yang akan melakukan pengenalan gambar kata. Selain itu, saveFrame juga mengambil

objek Mat yang menjadi tempat di mana ROI disimpan sebagai parameter.

Fungsi saveFrame akan mengembalikan array double yang berisi parameter dari persegi bounding box ROI yang didapatkan setelah melakukan pengenalan gambar kata. Jika pengenalan kata tidak sukses, aplikasi akan memberi tahu pengguna melalui notifikasi pada layar. Jika pengenalan kata sukses, akan dilanjutkan dengan inisialisasi pelacak. Pertama, histogram HSV dari ROI akan dibuat sesuai spesifikasi pada rancangan arsitektur aplikasi. Setelah itu, variabel penanda pelacak akan diubah nilainya untuk menandai bahwa pelacak diaktifkan. Fungsi juga akan memanggil objek SaveROI untuk melakukan penyimpanan gambar ROI dan melakukan pengenalan kata. Objek SaveROI kemudian dieksekusi pada thread aplikasi yang berbeda. Kode sumber 4.7 menunjukkan fungsi onTouch pada kelas MainActivity.

```
1.  @Override
2.  public boolean onTouch(View v, MotionEvent event) {
3.     int action = event.getAction();
4.     if (action == MotionEvent.ACTION_DOWN) {
5.        tracking =0;
6.        hasil_ready=0;
7.        touch =1;
8.        Mat inputs_rgba = input_capture.clone();
9.        Mat inputs = new Mat();
10.       Mat asli_bgr = new Mat();
11.       Imgproc.cvtColor(inputs_rgba, inputs, Imgproc.COLOR_RGBA2
    GRAY);
12.       Imgproc.cvtColor(inputs_rgba, asli_bgr, Imgproc.COLOR_RGBA
    2BGR);
13.       inputs_rgba.release();
14.       click_x = (int) (event.getX() * ((float) inputs.cols() / (float) Res
    ources.getSystem().getDisplayMetrics().widthPixels));
15.       click_y = (int) (event.getY() * ((float) inputs.rows() / (float) R
    esources.getSystem().getDisplayMetrics().heightPixels));
16.       Mat result = new Mat();
17.       Toast.makeText(this, "Tunggu Sebentar...", Toast.LENGTH_SHO
    RT).show();
```

```
18.       result_val = saveFrame(inputs.getNativeObjAddr(), asli_bgr.get
    NativeObjAddr(), result.getNativeObjAddr(), click_x, click_y);
19.       if (result_val.length==1) {
20.          Toast.makeText(this, "Kata Tidak Terdeteksi", Toast.LENGT
    H_SHORT).show();
21.          inputs.release();
22.          result.release();
23.          asli_bgr.release();
24.          touch = 0;
25.          return true;
26.       } else {
27.          roi_hsv = new Mat();
28.          roi_hist = new Mat();
29. Imgproc.cvtColor(result,roi_hsv,Imgproc.COLOR_BGR2HSV);
30.          List<Mat> imageList = new ArrayList<>();
31.          imageList.add(roi_hsv);
32.          Imgproc.calcHist(imageList,new MatOfInt(0,1,2),new Mat(),
    roi_hist,new MatOfInt(8,3,3),new MatOfFloat(0,180,0,256,0,256));

33.          tracking = 2;
34.          inputs.release();
35.          asli_bgr.release();
36.          Imgproc.cvtColor(result,result,Imgproc.COLOR_BGR2GRAY);

37.          SaveROI s = new SaveROI(this);
38.          s.execute(result);
39.       }
40.    }
41.    touch =0 ;
42.    return true;
43. }
```

Kode Sumber 4.7 Fungsi onTouch pada kelas MainActivity

4.3.1.5 Implementasi Pengenalan Kata Menggunakan Tesseract OCR

Implementasi pengenalan kata menggunakan Tesseract OCR dilakukan dengan terlebih dahulu membuat kelas privat baru bernama SaveROI yang merupakan turunan dari kelas AsyncTask. AsyncTask dipilih agar SaveROI dapat dijalankan

pada thread aplikasi yang terpisah dari thread utama. Proses pegenalan kata akan didefinisikan pada fungsi doInBackground.

Pertama akan dilakukan penyimpanan ROI pada perangkat pengguna. ROI akan disimpan dalam file berformat .png. Setelah gambar berhasil disimpan, Tesseract OCR akan diinisialisasi. Inisialisasi dilakukan dengan membuat objek TessBaseAPI baru. Kemudian, akan dipanggil fungsi init pada objek tersebut untuk melakukan inisialisasi. Fungsi init menerima lokasi folder tessdata yang berisi eng.trainnedata pada aplikasi serta kode bahasa dari gambar teks yang ingin dikenali. Setelah itu, dipanggil fungsi setImage untuk memilih ROI sebagai gambar masukan Tesseract OCR, dan diikuti pemanggilan fungsi getUTF8Text untuk mendapatkan hasil pengenalan gambar kata berupa data String. Apabila Tesseract OCR berjalan dengan sukses, SaveROI akan memanggil objek TranslateProcess pada fungsi onPostExecute dan mengeksekusi objek tersebut pada thread terpisah. Fungsi TranslateProcess digunakan untuk mendapatkan hasil penerjemahan kata ke dalam bahasa Indonesia. Kode sumber 4.8 menunjukkan hasil implementasi kelas SaveROI.

```
1.  private class SaveROI extends AsyncTask<Mat, Integer, Integer> {

2.      private Context mContext;
3.      private String ocrResult;
4.      public SaveROI(Context context) {
5.          mContext = context;
6.      }
7.      protected void onProgressUpdate(Integer... progress) {
8.      }
9.      protected void onPostExecute(Integer result) {
10.         if(result == 1){
11.             Toast.makeText(mContext, "Pengenalan Kata Gagal!", Toast.L
    ENGTH_SHORT).show();
12.         }else if(result==2){
13.             TranslateProcess tp =  new TranslateProcess(mContext);
14.             tp.execute(ocrResult);
15.         }
16.     }
17.     @Override
```

```
18.    protected Integer doInBackground(Mat... mats) {
19.       Mat result = mats[0];
20.       int status = 2;
21.       Bitmap bmp = null;
22.       try {
23.          bmp = Bitmap.createBitmap(result.cols(), result.rows(), Bitm
    ap.Config.ARGB_8888);
24.          Utils.matToBitmap(result, bmp);
25.       } catch (CvException e) {
26.          Log.d(TAG, e.getMessage());
27.          status = 1;
28.       }
29.       FileOutputStream out = null;
30.       String filename = "roi.png";
31.       File sd = new File(Environment.getExternalStorageDirectory()
    + "/frames");
32.       Log.d(TAG, Environment.getExternalStorageDirectory() + "/fra
    mes");
33.       boolean success = true;
34.       if (!sd.exists()) {
35.          success = sd.mkdir();
36.       }
37.       if (success) {
38.          File dest = new File(sd, filename);
39.          try {
40.             out = new FileOutputStream(dest);
41.          bmp.compress(Bitmap.CompressFormat.PNG, 100, out);
42.          } catch (Exception e) {
43.             e.printStackTrace();
44.             Log.d(TAG, e.getMessage());
45.             return -1;
46.          } finally {
47.             try {
48.                if (out != null) {
49.                   out.close();
50.                   status = 1;
51.                   result.release();
52.                   try {
53.                      TessBaseAPI baseAPI = new TessBaseAPI();

54.                      baseAPI.init(Environment.getExternalStorageDirect
    ory().toString(), "eng");
55.                      baseAPI.setImage(dest);
```

```
56.                     ocrResult = baseAPI.getUTF8Text();
57.                     baseAPI.end();
58.                     status = 2;
59.                   } catch (Exception e) {
60.                     e.printStackTrace();
61.                     Log.d(TAG, e.getMessage());
62.                     status = 1;
63.                   }
64.                 }
65.               } catch (IOException e) {
66.                 e.printStackTrace();
67.               }
68.             }
69.           }
70.           return status;
71.       }
72.   }
```

Kode Sumber 4.8 Implementasi Kelas SaveROI

4.3.1.6 Implementasi Penerjemahan Kata

Implementasi penerjemahan kata dilakukan dengan terlebih dahulu membuat kelas privat baru bernama TranslateProcess yang merupakan turunan dari kelas AsyncTask

Pertama akan terlebih dulu dibuat String berupa URL HTTP untuk melakukan request ke alamat https://translate.yandex.net/api/v1.5/tr.json/translate. URL yang dibangun akan menyertakan beberapa variabel yang kemudian disusun menjadi sebuah URL permintaan HTTP GET. Dalam URL tersebut, akan disertakan variabel “key”, “lang”, “format”, dan “text”. Variabel “key” yang merupakan kunci API Yandex. Translate didapatkan dari situs web https://tech.yandex.com/translate/. Variabel “lang” akan berisi String “en-id” sebagai kode penerjemahan dari Bahasa Inggris ke bahasa Indonesia. Variabel format akan berisi String “plain”, karena teks yang dikirim ke Yandex. Translate akan berupa data plain text. Terakhir, variabel “text” akan berisi data String yang didapatkan dari hasil pengenalan kata.

Setelah request tersebut dikirim menggunakan URLConnection, didapatkan hasil terjemahan berupa data JSON. Dari data JSON tersebut, akan di-parse nilai status request dan kata yang sudah diterjemahkan ke bahasa Indonesia. Jika status bernilai "200", maka TranslateProcess akan membuat objek Mat kosong untuk menaruh gambar hasil terjemahan. Objek Mat tersebut akan dikirim ke fungsi C/C++ getTextMat untuk digambari dengan gambar hasil terjemahan. Kode sumber 4.9 menunjukkan hasil implementasi kelas TranslateProcess.

```
1.  public class TranslateProcess extends AsyncTask {
2.    private Context context;
3.    private int status;
4.    private String hasil;
5.    public TranslateProcess(Context context){
6.      this.context = context;
7.    }
8.    @Override
9.    protected Object doInBackground(Object[] objects) {
10.     try{
11.       String kata = (String)objects[0];
12.       String link = "https://translate.yandex.net/api/v1.5/tr.json/
translate";
13.       String data = URLEncoder.encode("key","UTF-
8")+"="+URLEncoder.encode("trnsl.1.1.20180312T052742Z.0352
502647a923c3.5c5bb1aa56ca8f77f9a663af8962d80bd4c624c5","
UTF-8");
14.       data += "&"+URLEncoder.encode("lang","UTF-
8")+"="+URLEncoder.encode("en-id","UTF-8");
15.       data += "&"+URLEncoder.encode("format","UTF-
8")+"="+URLEncoder.encode("plain","UTF-8");
16.       data += "&"+URLEncoder.encode("text","UTF-
8")+"="+URLEncoder.encode(kata,"UTF-8");
17.       URL url = new URL(link);
18.       URLConnection conn = url.openConnection();
19.       conn.setDoOutput(true);
20.       OutputStreamWriter wr = new OutputStreamWriter(conn.ge
tOutputStream());
21.       wr.write(data);
22.       wr.flush();
```

```
23.          BufferedReader reader = new BufferedReader(new InputStr
     eamReader(conn.getInputStream()));
24.          StringBuilder sb = new StringBuilder();
25.          String line = null;
26.          while((line=reader.readLine())!=null){
27.            sb.append(line);
28.            break;
29.          }
30.          JSONObject jsonOBJ = new JSONObject(sb.toString());
31.          JSONArray ja = jsonOBJ.getJSONArray("text");
32.          hasil =  ja.getString(0);
33.          return jsonOBJ.getInt("code");
34.        }catch (Exception e){
35.          return new String("Exception: "+e.getMessage());
36.        }
37.      }
38.     @Override
39.      protected void onPostExecute(Object o) {
40.        status= (int)o;
41.        if(status==200) {
42.          mat_hasil = new Mat(100,300, CvType.CV_8UC3);
43.        getTextMat(hasil,mat_hasil.getNativeObjAddr());
44.          hasil_ready = 1;
45.        }else{
46.          Toast.makeText(context,"Proses Penerjemahan Bermasalah!
     ",Toast.LENGTH_LONG).show();
47.        }
48.      }
49.  }
```

Kode Sumber 4.9 Implementasi Kelas TranslateProcess

4.3.2 Implementasi Fungsi-Fungsi C/C++

Pada bagian ini, akan dijelaskan implementasi dari fungsi-fungsi yang akan ditulis dan dijalankan dalam bahasa C/C++. Fungsi-fungsi tersebut akan didefinisikan di dalam file native-lib.cpp. Fungsi-fungsi yang didefinisikan adalah fungsi untuk melakukan pengambilan gambar kata berdasarkan masukan pengguna, pelacakan gambar kata pada *frame* masukan dari kamera perangkat, dan pembuatan objek Mat hasil terjemahan.

4.3.2.1 Implementasi Pengambilan Gambar Kata Berdasarkan Masukan Pengguna

Implementasi pengambilan gambar kata akan pada fungsi getWord. Fungsi getWord menerima *frame* masukan dari kamera dalam mode warna RGBA dan grayscale, posisi sentuhan pengguna, serta objek Mat untuk menaruh gambar kata yang didapatkan.

Pertama, diberikan filter Sobel Gx dan Gy pada *frame* masukan grayscale yang ditaruh dalam objek Mat terpisah untuk mendapatkan tepian vertikal dan horizontal. Setelah itu, dilakukan thresholding dan morfologi pada masing-masing objek Mat Sobel. Kemudian, dilakukan pencarian perkiraan tinggi kata. Pencarian dilakukan dengan melakukan iterasi pada objek Mat Sobel Gx seusuai dengan rancangan arsitektur aplikasi. Setelah perkiraan tinggi kata didapatkan, dilakukan penghitungan jarak antar huruf pada objek Mat Sobel Gy. Setelah jarak antar huruf didapatkan, dilakukan perhitungan lebar dari gambar kata yang dipilih, kemudian diikuti dengan penghitungan ulang tinggi gambar kata untuk mendapatkan tinggi kata yang sebenarnya. Proses-proses tersebut akan menghasilkan keluaran berupa batas atas,bawah,kiri dan kanan dari gambar kata yang dipilih.

Setelah batas-batas gambar kata didapatkan, Gambar kata akan diambil dari *frame* masukan pada posisi sesuai dengan batas-batas tersebut. Kode sumber 4.10 menunjukkan implementasi dari fungsi getWord.

```
1.  JNIEXPORT jdoubleArray JNICALL Java_com_antonius_ta_
    MainActivity_getWord(JNIEnv *env,jobject,jlong mat,j
    long asli,jlong output,jint x,jint y) {
2.      jdoubleArray roi_result;
3.      jdoubleArray if_error;
4.      roi_result = (env)->NewDoubleArray(4);
5.      if_error = (env)->NewDoubleArray(1);
6.      jdouble temp2[1];
7.      temp2[0]=1;
8.      (env)-
    >SetDoubleArrayRegion(if_error,0,1,temp2);
```

```
9.      Mat &image_g = *(Mat *) mat;
10.     Mat &image_output = *(Mat *) output;
11.     Mat &image_a = *(Mat *) asli;
12.     Mat image_asli = image_a.clone();
13.     Mat image_gray = image_g.clone();
14.     Mat sobelx64f;
15.     Mat sobely64f;
16.     Sobel(image_gray, sobelx64f, CV_64F, 1, 0, 3);
17.     Sobel(image_gray, sobely64f, CV_64F, 0, 1, 3);
18.     Mat abs_sobelx64f, ix;
19.     abs_sobelx64f = abs(sobelx64f);
20.     abs_sobelx64f.convertTo(ix, CV_8U);
21.     abs_sobelx64f.release();
22.     sobelx64f.release();
23.     Mat iy,abs_sobely64f;
24.     abs_sobely64f = abs(sobely64f);
25.     abs_sobely64f.convertTo(iy, CV_8U);
26.     abs_sobely64f.release();
27.     sobely64f.release();
28.     int line_left = x - 100;
29.     int line_right = x + 100;
30.     int flag = 0;
31.     int tr_x = 15;
32.     int tr_y = 15;
33.     for (int i = 0; i < ix.rows; i++) {
34.         for (int j = 0; j < ix.cols; j++) {
35.             if (ix.at<uchar>(i, j) < tr_x) {
36.                 ix.at<uchar>(i, j) = 0;
37.             } else {
38.                 ix.at<uchar>(i, j) = 255;
39.             }
40.         }
41.     }
42.     for (int i = 0; i < iy.rows; i++) {
43.         for (int j = 0; j < iy.cols; j++) {
44.             if (iy.at<uchar>(i, j) < tr_y) {
45.                 iy.at<uchar>(i, j) = 0;
46.             } else {
47.                 iy.at<uchar>(i, j) = 255;
48.             }
49.         }
50.     }
51.     morphologyEx(iy, iy, MORPH_OPEN, Mat());
```

```
52.     morphologyEx(ix, ix, MORPH_OPEN, Mat());
53.     int i,j;
54.     int batas_atas, batas_bawah;
55.     for (i = y - 1;; i -= 1) {
56.         for (j = line_left; j <= line_right; j++) {

57.             if (ix.at<uchar>(i, j) == 255) {
58.                 flag = 1;
59.                 break;
60.             }
61.         }
62.         if (flag == 0) {
63.             batas_atas = i;
64.             break;
65.         }
66.         flag = 0;
67.         if(i-1<0){
68.             return if_error;
69.         }
70.     }
71.     for (i = y + 1;; i += 1) {
72.         for (j = line_left; j <= line_right; j++) {

73.             if (ix.at<uchar>(i, j) == 255) {
74.                 flag = 1;
75.                 break;
76.             }
77.         }
78.         if (flag == 0) {
79.             batas_bawah = i;
80.             break;
81.         }
82.         flag = 0;
83.         if(i+1>=ix.rows){
84.             return if_error;
85.         }
86.     }
87.     flag = 0;
88.     int batas_kiri, batas_kanan;
89.     int alpha_space = 1;
90.     int clickpos_to_space_alpha_space = 0;
91.     for (i = x - 1;; i -= 1) {
92.         if (flag == 0) {
```

```
93.             clickpos_to_space_alpha_space += 1;
94.             for (j = batas_atas; j <= batas_bawah; j
    ++) {
95.                 if (iy.at<uchar>(j, i) == 255) {
96.                     flag = 1;
97.                     break;
98.                 }
99.             }
100.        }
101.        if (flag == 0 || flag == 2) {
102.            flag = 2;
103.            alpha_space += 1;
104.            for (j = batas_atas; j <= batas_bawah;
    j++) {
105.                if (iy.at<uchar>(j, i) == 255) {
106.                    flag = 3;
107.                    break;
108.                }
109.            }
110.        }
111.        if (flag == 3) {
112.            break;
113.        }
114.        if (flag == 1) {
115.            flag = 0;
116.        }
117.    }
118.    int alpha_space_tolerance = alpha_space;
119.    flag = 0;
120.    for (i = x - 1;; i -= 1) {
121.        for (j = batas_atas; j <= batas_bawah; j++)
     {
122.            if (iy.at<uchar>(j, i) == 255) {
123.                flag = 1;
124.                break;
125.            }
126.        }
127.        if (flag == 0) {
128.            i -= alpha_space;
129.            for(int z=i;z>=i-
    alpha_space_tolerance;z-=1){
130.                for (j = batas_atas; j <= batas_baw
    ah; j++) {
```

```
131.                     if (iy.at<uchar>(j, z) == 255)
    {
132.                         flag = 4;
133.                         break;
134.                     }
135.                 }
136.                 if(flag==4){
137.                     i=z;
138.                     break;
139.                 }
140.             }
141.             if(flag!=4){
142.                 flag = 3;
143.             }
144.         }
145.         if (flag == 3) {
146.             batas_kiri = i;
147.             break;
148.         }
149.         if (flag != 3) {
150.             flag = 0;
151.         }
152.         if(i-1<0){
153.             return if_error;
154.         }
155.     }
156.     for (i = x + 1;; i += 1) {
157.         for (j = batas_atas; j <= batas_bawah; j++)
     {
158.             if (iy.at<uchar>(j, i) == 255) {
159.                 flag = 1;
160.                 break;
161.             }
162.         }
163.         if (flag == 0) {
164.             i += alpha_space;
165.             for(int z=i;z<=i+alpha_space_tolerance;
    z+=1){
166.                 for (j = batas_atas; j <= batas_baw
    ah; j++) {
167.                     if (iy.at<uchar>(j, z) == 255)
    {
168.                         flag = 4;
```

```
169.                         break;
170.                     }
171.                 }
172.                 if(flag==4){
173.                     i=z;
174.                     break;
175.                 }
176.             }
177.             if(flag!=4){
178.                 flag = 3;
179.             }
180.         }
181.         if (flag == 3) {
182.             batas_kanan = i;
183.             break;
184.         }
185.         if (flag != 3) {
186.             flag = 0;
187.         }
188.         if(i+1>=iy.cols){
189.             return if_error;
190.         }
191.     }
192.     for (i = y - 1;; i -= 1) {
193.         for (j = batas_kiri; j <= batas_kanan; j++)
    {
194.             if (ix.at<uchar>(i, j) == 255) {
195.                 flag = 1;
196.                 break;
197.             }
198.         }
199.         if (flag == 0) {
200.             batas_atas = i;
201.             break;
202.         }
203.         flag = 0;
204.         if(i-1<0){
205.             return if_error;
206.         }
207.     }
208.     for (i = y + 1;; i += 1) {
209.         for ( j = batas_kiri; j <= batas_kanan; j++
   ) {
```

```
210.             if (ix.at<uchar>(i, j) == 255) {
211.                 flag = 1;
212.                 break;
213.             }
214.         }
215.         if (flag == 0) {
216.             batas_bawah = i;
217.             break;
218.         }
219.         flag = 0;
220.         if(i+1>=ix.rows){
221.             return if_error;
222.         }
223.     }
224.     cv::Rect2d roi;
225.     roi.x = batas_kiri-10;
226.     roi.y = batas_atas-10;
227.     roi.width = (batas_kanan+10) - (batas_kiri-
    10);
228.     roi.height = (batas_bawah+10) - (batas_atas-
    10);
229.     jdouble temp1[4];
230.     temp1[0]=roi.x;
231.     temp1[1]=roi.y;
232.     temp1[2]=roi.width;
233.     temp1[3]=roi.height;
234.     (env)-
    >SetDoubleArrayRegion(roi_result,0,4,temp1);
235.     image_output= image_asli(roi);
236.     image_gray.release();
237.     image_asli.release();
238.     ix.release();
239.     iy.release();
240.     return roi_result;
241. }
```

Kode Sumber 4.10 Implementasi Fungsi getWord

4.3.2.2 Implementasi Pelacakan Gambar Kata Pada Frame Masukan dari Kamera Perangkat

Implementasi pelcakan gambar kata pada *frame* masukan dari kamera perangkat dilakukan pada fungsi trackFrame. Fungsi trackFrame menerima masukan berupa *frame* masukan dari kamera perangat, histogram HSV dari gambar kata, array tipe data double yang berisi parameter-parameter dari persegi bounding box gambar kata, posisi sentuhan pengguna, serta objek Mat dari gambar hasil terjemahan.

Pertama, *frame* masukan dari kamera akan diubah mode warnanya menjadi HSV. Persegi bounding box berdasarkan array tipe data double pada parameter fungsi juga dibuat. Pelacakan dilakukan dengan mengambil gambar di posisi tempat bounding bix berada pada *frame* masukan HSV. Dari potongan gambar tersebut dibuat histogramnya. Histogram tersebut dibandingkan dengan histogram HSV gambar text menggunakan fungsi compHist. Mode perbandingan yang digunakan adalah CV_COMP_CORREL.

Setelah pelacakan dilakuakan dan perkiraan posisi gambar kata pada *frame* masukan didapatkan, persegi bounding box hasil pelacakan akan digambar pada *frame* masukan. Jika hasil terjemahan sudah siap, hasil terjemahan juga akan digambar pada *frame* masukan. Frame masukan yang sudah digamabri tersebut akan ditampilkan pada layar perangkat. Kode sumber 4.11 menunjukkan hasil implementasi dari fungsi trackFrame.

```
1.  JNIEXPORT jdoubleArray JNICALL Java_com_antonius_ta_
    MainActivity_trackFrame2(JNIEnv *env,jobject,jlong i
    nput_frame,jlong roi_hist,jdoubleArray roi_box_param
    ,jlong mat_hasil,jint moving) {
2.      jdoubleArray bound_result;
3.      Mat &roi_h = *(Mat *) roi_hist;
4.      Mat &input_rgba = *(Mat *) input_frame;
5.      bound_result = (env)->NewDoubleArray(4);
6.      Mat input_bgr, input_hsv, mask;
7.      cvtColor(input_rgba, input_bgr, COLOR_RGBA2BGR);
```

```
8.      cvtColor(input_bgr, input_hsv, COLOR_BGR2HSV);
9.      double *roi_box_p;
10.     roi_box_p = (env)-
    >GetDoubleArrayElements(roi_box_param, NULL);
11.     Rect roi_box;
12.     roi_box.x = (int) roi_box_p[0];
13.     roi_box.y = (int) roi_box_p[1];
14.     roi_box.width = (int) roi_box_p[2];
15.     roi_box.height = (int) roi_box_p[3];
16.     float hranges[] = {0, 180};
17.     float sranges[] = {0, 256};
18.     const float *phranges[] = {hranges, sranges, sra
    nges};
19.     int ch[] = {0, 1, 2};
20.     double result = 0;
21.     Rect result_box = roi_box;
22.     int bins[] = {8, 3, 3};
23.     int dims = 3;
24.     Mat temp_hsv, imageList[1], temp_hist;
25.     temp_hsv = input_hsv(roi_box).clone();
26.     imageList[0] = temp_hsv;
27.     int frame_range = 50;
28.     int frame_increment = 2;
29.     calcHist(imageList, 1, ch, mask, temp_hist, dims
    , bins, phranges);
30.     double comps = compareHist(roi_h, temp_hist, CV_
    COMP_CORREL);
31.     result = comps;
32.     result_box = roi_box;
33.     double pass = 0.005;
34.     Rect temp;
35.     for (int i = roi_box.x+frame_increment; i <= roi
    _box.x + frame_range; i += frame_increment) {
36.         if ((i + roi_box.width) > input_hsv.cols) {

37.             break;
38.         }
39.         temp.x = i;
40.         temp.y = (int) roi_box_p[1];
41.         temp.width = roi_box.width;
42.         temp.height = roi_box.height;
43.         temp_hsv = input_hsv(temp).clone();
44.         imageList[0] = temp_hsv;
```

```
45.         calcHist(imageList, 1, ch, mask, temp_hist,
    dims, bins, phranges);
46.         comps = compareHist(roi_h, temp_hist, CV_COM
    P_CORREL);
47.         if (comps > result + pass) {
48.             result = comps;
49.             result_box = temp;
50.         }
51.     }
52.     for (int i = roi_box.x-
    frame_increment; i >= roi_box.x - frame_range; i -
    = frame_increment) {
53.         if (i < 0) {
54.             break;
55.         }
56.
57.         temp.x = i;
58.         temp.y = (int) roi_box_p[1];
59.         temp.width = roi_box.width;
60.         temp.height = roi_box.height;
61.         temp_hsv = input_hsv(temp).clone();
62.         imageList[0] = temp_hsv;
63.         calcHist(imageList, 1, ch, mask, temp_hist,
    dims, bins, phranges);
64.         comps = compareHist(roi_h, temp_hist, CV_COM
    P_CORREL);
65.         if (comps > result + pass) {
66.             result = comps;
67.             result_box = temp;
68.         }
69.     }
70.     for (int i = roi_box.y-
    frame_increment; i >= roi_box.y - frame_range; i -
    = frame_increment) {
71.         if (i < 0) {
72.             break;
73.         }
74.         temp.x = (int) roi_box_p[0];
75.         temp.y = i;
76.         temp.width = roi_box.width;
77.         temp.height = roi_box.height;
78.         temp_hsv = input_hsv(temp).clone();
79.         imageList[0] = temp_hsv;
```

```
80.         calcHist(imageList, 1, ch, mask, temp_hist,
    dims, bins, phranges);
81.         comps = compareHist(roi_h, temp_hist, CV_COM
    P_CORREL);
82.         if (comps > result + pass) {
83.             result = comps;
84.             result_box = temp;
85.         }
86.     }
87.     for (int i = roi_box.y+frame_increment; i <= roi
    _box.y + frame_range; i += frame_increment) {
88.         if (i + roi_box.height > input_hsv.rows) {
89.             break;
90.         }
91.         temp.x = (int) roi_box_p[0];
92.         temp.y = i;
93.         temp.width = roi_box.width;
94.         temp.height = roi_box.height;
95.         temp_hsv = input_hsv(temp).clone();
96.         imageList[0] = temp_hsv;
97.         calcHist(imageList, 1, ch, mask, temp_hist,
    dims, bins, phranges);
98.         comps = compareHist(roi_h, temp_hist, CV_COM
    P_CORREL);
99.         if (comps > result + pass) {
100.             result = comps;
101.             result_box = temp;
102.         }
103.     }
104.     temp_hsv.release();
105.     input_hsv.release();
106.     temp_hist.release();
107.     imageList->release();
108.     mask.release();
109.     rectangle(input_bgr, result_box, Scalar(255, 0,
     0), 3);
110.     if (mat_hasil != -1) {
111.         Mat &result_mat = *(Mat *) mat_hasil;
112.         if(result_box.y-
    100>=0&&result_box.x+300<input_bgr.cols) {
113.             result_mat.copyTo(
114.                     input_bgr.rowRange(result_box.y
     - 100, result_box.y).colRange(result_box.x,
```

```
115.
                                    result_box.x +
116.
                                    300));
117.         }
118.     }
119.     cvtColor(input_bgr, input_rgba, COLOR_BGR2BGRA)
    ;
120.     input_bgr.release();
121.     imageList->release();
122.     jdouble temp1[4];
123.     temp1[0] = result_box.x;
124.     temp1[1] = result_box.y;
125.     temp1[2] = result_box.width;
126.     temp1[3] = result_box.height;
127.     (env)-
    >SetDoubleArrayRegion(bound_result, 0, 4, temp1);
128.     return bound_result;
129. }
```

Kode Sumber 4.11 Implementasi Fungsi trackFrame

4.3.2.3 Implementasi Pembuatan Gambar Hasil Terjemahan

Implementasi pembuatan gambar hasil terjemahan dilakukan pada fungsi getTextMat. Fungsi getTextMat menerima masukan berupa String hasil terjemahan dan objek Mat tempat gambar hasil terjemahan.

Data String dari bahasa pemrograman Java harus dikonversi terlebih dahulu menjadi data string C/C++ agar dapat digunakan pada fungsi getTextSize dan putText milik OpenCV. Konversi dilakukan dengan cara mengubah String Java menjadi string UTF8 menggunakan fungsi GetStringUTFChars dan GetStringUTFLength. Setelah itu String UTF8 dibangun dari hasil konversi. Kode sumber 4.12 menunjukkan hasil implementasi fungsi konversi String Java ke String UTF8.

```
1.  string convertJavaStringToUTFString(JNIEnv* env, jstring java_stri
    ng) {
2.    const char* javaStringChars = env-
    >GetStringUTFChars(java_string, (jboolean *)0);
```

```
3.     const jsize javaStringLength = env-
    >GetStringUTFLength(java_string);
4.     string UTFString(javaStringChars, javaStringLength);
5.     env->ReleaseStringUTFChars(java_string, javaStringChars);
6.     return UTFString;
7.  }
```

Kode Sumber 4.12 Implementasi Fungsi Konversi String Java ke String UTF8

Implementasi fungs getTextMat dimulai dengan mengkonversi String hasil terjemahan menggunakan fungsi convertJavaStringToUTFString yang telah dibuat sebelumnya. Setelah itu, ukuran kata yang akan digambar dicari menggunakan fungsi getTextSize. Keluaran dari fungsi getTextSize digunakan untuk membuat objek Mat sementara untuk menaruh gambar kata tersebut. Gambar kata digambar menggunakan fungsi putText. Setelah objek Mat sementara berisi gambar teks, objek Mat tersebut akan di pindahkan ke objek Mat tempat gambar hasil terjemahan untuk kemudian digambar pada layar. Kode sumber 4.13 menunjukkan hasil implementasi fungsi getTextMat.

```
1.  JNIEXPORT void JNICALL Java_com_antonius_ta_MainActivity_getT
    extMat(JNIEnv *env,jobject,jstring string_hasil,jlong result_mat) {
2.     Mat &rm = *(Mat *) result_mat;
3.     string hasil_string = convertJavaStringToUTFString(env, string_h
    asil);
4.     int baseline = 0;
5.     Size textSize = getTextSize(hasil_string, FONT_HERSHEY_DUPLE
    X,2, 2, &baseline);
6.     baseline += 2;
7.     Mat textImg(textSize.height + baseline ,textSize.width,rm.type());
8.     textImg.setTo(Scalar(255,255,255));
9.     Point textOrg((textImg.cols -
     textSize.width) / 2,(textImg.rows + textSize.height) / 2);
10.    putText(textImg, hasil_string, textOrg, FONT_HERSHEY_DUPLEX,
    2, Scalar(0,0,0), 2, LINE_AA);
11.    resize(textImg,rm,rm.size(),0,0);
12. }
```

Kode Sumber 4.13 Implementasi Fungsi getTextMat

BAB V
PENGUJIAN DAN EVALUASI

Bab ini berisi bahasan mengenai uji coba dan evaluasi dari hasil implementasi aplikasi penerjemah gambar teks berbahasa Inggris menggunakan teknologi realitas tertambah pada perangkat berbasis Android. Uji coba yang dilakukan adalah uji coba kotak hitam dan uji coba usabilitas pada sejumlah pengguna uji.

5.1 Lingkungan Uji Coba

Proses uji coba dilakukan pada ponsel pintar (*smartphone)* Asus Zenfone 2 Laser. Pada uji coba ini, lingkungan dibedakan menjadi lingkungan perangkat keras dan lingkungan perangkat lunak. Penjelasan spesifikasi perangkat keras dan perangkat lunak pada lingkungan uji coba dapat dilihat pada Tabel 5.1.

Tabel 5.1 Lingkungan Perangkat Uji Coba

Perangkat Keras	Prosesor: - Quad-core 1.2 GHz Cortex-A53 Graphics Processing Unit (GPU): - Adreno 306 450 MHz Memori: - 2048 MB RAM Resolusi Layar: 1280 x 720 piksel
Perangkat Lunak	Sistem Operasi : - Android Versi 6.0 (Marshmallow)

5.2 Pengujian Kotak Hitam

Pengujian kotak hitam adalah pengujian yang dilakukan untuk Pengujian kotak hitam adalah pengujian yang dilakukan untuk memeriksa apakah setiap komponen aplikasi telah berjalan

dengan baik atau tidak tanpa memperhatikan kode sumber dari aplikasi tersebut. Pengujian dilakukan dengan memberikan masukan dengan kondisi tertentu ke dalam setiap komponen aplikasi. Keluaran dari setiap komponen aplikasi akan diperiksa untuk kemudian dieveluasi apakah komponen tersebut berjalan dengan baik atau tidak. Komponen aplikasi yang akan diujikan adalah pengambilan ROI, pengenalan kata dengan OCR, penerjemahan kata, penggambaran hasil terjemahan, dan pelacakan ROI.

5.2.1 Data Uji yang Digunakan

Data uji yang digunakan bertujuan untuk menguji kemampuan setiap komponen aplikasi. Data uji akan terbagi menjadi dua jenis, yaitu data uji buatan dan data uji objek nyata. Pembagian data uji dilakukan untuk menguji kemampuan aplikasi dalam menangani keadaan uji dan keadaan dunia nyata sehari-hari.

Data uji buatan akan dicetak pada kertas HVS ukuran A4. Faktor-faktor pengujian yang dipilih digunakan dalam data uji adalah jumlah kata, bentuk karakter, dan warna. Untuk jumlah kata, data uji buatan dibagi menjadi dua, yaitu satuan dan banyak. Untuk kata satuan, kata yang digunakan adalah “Copyright”. Kata ini dipilih karena terdiri dari karakter yang semuanya berbeda. Hal ini dilakukan untuk menguji kemampuan pengambilan ROI pada bentuk huruf yang berbeda-beda. Untuk kata banyak, digunakan kata-kata “One”, “Two”, “Three”, “Four”, “Five”, “Six”, “Seven”, “Eight”, dan “Nine”. Kata-kata ini dipilih untuk mencoba karakter-karakter yang tidak terdapat pada kata “Copyrights” serta panjang kata-kata tersebut yang pendek memungkinkan kata-kata tersebut untuk disusun. Data uji kata banyak disusun menjadi tiga kelompok, yaitu “One Two Three”, “Four Five Six”, dan “Seven Eight Nine”. Kelompok-kelompok tersebut dilatakan berurutan dari atas ke bawah.

Pada data uji kata satuan dan kata banyak, diberikan variasi warna pada huruf dan latar belakangnya. Variasi warna diberikan

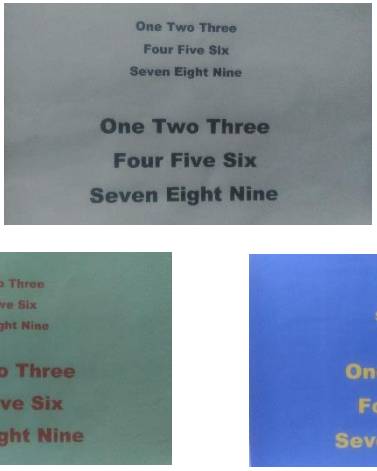
untuk menguji kemampuan pelacak yang berbasis histogram HSV. Warna yang digunakan adalah hitam di atas latar belakang putih, merah di atas latar belakang putih, dan warna kuning di atas latar belakang biru. Warna-warna tersebut dipilih untuk mengetahui kemampuan pelacak dalam menangani warna grayscale maupun warna spektrum biasa.

Untuk data uji kata satuan, diberikan variasi jenis *font*. Jenis *font* yang digunakan adalah New Courier dan Consolas untuk jenis *font* dengan luas wilayah karakter yang sama (fixed-pitch) serta Arial Black dan Times New Roman untuk jenis *font* dengan luas wilayah karakter yang tidak sama (non fixed-pitch). Pembedaan tersebut dilakukan untuk menguji kemampuan pengambilan ROI jika dihadapkan dengan jarak antar huruf yang sama dan yang berbeda-beda. Arial Black dan Times New Roman dipilih juga untuk membedakan *font* huruf dengan ketebalan yang berbeda. Pada data uji kata banyak, jenis *font* yang digunakan hanya Arial Black.

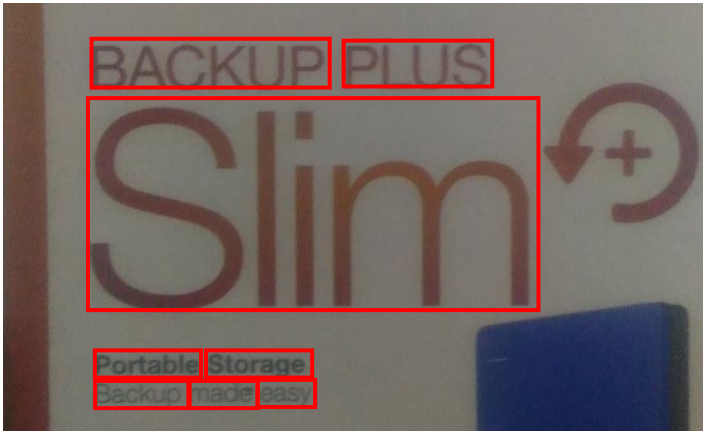
Untuk data uji kata satuan dan kata banyak, diberikan variasi ukuran huruf. Ukuran huruf yang digunakan adalah 28 point dan 48 point. Variasi ini dilakukan untuk menguji pengaruh ukuran huruf terhadap komponen aplikasi. Gambar 5.1 memperlihatkan data uji buatan untuk kata satuan dan gambar 5.2 memperlihatkan data uji buatan untuk kata banyak.

Untuk data uji yang menggunakan objek nyata, digunakan sisi depan dari kardus kemasan produk “Seagate Backup Plus Slim 2 TB”. Data uji yang digunakan adalah kata-kata dalam Bahasa Inggris diluar nama merek dan singkatan. Kata-kata tersebut adalah: “BACKUP”, “PLUS”, “Slim”, “Portable”, “Storage”, “Backup”, “made”, dan “easy”. Gambar 5.3 menunjukkan kata-kata dari sisi depan dari kardus produk “Seagate Backup Plus Slim 2 TB” yang digunakan sebagai data uji.

Gambar 5.1 Data Uji Buatan Kata Satuan



Gambar 5.2 Data Uji Buatan Kata Banyak



Gambar 5.3 Gambar Kata-Kata yang Digunakan sebagai Data Uji Bentuk Nyata

5.2.2 Skenario Pengujian Pengambilan Area ROI

Skenario pengujian pada tahap ini adalah pengujian yeng digunakan untuk memeriksa apakah pengambilan area ROI berjalan dengan baik atau belum. Skenario yang digunakan untuk melakukan pengujian pengambilan area ROI terbagi menjadi dua, yaitu pengujian menggunakan data uji buatan dan penguian menggunakan data uji objek nyata.

5.2.2.1 Skenario Pengujian Pengambilan Area ROI pada Data Uji Buatan

Pengujian akan dilakukan dengan menggunakan data uji buatan yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan meletakan kamera dihadapan data uji dengan jarak 25 sentimeter. Perangakat bergerak uji diletakan di atas tripod. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Sentuhan dilakukan sebanyak satu kali pada bagian tengah masing-masing gambar kata. Untuk setiap kasus sukses bernilai 1, sementara untuk setiap kasus gagal akan diberi nilai 0. Tabel 5.3 memperlihatkan skenario dari pengujian pengambilan ROI pada data uji buatan. Hasil dari pengujian pengambilan area ROI ditunjukan pada Tabel 5.3 dan Tabel 5.4.

Tabel 5.2 Pengujian Pengambilan Area ROI pada Data Uji Buatan

ID	T-001-1
Nama	Pengambilan Area ROI pada Data Uji Buatan
Tujuan uji coba	Mengetahui apakah proses pengambilan ROI dapat berjalan dengan baik pada kondisi pengujian menggunakan data uji buatan
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang diletakan pada jarak 25 sentimeter didepan data uji buatan dalam ruangan dengan pencahayaan baik.
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji buatan
Keluaran yang diharapkan	Area ROI yang sesuai dengan gambar kata yang dipilih.

Tabel 5.3 Pengujian Pengambilan Area ROI pada Data Uji Kata Satuan

Jenis *Font*	Ukuran	Warna			Rata-Rata Nilai Ukuran 28 pt	Rata-Rata Nilai Ukuran 48 pt	**Rata-Rata Nilai Per *Font***
		Hitam di atas putih	Merah diatas putih	Kuning di atas biru			
Consolas	28 pt	1	1	1	1		**0.83**
	48 pt	1	1	0		0.67	
New Courier	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Arial Black	28 pt	1	1	1	1		**0.83**
	48 pt	1	0	1		0.67	

Times New Roman	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Rata-Rata Nilai		**1**	**0.87**	**0.87**	**1**	**0.83**	**0.91**

Tabel 5.4 Pengujian Pengambilan Area ROI pada Data Uji Kata Banyak

Kata	Ukuran	Warna			Rata-Rata Nilai Ukuran 28 pt	Rata-Rata Nilai Ukuran 48 pt	**Rata-Rata Nilai Per Kata**
		Hitam di atas putih	Merah diatas putih	Kuning di atas biru			
One	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Two	28 pt	0	0	1	0.33		**0.33**
	48 pt	0	1	0		0.33	
Three	28 pt	0	0	0	0		**0.5**
	48 pt	1	1	1		1	
Four	28 pt	1	1	0	0.67		**0.83**
	48 pt	1	1	1		1	
Five	28 pt	0	1	0	0.33		**0.66**
	48 pt	1	1	1		1	
Six	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Seven	28 pt	0	1	0	0.33		**0.66**
	48 pt	1	1	1		1	
Eight	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Nine	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Rata-Rata Nilai		**0.72**	**0.89**	**0.72**	**0.63**	**0.93**	**0.78**

Dari hasil uji dengan data kata satuan pada Tabel 5.3 di atas, terlihat untuk nilai pengujian pada setiap warna cukup bagus , dengan nilai 1 untuk warna teks hitam di atas latar belakang putih dan 0.87 untuk warna teks merah di atas latar belakang putih dan warna teks kuning di atas latar belakang biru. Untuk variabel ukuran, nilai pengujian juga cukup bagus, dengan nilai 1 untuk ukuran 28 point dan nilai 0.83 untuk ukuran 48 point.

Untuk variabel jenis *font*, nilai pengujian sebesar untuk jenis *font* New Courier dan Times New Roman, dan 0.83 untuk jenis *font* Consolas dan Arial Black.

Dari hasil uji dengan data kata banyak pada Tabel 5.4 di atas, terdapat sejumlah kata dengan nilai rata-rata agak rendah, yaitu kata “Two”, “Three”, “Five”, dan “Seven” Untuk ukuran huruf, nilai rata-rata untuk ukuran 28 point juga agak rendah, dengan rata-rata nilai 0.63. Menurut penulis, hal tersebut dapat dikarenakan oleh bentuk tepian yang terdeteksi oleh aplikasi. Bentuk huruf tertentu serta ukuran ROI yang kecil dapat mempengaruhi bentuk tepian yang terdeteksi.

5.2.2.2 Skenario Pengujian Pengambilan Area ROI pada Data Uji Objek Nyata

Pengujian akan dilakukan dengan menggunakan data uji objek nyata yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan memegang perangkat uji di depan objek uji dengan keadaan melintang. Jarak kamera disesuiakan dengan ukuran kata yang diuji. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).
Sentuhan dilakukan pada bagian tengah masing-masing gambar kata. Untuk setiap kasus sukses bernilai 1, sementara untuk setiap kasus gagal akan diberi nilai 0. Tabel 5.5 memperlihatkan skenario dari pengujian pengambilan ROI pada data uji objek nyata. Hasil dari pengujian pengambilan area ROI ditunjukan pada Tabel 5.6.

Tabel 5.5 Pengujian Pengambilan Area ROI pada Data Uji Objek Nyata

ID	T-001-2
Nama	Pengambilan Area ROI pada Data Uji Objek Nyata
Tujuan uji coba	Mengetahui apakah proses pengambilan ROI dapat berjalan dengan baik pada kondisi pengujian menggunakan data uji objek nyata
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada

	perangkat uji yang dipegang secara melintang di depan data uji
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji objek nyata
Keluaran yang diharapkan	Area ROI yang sesuai dengan gambar kata yang dipilih.

Tabel 5.6 Pengujian Pengambilan Area ROI pada Data Uji Objek Nyata

Kata	Nilai
BACKUP	1
PLUS	1
Slim	0
Portable	1
Storage	1
Backup	1
made	1
easy	1
Rata-Rata	**0.87**

Dari hasil uji menggunakan data uji objek nyata, terlihat bahwa semua ROI berhasil didapatkan dengan baik, kecuali untuk kata “Slim”. Menurut pengamatan penulis pada hasil deteksi tepi, kesalahan terjadi akibat adanya tepi dari gambar diluar gambar teks “Slim” yang masuk ke dalam wilayah gambar teks tersebut, sehingga aplikasi menganggap tepian tersebut adalah tepian dari gambar teks “Slim” dan terus mendeteksi wilayah ROI hingga akhirnya keluar dari wilayah *frame* dan menghasilkan kesalahan.

5.2.3 Skenario Pengujian Pengenalan Kata dengan OCR

Skenario pengujian pada tahap ini adalah pengujian yeng digunakan untuk memeriksa apakah pengenalan kata dengan OCR telah berjalan dengan baik atau belum. Skenario yang

digunakan untuk melakukan pengujian pengenalan kata dengan OCR terbagi menjadi dua, yaitu pengujian menggunakan data uji buatan dan pengujian menggunakan data uji objek nyata.

5.2.3.1 Skenario Pengujian Pengenalan Kata dengan OCR pada Data Uji Buatan

Pengujian akan dilakukan dengan menggunakan data uji buatan yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan meletakan kamera dihadapan data uji dengan jarak 25 sentimeter. Perangakat bergerak uji diletakan di atas tripod. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Pengujian OCR dilakukan dalam satu rangkaian dengan pengujian pengambilan area ROI. Apabila pada saat pengujian ROI area ROI tidak bisa didapatkan, maka hasil OCR dianggap tidak ada. Untuk setiap kasus sukses bernilai 1, sementara untuk setiap kasus gagal akan diberi nilai 0. Pengenalan OCR dilihat kesamaan antara hasil OCR dengan gambar ROI yang terambil. Tabel 5.4 memperlihatkan skenario dari pengujian pengenalan kata dengan OCR pada data uji buatan. Hasil dari pengenalan kata dengan OCR ditunjukan pada Tabel 5.8 dan Tabel 5.9.

Tabel 5.7 Pengujian Pengenalan Kata dengan OCR pada Data Uji Buatan

ID	T-002-1
Nama	Pengenalan Kata dengan OCR pada Data Uji Buatan
Tujuan uji coba	Mengetahui apakah proses pengenalan kata menggunakan OCR berjalan dengan baik pada kondisi pengujian menggunakan data uji buatan
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang diletakan pada jarak 25 sentimeter didepan data uji dalam ruangan dengan pencahayaan baik.
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji buatan

Keluaran yang diharapkan	Hasil pengenalan kata yang sesuai dengan gambar teks pada wilayah ROI.

Tabel 5.8 Pengujian Pengenalan Kata dengan OCR pada Data Uji Kata Satuan

Jenis *Font*	Ukuran	Warna			Rata-Rata Nilai Ukuran 28 pt	Rata-Rata Nilai Ukuran 48 pt	**Rata-Rata Nilai Per *Font***
		Hitam di atas putih	Merah diatas putih	Kuning di atas biru			
Consolas	28 pt	1	1	1	1		**0.83**
	48 pt	1	1	0		0.67	
New Courier	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Arial Black	28 pt	1	1	1	1		**0.83**
	48 pt	1	0	1		0.67	
Times New Roman	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Rata-Rata Nilai		**1**	**0.87**	**0.87**	**1**	**0.83**	**0.91**

Tabel 5.9 Pengujian Pengenalan Kata dengan OCR pada Data Uji Kata Banyak

Kata	Ukuran	Warna			Rata-Rata Nilai Ukuran 28 pt	Rata-Rata Nilai Ukuran 48 pt	**Rata-Rata Nilai Per Kata**
		Hitam di atas putih	Merah diatas putih	Kuning di atas biru			
One	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Two	28 pt	0	0	1	0.33		**0.33**
	48 pt	0	1	0		0.33	
Three	28 pt	0	0	0	0		**0.5**
	48 pt	1	1	1		1	
Four	28 pt	1	1	0	0.67		**0.83**
	48 pt	1	1	1		1	
Five	28 pt	0	1	0	0.33		**0.66**

	48 pt	1	1	1		1	
Six	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Seven	28 pt	0	1	0	0.33		**0.66**
	48 pt	1	1	1		1	
Eight	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Nine	28 pt	1	1	0	0.67		**0.67**
	48 pt	1	1	0		0.67	
Rata-Rata Nilai		**0.72**	**0.89**	**0.61**	**0.59**	**0.89**	**0.74**

Dari hasil uji dengan data kata satuan pada Tabel 5.8 di atas, terlihat untuk nilai pengujian pada setiap kasus tetap sama. Artinya, untuk setiap kasus pengambilan ROI yang sukses, OCR berjalan dengan baik dan memberikan hasil yang sesuai dengan gambar kata yang dipilih.

Dari hasil uji dengan data kata banyak pada Tabel 5.9 di atas, terlihat adanya perbedaan nilai pada kata “Nine” dengan warna teks kuning di atas latar belakang biru. OCR tidak dapat memberikan hasil pengenalan untuk kasus tersebut meskipun ROI telah terambil dengan baik. Diluar kasus tersebut, OCR dapat memberikan hasil yang sesuai dengan ROI yang didapatkan.

5.2.3.2 Skenario Pengujian Pengenalan Kata dengan OCR pada Data Uji Objek Nyata

Pengujian akan dilakukan dengan menggunakan data uji objek nyata yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan memegang perangkat uji di depan objek uji dengan keadaan melintang. Jarak kamera disesuiakan dengan ukuran kata yang diuji. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Pengujian OCR dilakukan dalam satu rangkaian dengan pengujian pengambilan area ROI. Apabila pada saat pengujian ROI area ROI tidak bisa didapatkan, maka hasil OCR dianggap tidak ada. Untuk setiap kasus sukses bernilai 1, sementara untuk setiap kasus gagal akan diberi nilai 0. Pengenalan OCR dilihat

kesamaan antara hasil OCR dengan gambar ROI yang terambil. Tabel 5.10 memperlihatkan skenario dari pengujian pengenalan kata dengan OCR pada data uji objek nyata. Hasil dari pengujian pengenalan kata dengan OCR ditunjukkan pada Tabel 5.11

Tabel 5.10 Pengujian Pengenalan Kata dengan OCR pada Data Uji Objek Nyata

ID	T-002-2
Nama	Pengenalan Kata dengan OCR pada Data Uji Objek Nyata
Tujuan uji coba	Mengetahui apakah proses pengenalan kata menggunakan OCR berjalan dengan baik pada kondisi pengujian menggunakan data uji objek nyata.
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang dipegang secara melintang di depan data uji
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji objek nyata
Keluaran yang diharapkan	Hasil pengenalan kata yang sesuai dengan gambar teks pada wilayah ROI.

Tabel 5.11 Pengujian Pengenalan Kata dengan OCR pada Data Uji Objek Nyata

Kata	Nilai
BACKUP	1
PLUS	1
Slim	0
Portable	1
Storage	1
Backup	1
made	1
easy	1
Rata-Rata	**0.87**

Dari Tabel 5.11, terlihat bahwa OCR dapat mengenali kata dengan baik untuk setiap ROI yang dapat terambil dengan sukses.

5.2.4 Skenario Pengujian Penerjemahan Kata

Skenario pengujian pada tahap ini adalah pengujian yeng digunakan untuk memeriksa apakah penerjemahan kata berjalan dengan baik atau tidak. Skenario yang digunakan untuk melakukan pengujian penerjemahan kata terbagi menjadi dua, yaitu pengujian menggunakan data uji buatan dan pengujian menggunakan data uji objek nyata.

5.2.4.1 Skenario Pengujian Penerjemahan Kata pada Data Uji Buatan

Pengujian akan dilakukan dengan menggunakan data uji buatan yang telah dijelaskan pada subbab 5.2.1. Pengujian penerjemahan kata dilakukan dengan memeriksa apakah kata-kata yang digunakan dalam data uji dapat diterjemahkan dengan baik. Pengujian ini dilukakukan dalam satu rangkaian dengan pengujian pengambilan area ROI dan pengenalan kata menggunakan OCR. Masukan dari pengujian ini adalah hasil pengenalan OCR dari setiap kata pada data uji yang dapat dengan sukses dikenali dan sesuai dengan gambar teks. Tabel 5.12 memperlihatkan skenario dari pengujian penerjemahan kata pada data uji buatan. Tabel 5.13 menunjukkan hasil dari pengujian penerjemahan kata pada data uji buatan.

Tabel 5.12 Pengujian Penerjemahan Kata pada Data Uji Buatan

ID	T-004-1
Nama	Penerjemahan Kata pada Data Uji Buatan
Tujuan uji coba	Mengetahui apakah kata yang digunakan pada data uji buatan dapat diterjemahkan dengan baik

Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang diletakan pada jarak 25 sentimeter didepan data uji dalam ruangan dengan pencahayaan baik
Masukan	Kata hasil pengenalan OCR yang sudah sesuai dengan gambar kata yang dipilih
Keluaran yang diharapkan	Hasil terjemahan dalam bahasa Indonesia yang sesuai.

Tabel 5.13 Hasil Pengujian Penerjemahan Kata-Kata pada Data Uji Buatan

Kata	Hasil Terjemahan	Status Pengujian
Copyrights	Hak Cipta	Berhasil
One	Satu	Berhasil
Two	Dua	Berhasil
Three	Tiga	Berhasil
Four	Empat	Berhasil
Five	Lima	Berhasil
Six	Enam	Berhasil
Seven	Tujuh	Berhasil
Eight	Delapan	Berhasil
Nine	Sembilan	Berhasil

Dari Tabel 5.13, terlihat bahwa setiap kata bahasa Inggris yang digunakan pada data uji buatan dapat diterjemahkan dengan baik ke dalam bahasa Indonesia.

5.2.4.2 Skenario Pengujian Penerjemahan Kata pada Data Uji Objek Nyata

Pengujian akan dilakukan dengan menggunakan data uji objek nyata yang telah dijelaskan pada subbab 5.2.1. Pengujian penerjemehan kata dilakukan dengan memeriksa apakah kata-kata yang digunakan dalam data uji dapat diterjemahkan dengan baik. Pengujian ini dilukakukan dalam satu rangkaian dengan

pengujian pengambilan area ROI dan pengenalan kata menggunakan OCR. Masukan dari pengujian ini adalah hasil pengenalan OCR dari setiap kata pada data uji yang dapat dengan sukses dikenali dan sesuai dengan gambar teks. Tabel 5.14 memperlihatkan skenario dari pengujian penerjemahan kata pada data uji objek nyata. Tabel 5.15 menunjukkan hasil dari pengujian penerjemahan kata pada data uji objek nyata.

Tabel 5.14 Pengujian Penerjemahan Kata pada Data Uji Objek Nyata

ID	T-004-2
Nama	Penerjemahan Kata pada Data Uji Objek Nyata
Tujuan uji coba	Mengetahui apakah kata yang digunakan pada data uji objek nyata dapat diterjemahkan dengan baik
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang dipegang secara melintang di depan data uji
Masukan	Kata hasil pengenalan OCR yang sudah sesuai dengan gambar kata yang dipilih
Keluaran yang diharapkan	Hasil terjemahan dalam Bahasa Indonesia yang sesuai.

Tabel 5.15 Hasil Pengujian Penerjemahan Kata-Kata pada Data Uji Objek Nyata

Kata	Hasil Terjemahan	Status Pengujian
BACKUP	CADANGAN	Berhasil
PLUS	PLUS	Gagal
Slim	-	Gagal
Portable	Portable	Gagal
Storage	Penyimpanan	Berhasil
Backup	Cadangan	Berhasil
made	dibuat	Berhasil
easy	mudah	Berhasil

Dari Tabel x, terlihat bahwa ada dua kata yang tidak dapat diterjemahkan dengan baik oleh Yandex. Translate API diluar kata yang gagal dieteksi (“Slim”), yaitu “PLUS” dan “Portable”. Hal ini menunjukkan Yandex. Translate API masih belum mampu untuk menerjemahkan kata-kata tertentu ke dalam Bahasa Indonesia, meskipun kata tersebut memiliki sudah memliki arti dalam Bahasa Indonesia.

5.2.5 Skenario Pengujian Penggambaran Hasil Penerjemahan

Skenario pengujian pada tahap ini adalah pengujian yeng digunakan untuk memeriksa apakah penggambaran hasil penerjemahan berjalan dengan baik atau tidak. Skenario yang digunakan untuk melakukan pengujian penggambaran hasil terjemahan terbagi menjadi dua, yaitu pengujian menggunakan data uji buatan dan pengujian menggunakan data uji objek nyata.

5.2.5.1 Skenario Pengujian Penggambaran Hasil Penerjemahan pada Data Uji Buatan

Pengujian akan dilakukan dengan menggunakan data uji buatan yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan meletakan kamera dihadapan data uji dengan jarak 25 sentimeter. Perangakat bergerak uji diletakan di atas tripod. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Pengujian penggambaran hasil penerjemahan dilakukan dalam satu rangkaian dengan pengujian pengambilan area ROI dan pengenalan kata dengan OCR. Apabila pada suatu kasus pengujian ROI area ROI tidak bisa didapatkan atau OCR gagal mengenali gambar kata, maka kasus pengujian tersebut dianggap gagal. Untuk setiap kasus sukses bernilai 1, sementara untuk setiap kasus gagal akan diberi nilai 0. Penggambaran hasil terjemahan dilihat dari tampilnya hasil penerjemahan pada layar aplikasi. Tabel 5.16 memperlihatkan skenario dari pengujian

penggambran hasil penerjemahan pada data uji buatan. Hasil dari pengujian penggambaran hasil penerjemahan ditunjukkan pada Tabel 5.17 dan Tabel 5.18.

Tabel 5.16 Pengujian Penggambaran Hasil Penerjemahan pada Data Uji Buatan

ID	T-005-1
Nama	Penggambaran Hasil Penerjemahan pada Data Uji Buatan
Tujuan uji coba	Mengetahui apakah penggambaran hasil penerjemahan berjalan dengan baik pada kondisi pengujian menggunakan data uji buatan.
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang diletakan pada jarak 25 sentimeter didepan data uji dalam ruangan dengan pencahayaan baik.
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji buatan
Keluaran yang diharapkan	Gambar hasil penerjemahan

Tabel 5.17 Pengujian Penggambaran Hasil Penerjemahan pada Data Uji Kata Satuan

Jenis *Font*	Ukuran	Warna			Rata-Rata Nilai Ukuran 28 pt	Rata-Rata Nilai Ukuran 48 pt	**Rata-Rata Nilai Per *Font***
		Hitam di atas putih	Merah diatas putih	Kuning di atas biru			
Consolas	28 pt	1	1	1	1		**0.83**
	48 pt	1	1	0		0.67	
New Courier	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Arial Black	28 pt	1	1	1	1		**0.83**
	48 pt	1	0	1		0.67	
Times New	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	

Roman							
Rata-Rata Nilai		**1**	**0.87**	**0.87**	**1**	**0.83**	**0.91**

Tabel 5.18 Pengujian Penggambaran Hasil Penerjemahan pada Data Uji Kata Banyak

Kata	Ukuran	Warna			Rata-Rata Nilai Ukuran 28 pt	Rata-Rata Nilai Ukuran 48 pt	**Rata-Rata Nilai Per Kata**
		Hitam di atas putih	Merah diatas putih	Kuning di atas biru			
One	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Two	28 pt	0	0	1	0.33		**0.33**
	48 pt	0	1	0		0.33	
Three	28 pt	0	0	0	0		**0.5**
	48 pt	1	1	1		1	
Four	28 pt	1	1	0	0.67		**0.83**
	48 pt	1	1	1		1	
Five	28 pt	0	1	0	0.33		**0.66**
	48 pt	1	1	1		1	
Six	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Seven	28 pt	0	1	0	0.33		**0.66**
	48 pt	1	1	1		1	
Eight	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Nine	28 pt	1	1	0	0.67		**0.67**
	48 pt	1	1	0		0.67	
Rata-Rata Nilai		**0.72**	**0.89**	**0.61**	**0.59**	**0.89**	**0.74**

Dari hasil pengujian di atas, terlihat bahwa untuk setiap pengujian OCR yang berhasil, hasil terjemahan dapat tergambar dengan baik.

5.2.5.2 Skenario Pengujian Penggambaran Hasil Penerjemahan pada Data Uji Objek Nyata

Pengujian akan dilakukan dengan menggunakan data uji objek nyata yang telah dijelaskan pada subbab 5.2.1. Pengujian

dilakukan dengan memegang perangkat uji di depan objek uji dengan keadaan melintang. Jarak kamera disesuiakan dengan ukuran kata yang diuji. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Pengujian penggambaran hasil penerjemahan dilakukan dalam satu rangkaian dengan pengujian pengambilan area ROI dan pengenalan kata dengan OCR. Apabila pada suatu kasus pengujian ROI area ROI tidak bisa didapatkan atau OCR gagal mengenali gambar kata, maka kasus pengujian tersebut dianggap gagal. Untuk setiap kasus sukses bernilai 1, sementara untuk setiap kasus gagal akan diberi nilai 0. Penggambaran hasil terjemahan dilihat dari tampilnya hasil penerjemahan pada layar aplikasi. Tabel 5.19 memperlihatkan skenario dari pengujian penggambran hasil penerjemahan pada data uji objek nyata. Hasil dari pengujian penggambaran hasil penerjemahan ditunjukkan pada Tabel 5.20.

Tabel 5.19 Pengujian Penggambaran Hasil Penerjemahan pada Data Uji Objek Nyata

ID	T-005-2
Nama	Penggambaran Hasil Penerjemahan pada Data Objek Nyata
Tujuan uji coba	Mengetahui apakah penggambaran hasil penerjemahan berjalan dengan baik pada kondisi pengujian menggunakan data uji objek nyata.
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang dipegang secara melintang di depan data uji.
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji buatan
Keluaran yang diharapkan	Gambar hasil penerjemahan

Tabel 5.20 Hasil Pengujian Penggambaran Hasil Penerjemahan Kata-Kata pada Data Uji Objek Nyata

Kata	Nilai
BACKUP	1
PLUS	1
Slim	0
Portable	1
Storage	1
Backup	1
made	1
easy	1
Rata-Rata	**0.87**

Dari hasil pengujian di atas, terlihat bahwa untuk setiap pengujian OCR yang berhasil, hasil terjemahan dapat tergambar dengan baik.

5.2.6 Skenario Pengujian Pelacakan Area ROI (Keadaan Diam)

Skenario pengujian pada tahap ini adalah pengujian yeng digunakan untuk memeriksa apakah pelacakan area ROI berjalan dengan baik atau tidak pada keadaan diam. Skenario yang digunakan untuk melakukan pengujian pelacakan area ROI pada keadaan diam terbagi menjadi dua, yaitu pengujian menggunakan data uji buatan dan pengujian menggunakan data uji objek nyata.

5.2.6.1 Skenario Pengujian Pelacakan Area ROI (Keadaan Diam) pada Data Uji Buatan

Pengujian akan dilakukan dengan menggunakan data uji buatan yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan meletakan kamera dihadapan data uji dengan jarak 25 sentimeter. Perangakat bergerak uji diletakan di atas tripod. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Sentuhan dilakukan pada bagian tengah masing-masing gambar kata. Apabila setelah melakukan sentuhan ROI agal terambil, sentuhan diulangi hingga ROI terambil dengan baik dan sesuai dengan area gambar kata yang diuji. Pada setiap kata yang diuji, hasil uji dikatakan sukses apabila lebih dari 50 persen gambar kata berada pada persegi ROI selama 2 detik setelah hasil penerjemahan tergambar pada layar. Waktu 2 detik dipilih untuk mensimulasikan waktu yang dibutuhkan pengguna untuk melihat hasil terjemahan. Tabel 5.21 memperlihatkan skenario dari pengujian pelacakan area ROI pada keadaan diam pada data uji buatan. Tabel 5.22 dan 5.23 menjukkan hasil dari pengujian pelacakan area ROI pada keadaan diam.

Tabel 5.21 Pengujian Pelacakan Area ROI (Keadaan Diam) pada Data Uji Buatan

ID	T-006-1
Nama	Pengujian Pelacakan Area ROI (Keadaan Diam) pada Data Uji Buatan
Tujuan uji coba	Mengetahui apakah pelacakan area ROI berjalan dengan baik pada kondisi diam menggunakan data uji buatan.
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang diletakan pada jarak 25 sentimeter didepan data uji dalam ruangan dengan pencahayaan baik.
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji buatan
Keluaran yang diharapkan	ROI terlacak dengan baik sesuai dengan posisi gambar ROI yang sebenarnya.

Tabel 5.22 Pengujian Pelacakan Area ROI (Keadaan Diam) pada Data Uji Kata Satuan

Jenis *Font*	Ukuran	Warna			Rata-Rata Nilai Ukuran 28 pt	Rata-Rata Nilai Ukuran 48 pt	**Rata-Rata Nilai Per *Font***
		Hitam di atas putih	Merah diatas putih	Kuning di atas biru			
Consolas	28 pt	0	1	1	0.67		**0.83**
	48 pt	1	1	1		1	
New Courier	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Arial Black	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Times New Roman	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Rata-Rata Nilai		**0.87**	**1**	**1**	**0.92**	**1**	**0.96**

Tabel 5.23 Pengujian Pelacakan Area ROI (Keadaan Diam) pada Data Uji Kata Banyak

Kata	Ukuran	Warna			Rata-Rata Nilai Ukuran 28 pt	Rata-Rata Nilai Ukuran 48 pt	**Rata-Rata Nilai Per Kata**
		Hitam di atas putih	Merah diatas putih	Kuning di atas biru			
One	28 pt	0	1	1	0.67		**0.83**
	48 pt	1	1	1		1	
Two	28 pt	0	1	1	0.67		**0.67**
	48 pt	0	1	1		0.67	
Three	28 pt	1	1	1	1		**1**
	48 pt	1	1	1		1	
Four	28 pt	0	1	1	0.67		**0.67**
	48 pt	0	1	1		0.67	
Five	28 pt	1	0	1	0.67		**0.83**
	48 pt	1	1	1		1	
Six	28 pt	0	1	1	0.67		**0.83**
	48 pt	1	1	1		1	
Seven	28 pt	0	1	1	0.67		**0.83**
	48 pt	1	1	1		1	

Eight	28 pt	1	1	1	1		**0.83**
	48 pt	1	1	0		0.67	
Nine	28 pt	1	1	0	0.67		**0.83**
	48 pt	1	1	1		1	
Rata-Rata Nilai		**0.61**	**0.94**	**0.89**	**0.74**	**0.89**	**0.81**

Dari hasil pengujian untuk kata satuan pada tabel x, terlihat bahwa nilai pengujian cukup baik untuk seluruh variabel. Dari hasil pengujian untuk kata banyak pada tabel x, terlihat barwa nilai pengujian rata-rata untuk warna hitam di atas latar belakang putih berada jauh dibawah kedua warna lainnya. Menurut penulis, hal ini dapat disebabkan karena pembagian bin histogram pada *channel* warna V dari ruang warna HSV yang digunakan pada algoritma pelacakan, yang menandakan gelap terangnya suatu warna.

5.2.6.2 Skenario Pengujian Pelacakan Area ROI (Keadaan Diam) pada Data Uji Objek Nyata

Pengujian akan dilakukan dengan menggunakan data uji objek nyata yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan memegang perangkat uji di depan objek uji dengan keadaan melintang. Jarak kamera disesuaikan dengan ukuran kata yang diuji. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Sentuhan dilakukan pada bagian tengah masing-masing gambar kata. Apabila setelah melakukan sentuhan ROI agal terambil, sentuhan diulangi hingga ROI terambil dengan baik dan sesuai dengan area gambar kata yang diuji. Pada setiap kata yang diuji, hasil uji dikatakan sukses apabila lebih dari 50 persen gambar kata berada pada persegi ROI selama 2 detik setelah hasil penerjemahan tergambar pada layar. Waktu 2 detik dipilih untuk mensimulasikan waktu yang dibutuhkan pengguna untuk melihat hasil terjemahan. Tabel 5.24 memperlihatkan skenario dari pengujian pelacakan area ROI pada keadaan diam pada data uji

objek nyata. Tabel 5.25 menjukkan hasil dari pengujian pelacakan area ROI pada keadaan diam.

Tabel 5.24 Pengujian Pelacakan Area ROI (Keadaan Diam) pada Data Uji Objek Nyata

ID	T-006-2
Nama	Pengujian Pelacakan Area ROI (Keadaan Diam) pada Data Uji Buatan
Tujuan uji coba	Mengetahui apakah pelacakan area ROI berjalan dengan baik pada kondisi diam menggunakan data uji buatan.
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang dipegang secara melintang di depan data uji.
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji buatan
Keluaran yang diharapkan	ROI terlacak dengan baik sesuai dengan posisi gambar ROI yang sebenarnya.

Tabel 5.25 Hasil Pelacakan Area ROI (Keadaan Diam) pada Data Uji Objek Nyata

Kata	Nilai
BACKUP	1
PLUS	1
Slim	0
Portable	1
Storage	1
Backup	0
made	0
easy	0
Rata-Rata	**0.5**

Dari hasil pengujian di atas, diluar kata “Slim” yang tidak dapat terdeteksi, terlihat adanya kegagalan pada kata “Backup”,

“made”, dan “easy”. Menurut penulis, kesalahan dapat terjadi dengan penyebab yang sama seperti pada pengujian dengan data uji buatan, yaitu karena pembagian bin histogram pada *channel* warna V dari ruang warna HSV yang digunakan pada algoritma pelacakan, yang menandakan gelap terangnya suatu warna.

5.2.7 Skenario Pengujian Proses Pelacakan Area ROI (Keadaan Bergerak)

Skenario pengujian pada tahap ini adalah pengujian yeng digunakan untuk memeriksa apakah pelacakan area ROI berjalan dengan baik atau tidak pada keadaan bergerak. Skenario yang digunakan untuk melakukan pengujian pelacakan area ROI pada keadaan bergerak terbagi menjadi dua, yaitu pengujian menggunakan data uji buatan dan pengujian menggunakan data uji objek nyata.

5.2.7.1 Skenario Pengujian Proses Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Buatan

Pengujian akan dilakukan dengan menggunakan data uji buatan yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan meletakan kamera dihadapan data uji dengan jarak 25 sentimeter. Perangakat bergerak uji diletakan di atas tripod. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Sentuhan dilakukan pada bagian tengah masing-masing gambar kata. Apabila setelah melakukan sentuhan ROI agal terambil, sentuhan diulangi hingga ROI terambil dengan baik dan sesuai dengan area gambar kata yang diuji. Pada setiap pengujian, perangkat akan digerakan ke atas, bawah, kiri dan kanan secara berurutan dengan jarak 1.5 cm. Pada setiap kata yang diuji, hasil uji dikatakan sukses apabila lebih dari 50 persen gambar kata berada pada persegi ROI. Tabel 5.26 memperlihatkan skenario dari pengujian pelacakan area ROI pada keadaan bergerak pada

data uji buatan. Tabel 5.27 hingga 5.30 menjukkan hasil dari pengujian pelacakan area ROI pada keadaan bergerak.

Tabel 5.26 Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Buatan

ID	T-006-1
Nama	Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Buatan
Tujuan uji coba	Mengetahui apakah pelacakan area ROI berjalan dengan baik pada kondisi bergerak menggunakan data uji buatan.
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang diletakan pada jarak 25 sentimeter didepan data uji dalam ruangan dengan pencahayaan baik.
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji buatan
Keluaran yang diharapkan	ROI terlacak dengan baik sesuai dengan posisi gambar ROI yang sebenarnya.

Tabel 5.27 Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Kata Satuan Ukuran Huruf 28 Point

Warna	Arah	Jenis *Font*				**Nilai Rata-Rata per Warna**
		Consolas	New Courier	Arial Black	Times New Roman	
Hitam di atas putih	Atas	0	0	0	1	**0.19**
	Bawah	0	0	0	1	
	Kiri	0	0	0	1	
	Kanan	0	0	0	0	
Merah di atas putih	Atas	1	0	1	1	**0.75**
	Bawah	1	0	1	1	
	Kiri	1	0	1	1	
	Kanan	1	0	1	1	
Kuning di atas biru	Atas	1	1	1	0	**0.75**
	Bawah	1	1	1	0	
	Kiri	1	1	1	0	
	Kanan	1	1	1	0	

Nilai Rata-Rata per *Font*	**0.67**	**0.33**	**0.67**	**0.58**	
Rata-Rata					**0.56**

Tabel 5.28 Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Kata Satuan Ukuran Huruf 48 Point

Warna	Arah	Jenis *Font*				**Nilai Rata-Rata per Warna**
		Consolas	New Courier	Arial Black	Times New Roman	
Hitam di atas putih	Atas	1	1	1	1	**0.94**
	Bawah	1	1	1	1	
	Kiri	1	1	1	1	
	Kanan	1	0	1	1	
Merah di atas putih	Atas	1	1	1	0	**0.75**
	Bawah	1	1	1	0	
	Kiri	1	1	1	0	
	Kanan	1	1	1	0	
Kuning di atas biru	Atas	0	1	1	0	**0.69**
	Bawah	1	1	1	0	
	Kiri	1	1	1	0	
	Kanan	1	1	1	0	
Nilai Rata-Rata per *Font*		**0.92**	**0.92**	**1**	**0.25**	
Rata-Rata						**0.79**

Tabel 5.29 Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Kata Banyak Ukuran Huruf 28 Point

Kata	Arah	Warna			Nilai Rata-Rata per Kata
		Hitam di atas putih	Merah di atas putih	Kuning di atas biru	
One	Atas	0	1	1	**0.67**
	Bawah	1	1	1	
	Kiri	0	1	1	
	Kanan	0	1	0	
Two	Atas	0	1	1	**0.42**
	Bawah	0	1	1	
	Kiri	0	0	1	
	Kanan	0	0	0	

Three	Atas	0	1	0	**0.58**
	Bawah	0	1	0	
	Kiri	1	1	0	
	Kanan	1	1	1	
Four	Atas	0	1	0	**0.25**
	Bawah	0	0	1	
	Kiri	0	0	1	
	Kanan	0	0	0	
Five	Atas	0	0	1	**0.17**
	Bawah	0	0	1	
	Kiri	0	0	0	
	Kanan	0	0	0	
Six	Atas	0	1	0	**0.5**
	Bawah	1	1	0	
	Kiri	0	1	0	
	Kanan	1	0	1	
Seven	Atas	0	1	1	**0.5**
	Bawah	0	1	1	
	Kiri	0	1	1	
	Kanan	0	0	0	
Eight	Atas	0	1	0	**0.5**
	Bawah	1	1	1	
	Kiri	0	0	0	
	Kanan	1	1	0	
Nine	Atas	0	0	1	**0.33**
	Bawah	0	0	1	
	Kiri	0	0	0	
	Kanan	0	1	1	
Nilai Rata-Rata per Warna		**0.19**	**0.58**	**0.53**	
Rata-Rata:					**0.44**

Tabel 5.30 Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Kata Banyak Ukuran Huruf 48 Point

Kata	Arah	Warna			Nilai Rata-Rata per Kata
		Hitam di atas putih	Merah di atas putih	Kuning di atas biru	
One	Atas	1	1	1	**0.83**
	Bawah	1	1	1	

	Kiri	1	1	1	
	Kanan	0	0	1	
Two	Atas	1	0	1	**0.58**
	Bawah	1	0	1	
	Kiri	0	0	1	
	Kanan	1	0	1	
Three	Atas	1	1	1	**0.83**
	Bawah	1	1	1	
	Kiri	0	1	1	
	Kanan	1	1	1	
Four	Atas	1	1	1	**0.75**
	Bawah	1	1	1	
	Kiri	1	0	1	
	Kanan	0	0	1	
Five	Atas	1	1	1	**0.67**
	Bawah	1	1	1	
	Kiri	0	0	0	
	Kanan	1	0	1	
Six	Atas	1	1	1	**0.75**
	Bawah	1	1	1	
	Kiri	0	1	0	
	Kanan	1	0	1	
Seven	Atas	0	1	1	**0.67**
	Bawah	0	1	1	
	Kiri	0	1	1	
	Kanan	0	1	1	
Eight	Atas	1	1	1	**0.75**
	Bawah	1	1	1	
	Kiri	0	1	0	
	Kanan	1	0	1	
Nine	Atas	1	1	1	**0.92**
	Bawah	1	1	1	
	Kiri	0	1	1	
	Kanan	1	1	1	
Nilai Rata-Rata per Warna		**0.67**	**0.69**	**0.89**	
Rata-Rata					**0.75**

5.2.7.2 Skenario Pengujian Proses Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Objek Nyata

Pengujian akan dilakukan dengan menggunakan data uji objek nyata yang telah dijelaskan pada subbab 5.2.1. Pengujian dilakukan dengan meletakan kamera dihadapan data uji dengan jarak 25 sentimeter. Perangakat bergerak uji dipegang dengan tangan untuk mensimulasikan penggunaan oleh pengguna. Pengujian dilakukan pada ruangan dengan pencahayaan yang baik (lampu LED 14 Watt).

Sentuhan dilakukan pada bagian tengah masing-masing gambar kata. Apabila setelah melakukan sentuhan ROI agal terambil, sentuhan diulangi hingga ROI terambil dengan baik dan sesuai dengan area gambar kata yang diuji. Pada setiap pengujian, perangkat akan digerakan ke atas, bawah, kiri dan kanan secara berurutan dengan jarak 1.5 cm. Pada setiap kata yang diuji, hasil uji dikatakan sukses apabila lebih dari 50 persen gambar kata berada pada persegi ROI. Tabel 5.31 memperlihatkan skenario dari pengujian pelacakan area ROI pada keadaan bergerak pada data uji objek nyata. Tabel 5.32 menjukkan hasil dari pengujian pelacakan area ROI pada keadaan bergerak.

Tabel 5.31 Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Objek Nyata

ID	T-006-1
Nama	Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Objek Nyata
Tujuan uji coba	Mengetahui apakah pelacakan area ROI berjalan dengan baik pada kondisi bergerak menggunakan data uji objek nyata.
Kondisi awal	Aplikasi dalam keadaan aktif serta berjalan pada perangkat uji yang dipegang secara melintang di depan data uji.
Masukan	Sentuhan pada bagian tengah dari setiap gambar kata yang diuji pada data uji buatan
Keluaran	ROI terlacak dengan baik sesuai dengan posisi

yang diharapkan	gambar ROI yang sebenarnya.

Tabel 5.32 Pengujian Pelacakan Area ROI (Keadaan Bergerak) pada Data Uji Objek Nyata

Kata	Arah	Hasil	**Rata-Rata per Kata**
BACKUP	Atas	1	**1**
	Bawah	1	
	Kiri	1	
	Kanan	1	
PLUS	Atas	1	**0.75**
	Bawah	1	
	Kiri	0	
	Kanan	1	
Slim	Atas	0	**0**
	Bawah	0	
	Kiri	0	
	Kanan	0	
Portable	Atas	0	**0**
	Bawah	0	
	Kiri	0	
	Kanan	0	
Storage	Atas	0	**0**
	Bawah	0	
	Kiri	0	
	Kanan	0	
Backup	Atas	0	**0.25**
	Bawah	1	
	Kiri	0	
	Kanan	0	
made	Atas	0	**0.5**
	Bawah	1	

	Kiri	0	
	Kanan	1	
easy	Atas	0	**0**
	Bawah	0	
	Kiri	0	
	Kanan	0	
Rata-Rata		**0.31**	

5.3 Pengujian Aplikasi Terhadap Pengguna

Subbab ini membahas pengujian yang dilakukan pada pengguna aplikasi. Pengujian untuk pengguna dilakukan untuk mengetahui apakah aplikasi sudah sesuai dengan kebutuhan pengguna atau belum. Pengujian juga dilakukan untuk mengetahui apakah pengguna nyaman saat melakukan interaksi dengan aplikasi.

5.3.1 Skenario Uji Coba oleh Pengguna

Penguji diminta untuk mencoba aplikasi dengan cara menerjemahkan gambar kata pada beberapa objek yang dapat ditemui pada lingkungan sehari-hari. Objek yang digunakan adalah papan penunjuk bergambar [32], kardus kemasan produk "Seagate Backup Plus Slim 2 TB", dan koran TheJakartaPost. Penguji dipersilakan untuk mencoba aplikasi menggunakan kata-kata yang terdapat pada objek-objek tersebut. Gambar 5.4 hingga 5.6 menunjukkan objek uji yang diuji coba oleh pengguna.

Gambar 5.4 Data Uji Papan Penunjuk Bergambar

Gambar 5.5 Data Uji Kardus Kemasan Produk "Seagate Backup Plus Slim 2 TB"

Gambar 5.6 Data Uji Koran "TheJakartaPost

Jumlah pengguna yang akan menguji permainan sebanyak 5 (lima) orang. Pengujian dilakukan dengan menaruh objek di hadapan pengguna di dalam ruangan dengan pencahayaan yang baik. Pengguna memegang perangkat uji secara horizontal dan diarahkan ke arah objek uji. Sebelumnya, pengguna diberi penjelasan terlebih dahulu mengenai fungsi aplikasi dan bagaimana cara menggunakannya.

Setelah melakukan pengujian, tiap penguji akan mendapatkan kuesioner untuk mendapatkan penilaian pengguna mengenai kenyamanan dan kepuasan pengguna dalam menggunakan aplikasi yang diuji. Selain itu, pengguna juga diminta untuk memberikan kritik dan saran yang akan digunakan untuk pengembangan aplikasi di masa mendatang.

Aspek-aspek yang diukur dalam kuesioner pengguna terbagi menjadi tiga bagian, yaitu kenyamanan aplikasi, kinerja

aplikasi, dan kemudahan penggunaan aplikasi. Tiga aspek tersebut dibagi lagi ke dalam beberapa indikator. Tabel 5.33 menjelaskan aspek yang diukur beserta indikatornya.

Tabel 5.33 Aspek-Aspek Pengujian Beserta Indikatornya

Aspek	Indikator
Kenyamanan Aplikasi	Kenyamanan tampilan antar muka
	Kenyamanan menggunakan sentuhan sebagai metode masukan
	Kenyamanan melihat hasil terjemahan
Performa Aplikasi	Performa aplikasi
	Kecepatan mendapatkan hasil terjemahan
Kemudahan Penggunaan Aplikasi	Kemudahan penggunaan saat uji coba
	Kemudahan penggunaan untuk kebutuhan sehari-hari

Pemberian nilai terhadap indikator aspek-aspek pengujian dilakukan dengan meminta tanggapan pengguna terhadap pernyataan yang sesuai dengan indikator aspek pengujian. Skala penilaian menggunakan nilai angka dengan rentang nilai satu (1) sampai enam (6). Tabel 5.34 menjelaskan hubungan skala pengujian dengan tanggapan pengguna terhadap pernyataan uji.

Tabel 5.34 Keterangan Skala Indikator Aspek Pengujian

Skala	Keterangan
1	Sangat tidak setuju
2	Tidak setuju

3	Kurang setuju
4	Cukup setuju
5	Setuju
6	Sangat setuju

5.3.2 Daftar Penguji Aplikasi

Terdapat lima orang penguji seperti yang telah dijelaskan pada bab sebelumnya. Daftar nama penguji aplikasi dapat dilihat pada Tabel 5.35.

Tabel 5.35 Daftar Penguji Aplikasi

Nomor	Nama	Pekerjaan
1	Ilyas Bintang Prayogi	Mahasiswa
2	Vinsensia S. Zega	Mahasiswa
3	Shafly Naufal A.	Mahasiswa
4	Aufar Rizqi	Mahasiswa
5	Wildan Lutfi S.F.S.	Mahasiswa

5.3.3 Hasil Uji Coba Oleh Pengguna

Berikut ini akan dijelaskan hasil pengujian pada setiap aspek-aspek pengujian. Nilai dari setiap inidikator ditentukan dengan menghitung nilai rata-rata dari nilai yang diberikan dari setiap penguji terhadap indikator tersebut. Untuk menentukan keterangan dari nilai rata-rata setiap indikator, nilai rata-rata tersebut dibulatkan ke bilangan bulat terdekat untuk kemudian dicocokan dengan keterangan skala.

5.3.3.1 Hasil Penilaian Aspek Kenyamanan Aplikasi

Hasil pengujian untuk indikator aspek kenyamanan aplikasi dijelaskan pada Tabel 5.36.

Tabel 5.36 Penilaian Indikator Aspek Kenyamanan Aplikasi

Penguji	Nilai Indikator		
	Kenyamanan tampilan antar muka	Kenyamanan menggunakan sentuhan sebagai metode masukan	Kenyamanan melihat hasil terjemahan
Ilyas	4	5	4
Vinsensia	4	6	5
Shafly	6	5	5
Aufar	4	6	5
Wildan	6	4	4
Rata-Rata	**4.80**	**5.20**	**4.60**
Pembulatan Rata-Rata	**5**	**5**	**5**

Hasil pengujian menunjukkan bahwa berdasarkan rata-rata penilaian indikator, penguji merasa puas dengan setiap indikator dari aspek kenyamanan aplikasi.

5.3.3.2 Hasil Penilaian Aspek Performa Aplikasi

Hasil pengujian untuk indikator aspek performa aplikasi dijelaskan pada Tabel 5.37. Hasil pengujian menunjukkan bahwa berdasarkan rata-rata penilaian indikator, penguji merasa puas dengan setiap indikator dari aspek performa aplikasi.

5.3.3.3 Hasil Penilaian Aspek Kemudahan Pengunaan Aplikasi

Hasil pengujian untuk indikator aspek kemudahan penggunaan aplikasi dijelaskan pada Tabel 5.38.

Tabel 5.37 Penilaian Indikator Aspek Kenyamanan Aplikasi

Penguji	Nilai Indikator	
	Performa Aplikasi	Kecepatan Mendapatkan Hasil Terjemahan
Ilyas	4	5
Vinsensia	6	5
Shafly	5	5
Aufar	4	5
Wildan	5	6
Rata-Rata	**4.80**	**5.20**
Pembulatan Rata-Rata	**5**	**5**

Tabel 5.38 Penilaian Indikator Aspek Kemudahan Penggunaan Aplikasi

Penguji	Nilai Indikator	
	Kemudahan penggunaan saat uji coba	Kemudahan penggunaan untuk kebutuhan sehari-hari
Ilyas	4	4
Vinsensia	5	6
Shafly	5	5
Aufar	5	5
Wildan	5	4
Rata-Rata	**4.80**	**4.80**
Pembulatan Rata-Rata	**5**	**5**

Hasil pengujian menunjukkan bahwa berdasarkan rata-rata penilaian indikator, penguji merasa puas dengan setiap indikator dari aspek performa aplikasi.

5.4 Evaluasi

Subbab ini membahas mengenai evaluasi terhadap pengujian-pengujian yang telah dilakukan. Dalam hal ini, evaluasi menunjukkan data rekapitulasi dari hasil pengujian fungsionalitas dan pengujian non-fungsionalitas yang telah dilakukan sebelumnya.

5.4.1 Evaluasi Pengujian Fungsionalitas

Evaluasi pengujian fungsionalitas dilakukan untuk mengetahui bagaimana kinerja dari setiap komponen aplikasi serta mengetahui faktor-faktor yang mempengaruhi hasil pengujian tersebut. Evaluasi yang dilakukan adalah sebagai berikut:

1. Dari hasil pengujian pengambilan Area ROI untuk data uji buatan, terlihat bahwa untuk data uji kata satuan, hasil sudah cukup baik di setiap variabelnya. Namun, untuk beberapa kasus masih terdapat kesalahan dalam pengambilan ROI. Gambar 5.7 menunjukkan contoh kesalahan yang terjadi pada pengujian pengambilan area ROI pada data satuan.

Gambar 5.7 Kesalahan Pengambilan ROI pada Data Uji Kata Satuan

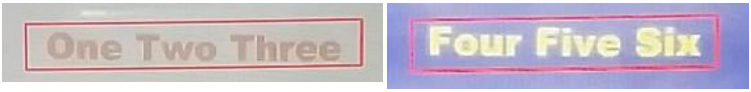
Dari contoh kesalahan pada Gambar 5.7, terlihat bahwa permasalahan yang terjadi adalah wilayah ROI yang terambil tidak melingkupi gambar kata secara utuh dan hanya

mengambil sebagian dari gambar kata. Gambar 5.8 menunjukkan hasil deteksi tepian horizontal *Iy* untuk data uji pada Gambar 5.7.

Gambar 5.8 Hasil Deteksi Tepian Horizontal (*Iy*) untuk Data Uji pada Gambar 5.7

Dari hasil deteksi tepi pada Gambar 5.8, terlihat bahwa kesalahan dapat terjadi akibat perbedaan jarak antar huruf antara "y" dan "r" dan antara "o" dan "p" Terlihat bahwa jarak antara huruf "o" dan "p" terlihat lebih luas. Hal ini dapat menunjukkan bahwa jarak huruf "o" dan "p" tidak tertangani oleh aplikasi. Mengacu pada algoritma pencarian lebar kata, jarak "o" dan "p" lebih dari dua kali jarak antar huruf yang didapatkan oleh aplikasi, dalam hal ini jarak antara "y" dan "r". Hal ini menunjukkan perlunya penanganan ruang kosong jarak antar huruf yang lebih baik dan mampu menyesuaikan jarak antar huruf yang berbeda-beda.

Dari hasil pengujian data uji buatan kata banyak , terdapat kata-kata yang lebih sulit dikenali oleh aplikasi, yaitu "Two","Three", "Five", dan "Seven". Pada kata-kata tersebut, kesalahan lebih banyak terjadi pada ukuran kata yang lebih kecil (28 point). Kesalahan yang terjadi adalah aplikasi tidak mampu mengenali kata "Three" dan "Seven" untuk beberapa kasus, sementara untuk kata "Two" dan "Five", ROI yang terambil melebihi wilayah yang seharusnya. Gambar 5.9 menunjukkan kesalahan pada kata "Two" dan "Five" pada ukuran 28 point. Gambar 5.10 menunjukkan hasil deteksi tepian horizontal *Iy* untuk kata "Two", "Three, "Five", dan "Seven"



(a) (b)

Gambar 5.9 Kesalahan Pengambilan ROI pada Data Uji Kata "Two"(a) dan "Five"(b) pada Ukuran 28 Point

(a) (b) (c) (d)

Gambar 5.10 Hasil Deteksi Tepian Horizontal (*Iy*) untuk Data Uji Kata "Two" (a), "Three" (b), Five (c), dan Seven (d) pada Ukuran 28 Point

Dari Gambar 5.10 terlihat bahwa terdapat huruf-huruf yang tepiannya saling berimpitan atau saling tumpang tindih antara yang satu dengan yang lain. Hal ini menyebabkan jarak antar huruf menjadi sulit untuk ditemukan. Untuk kasus kata "Two" dan "Five" pada gambar 5.6, hal tersebut menyebabkan aplikasi menganggap jarak dengan kata "One" dan "Four" sebagai jarak antar huruf dari "Two" dan "Four". Hasil deteksi tepi yang demikian dapat disebabkan karena jarak kamera dengan gambar kata, di mana kamera tidak dapat mengenali jarak antar huruf dengan jelas. Selain itu, jarak antar huruf dari jenis *font* Arial Black yang tidak sama ikut menjadi faktor kesalahan tersebut. Untuk mengatasi hal ini, diperlukan penanganan yang lebih baik terhadap data deteksi tepi, salah satunya dengan melakukan *preprocessing* yang lebih baik terhadap data deteksi tepi atau *frame* input. Selain itu, penanganan jarak antar huruf untuk jenis *font* dengan jarak antar huruf yang tidak sama atau berdempetan juga perlu dilakukan untuk menangani jenis *font* yang demikian.

Dari hasil pengujian untuk data objek nyata, terlihat bahwa kata “Slim” tidak dapat dikenali oleh aplikasi. Gambar 5.11 menunjukkan hasil deteksi tepian horizontal (*Iy*) untuk kata “Slim”

Gambar 5.11 Hasil Deteksi Tepian Horizontal (*Iy*) untuk Data Uji Kata “Slim”

Dari hasil deteksi tepian horizontal, terlihat masih adanya *noise* pada ruang kosong antar huruf pada kata “Slim” meski sangat sedikit. Namun, algoritma yang digunakan untuk mendeteksi batas-batas horizontal pada area gambar katadapat menyalahartikan *noise* tersebut sebagai tepian huruf, sehingga, dapat menimbulkan kesalahan pencarian batas-batas horizontal. Hal ini dapat diatasi dengan adanya prerocessing lebih lanjut yang mampu menghilangkan *noise* yang tersisa namun tidak memperburuk hasil deteksi tepi.

Selain itu pada pojok kanan bawah dari gambar 5.11 terdapat tepian gambar yang menjorok ke arah kata “Slim”. Meskipun tidak menabrak kata “Slim” secara langsung, tepian tersebut masuk ke baris tempat kata “Slim” berada. Hal tersebut dapat mengganggu hasil deteksi batas-batas horizontal dari kata “Slim”.

2. Dari hasil pengujian pengenalan kata dengan OCR, terlihat bahwa OCR sudah berjalan dengan cukup baik dimana untuk setiap kasus pengujian dengan wilayah ROI yang terambil dengan baik, OCR mampu mengenali setiap kata yang terdapat dalam area ROI tersebut. Namun, terdapat kesalahan

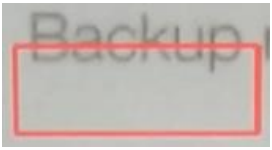
pengenalan untuk kasus pengujian kata “Nine” pada data uji buatan kata banyak.

3. Dari hasil pengujian penerjemahan kata, terlihat bahwa untuk kata-kata pada data uji buatan sudah dapat diterjemahkan dengan baik. Namun pada data uji objek nyata, masih terdapat kata-kata yang tidak dapat diterjemahkan seperti “PLUS” dan “Portable” pada data uji objek nyata. Hal ini terjadi karena Yandex. Translate API masih belum memliki kata dalam Bahasa Indonesia untuk kata-kata tertentu dalam bahasa Inggris.
4. Dari hasil pengujian penggambaran hasil penerjemahan, terlihat bahwa untuk setiap kasus pengujian dengan pengambilan ROI dan pengenalan kata dengan OCR yang sukses, hasil dapat tergambar pada layar aplikasi dengan baik.
5. Dari hasil pengujian pelacakan ROI pada kondisi diam, pada pengujian menggunakan data uji buatan terlihat bahwa niai pengujian cukup baik. Pada pengujian menggunakan data uji objek nyata, terlihat bahwa terjadi kegagalan pada kata “Backup”, “made”, dan “easy”, diluar dari kata “Slim” yang tidak dapat terdeteksi oleh aplikasi. Ketiga kata tersebut memiliki jenis *font* dan warna yang sama. Tabel 5.39 menunjukkan data bin pada histogram HSV ROI Kata “Backup” dengan ukuran 28 point dengan warna merah di atas latar belakang putih dengan frekuensi lebih dari nol. Gambar 5.12 menunjukkan gambar dari kegagalan yang dimaksud. Tabel 5.40 menunjukkan data bin pada histogram HSV hasil pelacakan kata “Backup” pada Gambar 5.12 dengan frekuensi lebih dari nol.

Tabel 5.39 Data Bin pada Histogram HSV ROI Kata “Backup” dengan Frekuensi Lebih dari Nol

Bin Histogram			Frekuensi
H	S	V	
3	0	0	154
3	0	1	447

4	0	0	575
4	0	1	11286



Gambar 5.12 Kegagalan Pelacakan Kata "Backup" pada Kondisi Diam

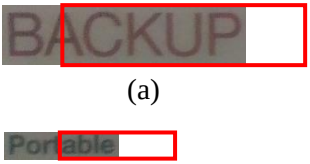
Tabel 5.40 Data Bin pada Histogram HSV Hasil Pelacakan ROI Kata "Backup" pada Gambar 5.12 dengan Frekuensi Lebih dari Nol

Bin Histogram			Frekuensi
H	S	V	
4	0	1	12462

Perhitungan derajat kesamaan dari histogram ROI dengan histogram daerah pelacakan yang gagal pada Gambar 5.12 menggunakan rumus Histogram Correlation memberikan nilai 0.999226. Hal ini menunjukkan bahwa kedua histogram tersebut memiliki tingkat kemiripan yang tinggi. Tabel 5.39 dan 5.40 menunjukkan bahwa sebagian besar piksel pada kedua gambar tersebut berada pada bin histogram yang sama, yaitu bin (4,0,1). Hal ini menunjukkan bahwa histogram tidak mencerminkan perbedaan yang signifikan antara daerah gambar dengan piksel berwarna gelap kehitaman dengan daerah dengan piksel yang lebih terang. Untuk mengatasi hal ini, diperlukan bin histogram yang lebih banyak pada *channel* Value, yang mengatur tingkat kecerahan warna, sehingga aplikasi dapat dengan lebih baik membedakan piksel berwarna

gelap menuju hitam, berwarna terang menuju putih, dan diantaranya.

6. Dari hasil pengujian pelacakan ROI pada kondisi bergerak, pada hasil pengujian menggunakan data uji buatan, terlihat bahwa ukuran kata 28 point memiliki nilai pengujian rata-rata yang lebih rendah dibandingkan dengan nilai rata-rata dari ukuran kata 48 point. Hal tersebut menunjukkan ukuran kata yang lebih besar dapat memberikan hasil pelacakan yang lebih stabil. Hal ini dapat disebabkan oleh jangkauan pelacakan yang ditetapkan sebanyak 50 piksel ke atas, bawah, kiri, dan kanan. Terdapat perbedaan antara pergeseran sebanyak 50 piksel pada ROI yang lebih besar dengan ROI yang lebih kecil. Gambar 5.13 menunjukkan visualisasi dari masalah jangkauan pelacakan tersebut.



(a)

(b)

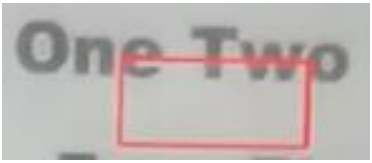
Gambar 5.13 Visualisasi Dampak Ukuran ROI terhadap Jangkauan Pelacakan

Pada Gambar 5.13, terlihat bahwa jangkauan pelacakan relatif terhadap ROI lebih besar pada Gambar 5.13b. dibandingkan dengan gambar 5.13a. Selain itu, pada hasil pengujian kata dengan ukuran 28 point, pelacakan pada warna hitam di atas putih mengalami nilai yang jauh lebih kecil dibandingkan dengan warna lainnya. Hal tersebut disebabkan karena nilai derajat kesamaan yang tinggi antara histogram ROI dengan histogram hasil pelacakan yang gagal pada data uji warna hitamd di atas background putih. Tabel 5.41 menunjukkan data bin pada histogram HSV ROI Kata “One” dengan frekuensi lebih dari nol. Gambar 5.14 menunjukkan contoh kesalahan pelacakan yang terjadi. Tabel 5.42 menunjukkan data bin pada

histogram HSV hasil pelacakan kata “One” pada gambar 5.14 dengan frekuensi lebih dari nol.

Tabel 5.41 Data Bin pada Histogram HSV ROI Kata “One” Ukuran 28 Point dengan Frekuensi Lebih dari Nol

Bin Histogram			Frekuensi
H	S	V	
0	0	0	136
0	0	1	283
1	0	0	26
1	0	1	30
2	0	0	26
2	0	1	56
3	0	0	145
3	0	1	2111
4	0	0	143
4	0	1	5750
5	0	0	76
5	0	1	277
6	0	0	14
6	0	1	183
7	0	0	18
7	0	1	86



Gambar 5.14 Kegagalan Pelacakan Kata “One” Ukuran 28 Point pada Kondisi Bergerak

Tabel 5.42 Data Bin pada Histogram HSV Hasil Pelacakan ROI Kata "One" pada Gambar 5.8 dengan Frekuensi Lebih dari Nol

Bin Histogram			Frekuensi
H	S	V	
0	0	0	153
0	0	1	19
1	0	0	139
1	0	1	4
2	0	0	310
2	0	1	74
3	0	0	507
3	0	1	3052
4	0	0	309
4	0	1	4752
5	0	0	16
5	0	1	8
7	0	0	15
7	0	1	2

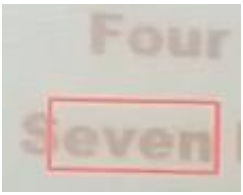
Perhitungan derajat kesamaan dari histogram ROI dengan histogram daerah pelacakan yang gagal pada Gambar 5.14 menggunakan rumus Histogram Correlation memberikan nilai 0.994974. Hal ini menunjukkan bahwa kedua histogram tersebut memiliki tingkat kemiripan yang tinggi. Hal ini menunjukkan bahwa histogram tidak mencerminkan perbedaan yang signifikan antara daerah gambar dengan piksel mayoritas berwarna gelap kehitaman dengan daerah dengan piksel mayoritas berwarna lebih terang.

Pada data uji kata banyak, terlihat bahwa nilai pengujian untuk kata "Four" dan "Five" untuk ukuran kata 28 point berada jauh dibawah kata-kata lainnya. Hal ini terjadi karena kesalahan pelacakan tidak hanya terjadi pada data uji warna hitam di atas

latar belakang putih, tetapi juga pada warna merah diatas latar belakang putih dan warna kuning di atas latar belakang biru. Tabel 5.43 menunjukkan data bin pada histogram HSV ROI Kata “Four” dengan ukuran 28 point dengan warna merah di atas latar belakang putih dengan frekuensi lebih dari nol. Gambar 5.15 menunjukkan contoh kesalahan pelacakan yang terjadi. Tabel 5.44 menunjukkan data bin pada histogram HSV hasil pelacakan kata “Four” pada Gambar 5.15 dengan frekuensi lebih dari nol. Dari Tabel 5.43 dan 5.44, terlihat ada sedikit perbedaan pada nilai histogram antara satu sama lain. Namun, hasil penghitungan derajat kesamaan menggunakan Histogram Correlation masih memberikan hasil yang cukup tinggi, yaitu 0.906388. Hal ini menunjukkan bahwa meskipun secara visual berbeda, derajat kesamaan histogram HSV dapat tetap tinggi

Tabel 5.43 Data Bin pada Histogram HSV ROI Kata “Four” Ukuran 28 Point dengan Frekuensi Lebih dari Nol

Bin Histogram			Frekuensi
H	S	V	
3	0	1	1768
4	0	1	7535
5	0	1	425



Gambar 5.15 Kegagalan Pelacakan Kata “One” Ukuran 28 Point pada Kondisi Bergerak

Tabel 5.44 Data Bin pada Histogram HSV Hasil Pelacakan ROI Kata "Four" pada Gambar 5.8 dengan Frekuensi Lebih dari Nol

Bin Histogram			Frekuensi
H	S	V	
3	0	1	338
4	0	1	4635
4	1	1	91
5	0	1	1266
5	1	1	350

5.4.2 Evaluasi Pengujian Aplikasi Terhadap Pengguna

Evaluasi pengujian aplikasi terhadap pengguna dilakukan untuk mengetahui tingkat kepuasan pengguna pada aspek-aspek pengujian. Tingkat kepuasan aspek pengujian ditentukan dengan mengambil nilai rata-rata dari setiap nilai rata-rata hasil penilaian penguji pada setiap indikatornya. Untuk menentukan keterangan dari nilai rata-rata setiap aspek, nilai rata-rata tersebut dibulatkan ke bilangan bulat terdekat untuk kemudian dicocokan dengan keterangan skala. Tabel 5.45 menunjukkan nilai rata-rata dari setiap aspek pengujian aplikasi.

Tabel 5.45 Hasil Penilaian Pengguna Terhadap Aspek Pengujian

Aspek	Indikator	Rata-Rata Nilai
Kenyamanan Aplikasi	Kenyamanan tampilan antar muka	4.80
	Kenyamanan menggunakan sentuhan sebagai metode masukan	5.20

	Kenyamanan melihat hasil terjemahan	4.60
Rata-Rata		**4.87**
Pembulatan		**5**
Performa Aplikasi	Performa Aplikasi	4.80
	Kecepatan Mendapatkan Hasil Terjemahan	5.20
Rata-Rata		**5.00**
Pembulatan		**5**
Kemudahan Penggunaan Aplikasi	Kemudahan penggunaan saat uji coba	4.80
	Kemudahan penggunaan untuk kebutuhan sehari-hari	4.80
Rata-Rata		**4.80**
Pembulatan		**5**

Dari hasil penghitungan rata-rata terhadap nilai rata-rata dari tiap indikator per aspeknya, penguji merasa puas dengan seluruh aspek pengujian aplikasi.

Penguji juga memberikan evaluasi terhadap aplikasi berupa kritik dan saran. Kritik dan saran dari setiap penguji ditunjukkan pada oleh Tabel 5.46

Tabel 5.46 Kritik dan Saran Penguji terhadap Aplikasi

Nama	Kritik dan Saran
Ilyas Bintang Prayogi	Tingkatkan FPS (*frame* per second)
Vinsensia S. Zega	Perlu dilakukan pengolahan terhadap hasil terjemahan agar huruf tidak kapital semua

Shafly Naufal A.	Memilih kata dapat dilakukan dengan menggambar kotak agar bisa mencakup satu paragraf. Hasil terjemahan lebih baik dipisahkan ke halaman baru
Aufar Rizqi	FPS jangan turun saat aplikasi digunakan
Wildan Lutfi S.F.S.	Buat petunjuk penggunaan agar dapat lebih mudah dipahami

(Halaman ini sengaja dikosongkan)

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas kesimpulan yang dapat diambil dari rumusan masalah dari pembuatan aplikasi dalam tugas akhir ini, serta hasil pengujian terhadap aplikasi. Selain itu, bab ini juga membahas saran untuk pengembangan aplikasi tugas akhir ini kedepannya berdasarkan hasil evaluasi pengujian aplikasi.

6.1. Kesimpulan

Dalam proses pengerjaan tugas akhir mulai dari tahap analisis, desain, implementasi, hingga pengujian didapatkan kesimpulan sebagai berikut:

1. Pengambilan area gambar teks sebagai ROI dilakukan dengan melakukan deteksi tepi horizontal dan vertikal menggunakan filter Sobel. Daerah ROI ditemukan dengan menentukan batas-batas ROI berdasarkan posisi sentuhan pengguna pada layar. Hasil uji coba kotak hitam menunjukkan nilai yang baik, dengan adanya masalah dalam menangani ruang antar huruf dan diperlukannya *preprocessing* untuk mengolah data tepian yang tidak baik
2. Pengenalan teks dilakukan dengan bantuan pustaka Tesseract OCR, sementara penerjemahan teks dari bahsa Inggris ke bahasa Indonesia dilakukan dengan kakas bantu Yandex. Translate API. Hasil uji coba kotak hitam menunjukkan nilai yang baik untuk pengenalan kata dengan OCR. Untuk perjemahan teks, masih terdapat bahasa dalam bahasa Inggris yang tidak mampu diterjemahkan ke bahasa Indonesia.
3. Penambahan grafis pada *frame* masukan dari kamera dengan hasil terjemahan dilakukan dengan membuat sebuah gambar menggunakan objek Mat OpenCV yang berisi hasil terjemahan. Hasil terjemahan digambar pada *frame* masukan di atas gambar teks yang diterjemahkan

berdsarkan posisi gambar teks yang didapatkan dari proses pelacakan.

4. Proses pelacakan ROI dilakukan dengan membandingkan histogram HSV dari ROI dengan histogram HSV dari daerah pelacakan. Pelacakan mencari tingkat kemiripan histogram yang paling tinggi dengan menggunakan rumus histogram correlation. Hasil uji coba kotak hitam menunjukkan nilai yang rendah saat pelacak digunakan dalam kondisi bergerak dikarenakan faktor jangkauan pelacakan dan histogram HSV
5. Pengujian aplikasi dilakukan dengan metode kotak hitam dan dengan pengujian oleh pengguna. Pengujian kotak hitam menguji kemampuan setiap komponen aplikasi menggunakan data uji buatan dan data uji yang diambil dari objek nyata.
6. Berdasarkan pengujian oleh pengguna, penguji merasa puas dengan kenyamanan, performa, dan kemudahan penggunaan aplikasi.

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang, berdasarkan pada hasil perancangan, implementasi dan uji coba yang telah dilakukan:

1. Mengembangkan tahapan *preprocessing* pada hasil deteksi tepi horizontal dan vertikal untuk mengatasi *noise* dan bentuk huruf yang tidak baik.
2. Menggunakan/membuat sistem pelacak lain yang dapat dijalankan dengan baik pada perangkat bergerak Android dalam melakukan pelacakan ROI sehingga pelacakan ROI dapat berjalan dengan lebih baik
3. Menggunakan kamera yang menghasilkan *frame* masukan dengan resolusi yang lebih kecil agar mengurangi konsumsi memori perangkat dan meningkatkan kecepatan aplikasi.

4. Mencoba perangkat bergerak Android lainnya dengan spesifikasi yang lebih baik sebagai perangkat uji untuk mengetahui batasan pengembangan performa aplikasi.
5. Melakukan eksplorasi untuk mencari kemungkinan metode masukan dari pengguna dengan cara yang lain untuk mengetahui metode yang paling efisien dan nyaman digunakan oleh pengguna

(Halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

[1] "Google Terjemahan." [Daring]. Tersedia pada: https://translate.google.co.id/?hl=id. [Diakses: 23-Jun-2018].

[2] "Menerjemahkan dengan ucapan - Komputer - Bantuan Google Translate." [Daring]. Tersedia pada: https://support.google.com/translate/answer/6142468?hl=id&ref_topic=7011659. [Diakses: 23-Jun-2018].

[3] "Menerjemahkan dengan tulisan tangan atau keyboard virtual - Komputer - Bantuan Google Translate." [Daring]. Tersedia pada: https://support.google.com/translate/answer/6142469?hl=id&ref_topic=7011659. [Diakses: 23-Jun-2018].

[4] V. Fragoso, S. Gauglitz, S. Zamora, J. Kleban, dan M. Turk, "TranslatAR: A mobile augmented reality translator," dalam *2011 IEEE Workshop on Applications of Computer Vision (WACV)*, 2011, hlm. 497–502.

[5] J. Gao, H.-S. J. Tsao, dan Y. Wu, *Testing and Quality Assurance for Component-based Software*. Artech House, 2003.

[6] P. Schueffel, "Schueffel (2017) The Concise FINTECH COMPENDIUM.pdf." [Daring]. Tersedia pada: http://www.heg-fr.ch/FR/HEG-FR/Communication-et-evenements/evenements/SiteAssets/Pages/patrick-schuffel/Schueffel%20(2017)%20The%20Concise%20FINTECH%20COMPENDIUM.PDF. [Diakses: 23-Jun-2018].

[7] H.-Q. Le dan J.-I. Kim, "An augmented reality application with hand gestures for learning 3D geometry," dalam *2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2017, hlm. 34–41.

[8] M. Cavallo dan A. G. Forbes, "Digitalquest: a mixed reality approach to scavenger hunts," dalam *2016 IEEE International Workshop on Mixed Reality Art (MRA)*, 2016, hlm. 11–15.

[9] J. Carmigniani dan B. Furht, "Augmented Reality: An Overview," dalam *Handbook of Augmented Reality*, B. Furht, Ed. New York, NY: Springer New York, 2011, hlm. 3–46.

[10] "Android Overview | Open Handset Alliance." [Daring]. Tersedia pada: http://www.openhandsetalliance.com/android_overview.html. [Diakses: 23-Jun-2018].

[11] "Android," *Android*. [Daring]. Tersedia pada: https://www.android.com/. [Diakses: 23-Jun-2018].
[12] "Application Fundamentals," *Android Developers*. [Daring]. Tersedia pada: https://developer.android.com/guide/components/fundamentals. [Diakses: 23-Jun-2018].
[13] "About - OpenCV library." [Daring]. Tersedia pada: https://opencv.org/about.html. [Diakses: 23-Jun-2018].
[14] "OpenCV4Android SDK — OpenCV 2.4.13.6 documentation." [Daring]. Tersedia pada: https://docs.opencv.org/2.4/doc/tutorials/introduction/android_binary_package/O4A_SDK.html.
[15] "What is OCR and OCR Technology." [Daring]. Tersedia pada: https://www.abbyy.com/en-apac/finereader/what-is-ocr/. [Diakses: 23-Jun-2018].
[16] "GitHub - tesseract-ocr/tesseract: Tesseract Open Source OCR Engine (main repository)." [Daring]. Tersedia pada: https://github.com/tesseract-ocr/tesseract. [Diakses: 23-Jun-2018].
[17] *tesseract: Tesseract Open Source OCR Engine (main repository)*. tesseract-ocr, 2018.
[18] R. Smith, "An Overview of the Tesseract OCR Engine," 2007, hlm. 629–633.
[19] R. Theis, *tess-two: Fork of Tesseract Tools for Android*. 2018.
[20] "Translate API — Yandex Technologies." [Daring]. Tersedia pada: https://tech.yandex.com/translate/. [Diakses: 23-Jun-2018].
[21] "Translate API — Troubleshooting — Yandex Technologies." [Daring]. Tersedia pada: https://tech.yandex.com/translate/doc/dg/concepts/faq-api-translate-docpage/. [Diakses: 23-Jun-2018].
[22] "Pricing." [Daring]. Tersedia pada: https://translate.yandex.com/developers/prices. [Diakses: 23-Jun-2018].
[23] "Feature Detectors - Sobel Edge Detector." [Daring]. Tersedia pada: http://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm. [Diakses: 05-Jun-2018].
[24] S. Trambadia dan H. Mayatra, "Food detection on plate based on the HSV color model," dalam *2016 Online International Conference on Green Engineering and Technologies (IC-GET)*, 2016, hlm. 1–6.

[25] S. M. Youssef, "ICTEDCT-CBIR: Integrating curvelet transform with enhanced dominant colors extraction and texture analysis for efficient content-based image retrieval," *Comput. Electr. Eng.*, vol. 38, no. 5, hlm. 1358–1376, Sep 2012.

[26] "OpenCV: Changing Colorspaces." [Daring]. Tersedia pada: https://docs.opencv.org/3.4.1/df/d9d/tutorial_py_colorspaces.html. [Diakses: 05-Jun-2018].

[27] "Image Analysis - Intensity Histogram." [Daring]. Tersedia pada: https://homepages.inf.ed.ac.uk/rbf/HIPR2/histgram.htm. [Diakses: 05-Jun-2018].

[28] "Histogram Comparison — OpenCV 2.4.13.4 documentation." [Daring]. Tersedia pada: https://docs.opencv.org/2.4.13.4/doc/tutorials/imgproc/histograms/histogram_comparison/histogram_comparison.html. [Diakses: 23-Jun-2018].

[29] "Morphology - Opening." [Daring]. Tersedia pada: http://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm. [Diakses: 05-Jun-2018].

[30] "OpenCV: Morphological Transformations." [Daring]. Tersedia pada: https://docs.opencv.org/3.4/d9/d61/tutorial_py_morphological_ops.html. [Diakses: 23-Jun-2018].

[31] "Getting Started with the NDK | Android NDK | Android Developers." [Daring]. Tersedia pada: https://developer.android.com/ndk/guides/. [Diakses: 23-Jun-2018].

[32] "Glow-in-the-Dark Emergency Exit Sign, SKU: S-6449," *SmartSign.com*. [Daring]. Tersedia pada: https://www.smartsign.com/Emergency-Exit-Route/Glow-in-the-Dark/SKU-S-6449.aspx. [Diakses: 11-Jul-2018].

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS

Antonius Kevin Wiguna, lahir di Bogor pada tanggal 15 Juni 1996. Penulis menempuh pendidikan mulai dari TK Mardi Yuana Bogor (2000-2001), TK Tunas Pertiwi Bogor (2001-2002), SD Regina Pacis Bogor (2002-2008), SMP Regina Pacis Bogor (2008-2011), SMA Regina Pacis Bogor (2011-2014), dan sekarang tengah menempuh pendidikan S1 Departemen Informatika Institut Teknologi Sepuluh Nopember. Penulis pernah aktif dalam beberapa organisasi, yaitu menjadi staf departemen Riset dan Teknologi Himpunan Mahasiswa Teknik Computer-ITS 2015-2016, staf departemen internal Keluarga Mahasiswa Katolik St. Ignatius ITS 2015-2016, dan wakil ketua departemen internal Keluarga Mahasiswa Katolik St. Ignatius ITS 2016-2017. Penulis mengambil bidang minat Interaksi Grafika dan Seni (IGS). Penulis dapat dikontak melalui email kevin.wiguna@gmail.com**.**

.